

## Gowin Design Constraints User Guide

SUG101-1.5E,11/23/2018

#### Copyright©2018 Guangdong Gowin Semiconductor Corporation. All Rights Reserved.

No part of this document may be reproduced or transmitted in any form or by any denotes, electronic, mechanical, photocopying, recording or otherwise, without the prior written consent of GOWINSEMI.

#### Disclaimer

GOWINSEMI<sup>®</sup>, LittleBee<sup>®</sup>, Arora<sup>™</sup>, and the GOWINSEMI logos are trademarks of GOWINSEMI and are registered in China, the U.S. Patent and Trademark Office, and other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders, as described at www.gowinsemi.com. GOWINSEMI assumes no liability and provides no warranty (either expressed or implied) and is not responsible for any damage incurred to your hardware, software, data, or property resulting from usage of the materials or intellectual property except as outlined in the GOWINSEMI Terms and Conditions of Sale. All information in this document should be treated as preliminary. GOWINSEMI may make changes to this document at any time without prior notice. Anyone relying on this documentation should contact GOWINSEMI for the current documentation and errata.

## **Revision History**

Date	Version	Description	
03/03/2016	1.09E	Initial version published.	
01/30/2018	1.1E 1.2E	<ol> <li>Reload function added in FloorPlanner.</li> <li>I3C mode added attribute in IO Constraint.</li> <li>Multiple VCCs added and merged into one VCC in package view of FloorPlanner.</li> <li>Position constraint removed from clock constraint.</li> <li>Supports GW1N-2B/GW1N-4B/GW1N-6ES/GW1N-9ES/GW1NR-9ES/GW1 NS-2/GW1NS-2C;</li> </ol>	
07/05/2018		<ol> <li>Remove the differential IO positive and negative poles in the Package View of FloorPlanner and IO Color;</li> <li>Change the constraint color in the Chip ArrayFloor of Planner;</li> <li>Change the Arribute into ALL/LUT/REG in the Reserve constraint of FloorPlanner;</li> <li>The TCL Console is removed in the FloorPlanner and changed it into the Message;</li> <li>Remove the check in the FloorPlanner;</li> <li>In the set_false_path, set_min_delay, set_max_delay and set_multicycle_path, keep one "from" and remove the "rise_from" and "fall_from"; keep one "to" and remove the "rise_to""fall_to".</li> </ol>	
09/21/2018	1.3E	The description in appendix A updated.	
10/26/2018	1.4E	<ol> <li>GW1NZ-1/GW1NSR-2C supported;</li> <li>The description in appendix A updated.</li> </ol>	
11/23/20181. GW1NSR-2 supported; 2. GW1N-6ES, GW1N-9ES, and GW1NR-9ES deleted.			

## Contents

Contentsi
List of Figuresv
List of Tablesx
1 About This Guide1
1.1 Purpose1
1.2 Supported Products 1
1.3 Related Documents 1
1.4 Abbreviations and Terminology2
1.5 Support and Feedback2
2 Introduction3
3 Physical Constraints5
3.1 Function
3.2 Start FloorPlanner5
3.3 Create and Open the Constraint File7
3.3.1 Create Constraints File7
3.3.2 FloorPlanner Output Constraint Files9
3.3.3 Open the Constraint File10
3.4 FloorPlanner View 11
3.4.1 Menu
3.4.2 List View
3.4.3 Package View
3.4.4 Chip Array View
3.4.5 Constraint View
3.5 Tcl Console View45

3.6 Create Constraints - Drag Mode45
3.6.1 Set I/O Constraints Position45
3.6.2 Create Primitive Constraints
3.6.3 Create Group Constraints50
3.6.4 Resource Reservation Creation
3.6.5 Create Clock Assignment53
3.6.6 Quadrant Constraints Creation
3.6.7 Create Hclk Constraints
3.6.8 Vref Constraints Creation
4 Timing Constraint
4.1 Static Timing Analysis (STA) Overview61
4.1.1 Basic Model of Static Timing Analysis61
4.1.2 Timing Analysis Terminology62
4.1.3 Timing Analysis Path62
4.1.4 Common Timing Checks63
4.2 Timing Constraint Editor64
4.2.1 Overview
4.2.2 Open Editor64
4.2.3 Create and Open the Constraint File65
4.2.4 Editor
4.2.5 Timing Constraint
4.3 Edit the SDC File70
4.4 Edit Timing Constraint71
4.4.1 Clock Constraint
4.4.2 I/O Delay
4.4.3 Path Constraint
4.4.4 Operation Conditions Constraints
4.4.5 Timing Report
4.4.6 Save and Export91
4.5 Priority of Timing Constraints91
5 Timing Analysis View92

Appendix A Physical Constraints Syntax Definition	
A.1 I/O Constraints	
A.2 PORT Attributes Constraints	
A.3 Primitive Constraints	100
A.4 Primitive Group Constraints	103
A.5 Resource Reservation	105
A.6 Relative Group Constraints	106
A.7 Vref Constraints	107
A.8 Quadrant Constraints	107
A.9 Clock Assignment	108
A.10 Hclk Constraints	110
Appendix B Timing Constraints Syntax Definition	111
B.1 Clock Constraints	111
B.1.1 create_clock	111
B.1.2 create_generated_clock	113
B.1.4 set_clock_uncertainty	116
B.1.5 set_clock_groups	117
B.2 I/O Constraints	118
B.2.1 set_input_delay	118
B.2.2 set_output_delay	120
B.3 Path Constraints	122
B.3.1 set_max_delay   set_min_delay	122
B.3.2 set_false_path	123
B.3.3 set_multicycle_path	124
B.4 Operation Conditions Constraints	126
B.5 Timing Report	127
B.5.1 report_timing	127
B.5.2 report_high_fanout_nets	128
B.5.3 report_route_congestion	129
B.5.4 report_min_pulse_width	130
B.5.5 report_max_frequency	131

B.5.6 report_exceptions
B.1.2 create_generated_clock
B.1.4 set_clock_uncertainty138
B.1.5 set_clock_groups
B.2 I/O Constraints
B.2.1 set_input_delay 139
B.2.2 set_output_delay141
B.3 Path Constraints
B.3.1 set_max_delay   set_min_delay143
B.3.2 set_false_path
B.3.3 set_multicycle_path146
B.4 Operation Conditions Constraints
B.5 Timing Report
B.5.1 report_timing
B.5.2 report_high_fanout_nets
B.5.3 report_route_congestion
B.5.4 report_min_pulse_width
B.5.5 report_max_frequency 152
B.5.6 report_exceptions

## List of Figures

	Figure 3-1 Start FloorPlanner From Tools Menu	. 6
	Figure 3-2 Process View	. 6
	Figure 3-3 Open New Physical Constraints File	. 7
	Figure 3-4 Create Physical Constraints File	. 8
	Figure 3-5 Manual Writing the Physical Constraint File	. 8
	Figure 3-6 New FloorPlanner	. 9
	Figure 3-7 Save Output File	. 10
	Figure 3-8 Open the Physical Constraint File	. 10
	Figure 3-9 FloorPlanner View	. 11
	Figure 3-10 File	. 12
	Figure 3-11 Select Device	. 13
	Figure 3-12 Tools	. 13
	Figure 3-13 Primitive Finder View	. 14
	Figure 3-14 New Primitive Group	. 15
	Figure 3-15 Correct Primitive Group Interface	. 16
	Figure 3-16 Invalid Position	. 17
	Figure 3-17 Invalid Position	. 17
	Figure 3-18 New Relative Group	. 18
	Figure 3-19 Group Interface in Correct Relative Position	. 18
	Figure 3-20 Reserved Constraint	. 19
	Figure 3-21 Timing Constraint	. 20
	Figure 3-22 Quadrant Constraints (GW1N)	. 20
	Figure 3-23 Quadrant Constraints (GW2A)	. 21
	Figure 3-24 Hclk Constraints	. 21
	Figure 3-25 Vref Constraints	. 22
SUG	101-1.5E	

Figure 3-26 Select One Hit Drag Primitive	23
Figure 3-27 Select One Hit Drag Location	23
Figure 3-28 One Hit Drag Constraints	24
Figure 3-29 Find View2	25
Figure 3-30 View	26
Figure 3-31 Windows	26
Figure 3-32 Help2	26
Figure 3-33 Project View	27
Figure 3-34 Netlist View	28
Figure 3-35 Show Bus and Non-Bus Conjunctively2	29
Figure 3-36 Hierarchical View	30
Figure 3-37 Timing Path	30
Figure 3-38 Netlist Right-Click Functions	31
Figure 3-39 Package View	32
Figure 3-40 Right-Click Function of Package View	33
Figure 3-41 Differential Pair Display	33
Figure 3-42 Chip Array View	35
Figure 3-43 Grid Constraint	36
Figure 3-44 Macro Element Constraint	36
Figure 3-45 Primitive Constraint	37
Figure 3-46 Right-Clicking on the Chip Array	38
Figure 3-47 Show Place View	39
Figure 3-48 Highlighted Timing Path	39
Figure 3-49 I/O Constraint View	41
Figure 3-50 Primitive Constraints View	41
Figure 3-51 Group Constraints View	42
Figure 3-52 Resource Reservation View	42
Figure 3-53 Clock Constraint View	43
Figure 3-54 Quadrant Constraints View	44
Figure 3-55 Hclk Constraints View	44
Figure 3-56 Vref Constraints View	45

Figure 3-57 Message View	45
Figure 3-58 Drag to Chip Array to set I/O Constraints	46
Figure 3-59 Drag to Package View and Set I/O Constraints	47
Figure 3-60 Set I/O Constraints Location Constraint by F3	48
Figure 3-61 Drag to Chip Array and Set Primitive Constraints	49
Figure 3-62 Set Location of Primitive Constraints by Pressing F3	50
Figure 3-63 Right-Click Menu of Group Constraints Editor	50
Figure 3-64 Create Primitive Group Constraints	51
Figure 3-65 Create Relative Group Constraints	51
Figure 3-66 Create Resource Reservation	52
Figure 3-67 Copy Location in Chip Array to Set Resource Reservation Constraint	53
Figure 3-68 Create Clock Assignment Constraints	54
Figure 3-69 Modify Clock Assignment Constraints	54
Figure 3-70 Create Quadrant Constraints	55
Figure 3-71 Modify Quadrant Constraints	56
Figure 3-72 Create Hclk Constraints	57
Figure 3-73 Modify Hclk Constraints	58
Figure 3-74 Create Vref Constraints	58
Figure 3-75 Vref Rename Check	59
Figure 3-76 Drag to Chip Array to Modify Location	59
Figure 3-77 Drag to Package View to set Vref Constraint Location	60
Figure 4-1 Basic Model of Timing Analysis	62
Figure 4-2 STA Four Timing Paths	63
Figure 4-3 Process View	65
Figure 4-4 Open New Timing Constraints File	66
Figure 4-5 Create Timing Constraint File	66
Figure 4-6 Open Timing Constraint File	67
Figure 4-7 Timing Constraint Editor	68
Figure 4-8 Netlist Tree View	68
Figure 4-9 Constraint Edit View	69
Figure 4-10 Open Timing Constraint View in Menu	69

Figure 4-11 Right Click to Open the Timing Constraint View	. 70
Figure 4-12 Edit SDC File	. 70
Figure 4-13 Create Clock Constraint	. 71
Figure 4-14 Objects	. 72
Figure 4-15 Create Clock Constraint	. 73
Figure 4-16 Clock Constraint List	. 73
Figure 4-17 Right-click Contents of Clock List	. 74
Figure 4-18 Create Generated Clock Constraints	. 75
Figure 4-19 Right-click Contents of Clock List	. 75
Figure 4-20 Create Clock Latency	. 76
Figure 4-21 Create Clock Uncertainty	. 77
Figure 4-22 Create Clock Group Constraints	. 78
Figure 4-23 Clock Group Member List	. 78
Figure 4-24 Create I/O Delay Constraints	. 80
Figure 4-25 Create False Path Constraints	. 81
Figure 4-26 Create Max/Min Delay Constraint	. 81
Figure 4-27 Create Multicycle Path Constraint	. 83
Figure 4-28 Create Operating Conditions Constraints	. 83
Figure 4-29 Report Timing Creation Interface	. 84
Figure 4-30 Report Timing Interface	. 85
Figure 4-31 Report High Fanout Nets Create Interface	. 86
Figure 4-32 Report High Fanout Nets Interface	. 86
Figure 4-33 Report Route Congestion Create	. 87
Figure 4-34 Report Route Congestion Interface	. 87
Figure 4-35 Report Min Pulse Width Interface	. 88
Figure 4-36 Report Min Pulse Width Interface	. 88
Figure 4-37 Report Exception Interface	. 89
Figure 4-38 Report Max Frequency Interface	. 89
Figure 4-39 Report Exception Interface	. 90
Figure 4-40 Report Exception Interface	. 90
Figure 4-41 View Timing Reporting Interface	. 91

Figure 5-1 Posp File Settings	. 93
Figure 5-2 Read .posp File	. 94
Figure 5-3 Read Timing Constraint File	. 94
Figure 5-4 Module Operation of Highlighting Key Path	. 95
Figure 5-5 Highlight Module of Key Path in Red	. 95
Figure 5-6 Key Path Signal Flow	. 96
Figure 5-7 Location After Adjustment	. 97

## **List of Tables**

Table 1-1: Abbreviations and Terminology	2
--	---

## **1** About This Guide

## **1.1 Purpose**

This guide predominantly describes the physical constraints and timing constraints of Gowin products. It also introduces the interface usage and grammar rules that underpin the Gowin physical constraints and timing constraints tools. It is designed to help users quickly implement physical constraints and timing constraints. The software screenshots and the supported products listed in this guide are based on Windows 1.8.1 Beta. As the software is subject to change without notice, some information may not remain relevant and may need to be adjusted according to the software that is in use.

## **1.2 Supported Products**

The information in this guide applies to the following products:

- GW2A series of FPGA products: GW2A-18 and GW2A-55
- GW2AR series of FPGA products: GW2AR-18
- GW1N series of FPGA products: GW1N-1, GW1N-2, GW1N-2B, GW1N-4, GW1N-4B, GW1N-6, and GW1N-9
- GW1NR series of FPGA products: GW1NR-4, GW1NR-4B, and GW1NR-9
- GW1NS series of FPGA products: GW1NS-2, GW1NS-2C
- GW1NZ series of FPGA products: GW1NZ-1
- GW1NSR series of FPGA products: GW1NSR-2C, GW1NSR-2

## **1.3 Related Documents**

The latest user guides are available on the Gowin website. Refer to the related documents at <u>www.gowinsemi.com</u>:

- GW2A series of FPGA Products Data Sheet
- GW2AR series of FPGA Products Data Sheet

- GW1N series of FPGA Products Data Sheet
- GW1NR series of FPGA Products Data Sheet
- GW1NS series of FPGA Products Package and Pinout
- GW1NZ series of FPGA Products Data Sheet
- GW1NSR series of FPGA Products Data Sheet
- Gowin Software User Guide
- GW1N series of FPGA Products Package and Pinout
- GW1NR series of FPGA Products Package and Pinout
- GW2A series of FPGA Products Package and Pinout
- GW2AR series of FPGA Products Package and Pinout
- GW1NS series of FPGA Products Package and Pinout
- GW1NZ series of FPGA Products Package and Pinout
- GW1NSR series of FPGA Products Package and Pinout

## **1.4 Abbreviations and Terminology**

Table 1-1 shows the abbreviations and terminology used in this guide.

Table 1-1: Abbreviations and Terminology

Abbreviations and Terminology	Full Name/Meaning
FPGA	Field Programmable Gate Array
I/O	Input/Output

## **1.5 Support and Feedback**

Gowin Semiconductor provides customers with comprehensive technical support. If you have any questions, comments, or suggestions, please feel free to contact us directly using the information provided below.

Website: www.gowinsemi.com

E-mail: support@gowinsemi.com

Tel: +86 755 8262 0391

# **2**Introduction

The Gowin Constraint Editor includes the Physical Constraints Editor (FloorPlanner) and the Timing Constraints Editor. Users can employ these editors to execute physical constraints and timing constraints.

The Gowin FloorPlanner tool was developed by Gowin for placement and routing and physical constraints editing. It can read and modify the attributes and location of I/O, Primitive, block (B-SRAM, DSP), Net, and Group, etc. It also supports the generation of new layout and constraints files based on the associated configuration. The I/O attribute information and location information on primitives and modules are specified in the provided files. The Gowin FloorPlanner provides a simple and quick layout and constraint editing function that was designed to effectively improve the efficiency with which users write physical constraint files.

The timing constraint editor tool can be used to create and modify timing constraint files, which can provide efficient network list lookup and effectively improve the efficiency with which users write timing constraint files.

The timing constraint editor incorporates the following features:

- Calculate delay value according to a specific timing model through the signal transmission path and then compare this value with the expected value to determine whether the user design meets the timing requirements;
- Provide user with an opportunity to improve design work rate and stability by timing constraint;
- Provide the default basic clock and timing analysis for the cross-clock domain, including two timing reports formats: HTML and text (HTML by default);
- Support the common timing constraints (including clock constraint, I/O port constraint, and timing path constraint), and the common timing report directives;
- Support the priority commonly used;
- Inspect setup time, hold time, recovery time, removal time, and MPW

by default, and customize the timing report content according to the timing report instructions.

# **3**<sub>Physical Constraints</sub>

Users can employ the Gowin FloorPlanner tool to create and modify physical constraint files. These files contain tabulated constraint editing and efficient network list lookup data that can increase the efficiency with which physical constraint files are written.

## **3.1 Function**

The Gowin FloorPlanner features include the following:

- Read user design file and constraint file and output constraint file;
- I/O port, primitive, and net constraints displayed in user design files;
- Create, edit, and modify constraint information;
- Chip array grid mode, macrocell mode, and primitive mode;
- Package view based on package;
- Synchronously display chip array and package view;
- Display and differentially display real-time information on a constrained position;
- Set location information by dragging;
- Support for one-hit drag and one-key generating constraints;
- I/O port attributes configuration, batch configuration;
- Display and edit clock assignment;
- Check legality of constraints information.

### 3.2 Start FloorPlanner

Two methods can be employed to launch the FloorPlanner:

1. Click "IDE > Tools" and open "FloorPlanner", as shown in Figure 3-1.

<u>T</u> oo	ls <u>W</u> indow <u>H</u> elp
ę	Start Page
s	Synplify Pro
8	IP Core Generator
	FloorPlanner
×	Timing Constraints Editor
	Simulation
	Gowin Analyzer Oscilloscope
Æ	Options

#### Figure 3-1 Start FloorPlanner From Tools Menu

2. After setting RTL project, double-click "FloorPlanner", as shown in Figure 3-2.

**Figure 3-2 Process View** 

Process	₽×
📗 Design Summary	
4 📝 User Constraints	
📕 FloorPlanner	
🔀 Timing Constraints Editor	
4 🥝 Synthesize (Synplify Pro)	
📄 Synthesis Report	
Netlist File	
4 🥝 Place & Route	
📄 Place & Route Report	
🧾 Timing Analysis Report	
📄 Ports & Pins Report	
📄 Power Analysis Report	
🕌 Program Device	
Design Process	

#### Note!

- If the Gowin FloorPlanner is required for constraints, the netlist file should be added first.
- When the Gowin FloorPlanner is started from the tools menu, the user will need to load the netlist file through the "File > new" option.
- When starting the Gowin FloorPlanner using the second approach, the netlist file will be automatically loaded.

## 3.3 Create and Open the Constraint File

Physical constraint files are required for the constraint position and attribute of I/O, Primitive/Net location, etc. in the project. Files can be physically restrained in the following two ways:

- 1. Manual writing;
- 2. Output the constraint file using Gowin FloorPlanner.

## 3.3.1 Create Constraints File

The constraint file can be created by the following steps:

- 1. Click "File  $\geq$  New" to open the "New" dialog box.
- 2. Select "Physical Constraints File," as shown in Figure 3-3.

#### Figure 3-3 Open New Physical Constraints File

🐝 New	S ×	
⊿ File	5	]
	Controlog File	
	Physical Constraints File	
	GAO Config File	
	GPA Config File	
	Memory Initialization File	
Create	a Physical Constraints file.	
	OK Cancel	
	OK Cancel	)

#### Note!

You can also open a "New" file in the following two ways:

- Using short cut "Ctrl+N";
- Clicking on the "New" icon on the toolbar.
- Click on the "OK" button, and the dialog box shown in Figure 3-4 will appear.

#### Figure 3-4 Create Physical Constraints File

🐝 New Phys	sical Constraints File		8 ×
Name:	led_water_1hz		. est 🔻
Create in:	E:\gowin=board\gw1n=4_case\led_water_1hz\src		Browse
	📝 Add to current project		
		OK	Cancel

- Name: Input the name of the physical constraint file. Ensure it is saved with the suffix of CST or .ucf.
- Create in: Click on the "Browse..." button and select the location in which the constraint file will be saved. The default location in which files are saved is the src folder.
- Add to current project: If this option is checked, the constraint file will be automatically added to the project.

Once the new physical constraints file has been added, it will open, and the user can subsequently edit it according to the Gowin physical constraint syntax. The constraint file can be manually written, as shown in Figure 3-5.

#### Figure 3-5 Manual Writing the Physical Constraint File

🐝 Gowin FPGA Designer - [led_water_1hz.	.cst *]						
🄀 File Edit Project Tools Window Help							
🗋 📂 🔚 🎼   🖶 🔺 🍾	Þ. 🛍   👪						
Design 🗗 🗙	1 //***********************************						
🔺 🧰 led_water_1hz	2						
@ GW1N-4-LQFP144-5	3 // IO Constraints 4 IO LOC "clk" 25						
	5 IO LOC "rst n" 34						
4 📂 Source Files	6 IO LOC "led[0]" 27						
📄 led_water_1hz.v	7 IO LOC "led[1]" 26						
4 🛅 Physical Constraint	8 IO_LOC "led[2]" 97						
led_water_1hz.cst	9 IO_LOC "led[3]" 24						
	10						
4 🛅 Timing Constraint	11						
led_water_1hz.sdc							
🦰 GAO Config File							

## 3.3.2 FloorPlanner Output Constraint Files

The Gowin FloorPlanner can output the new physical constraint and the modified physical constraint files. The process for doing this is outlined below:

- 1. Start the FloorPlanner as described in 3.2 Start FloorPlanner;
- From the File menu, choose "File> New..." to open the "New" dialog, as shown in Figure 3-6;
- 3. Enter the netlist file, select the device type, and click "OK".

#### Figure 3-6 New FloorPlanner

🐝 New	? <mark>x</mark>
Design File:	
test/CHUNXIANG/floorplan/fp_test/src/bsram.vm	Browse
Constraint File:	
	Browse
Part:	
GW1N-4 LQFP144	Select Part
OK	Cancel

#### Note!

Start the FloorPlanner using the first method described in 3.2.

You can also open a "New" project by using one of the following two ways:

- Use the shortcut: "Ctrl+N".
- Click on the "New" icon in the toolbar.

Please refer to the steps outlined below to create the physical constraints in the FloorPlanner:

- 1. Distribute the pin location by dragging.
- 2. Click "Save" to output the constraint files.
- 3. If required, modify the file name in the "Save" dialog box, as shown in Figure 3-7.

Figure	3-7	Save	Output	File
0	-			-

🗱 Save		×
◯◯ ▽ 📴 ► gowin ►	▼ 4 搜索 gowin	م
文件名(N): led_water_1hz.cst		•
保存类型(T): Constraint File(*.cst)		•
💌 浏览文件夹(B)	保存(S)	取消

#### 3.3.3 Open the Constraint File

The steps required to open the constraint file are as follows:

- 1. In the ID view, click "File  $\,>\,$  Open".
- 2. Open the "Open File" dialog box, as shown in Figure 3-8.

#### Figure 3-8 Open the Physical Constraint File

<ul> <li>织 ▼ 新建文件夹</li> <li>( 收藏夹</li> <li>↓ 下號</li> <li>■ 桌面</li> <li>1 最近访问的位置</li> </ul>	名称 「 fpga_project_12.cst	修改日期	NATU		•== •	0	
<ul> <li>( 收藏夹</li> <li>) 下载</li> <li>三 桌面</li> <li>2) 最近访问的位置</li> </ul>	fpga_project_12.cst	修改日期	No TTU			•	
■ 桌面 9월 最近访问的位置			类型	大小			
3 最近访问的位置		2017/12/22 11:25	CST 文件	1 KB			
	fpga_project_12.sdc	2017/12/20 14:17	SDC 文件	0 KB			
	📄 test.v	2017/12/22 11:09	V 文件	1 KB			
🚡 2345下载							
库 =							
₩ 视频							
■ 图片							
□ 迅雷下载							
计算机							<
▲ 本地磁盘 (C:)							
■ 本地磁盘 (C:) □ 本地磁盘 (D:)							
□ 本IB磁盘 (D:) ▼							-
文件名(	(N):			✓ All Files (*) (	* *)	<b>,</b>	
				All Files (*) (*			
				Analysis Des	ign Cons		c)
				FPGA Design			• •
				GOWIN FPG GOWIN Firm			°.gprj)
				GOWIN Osc			ao)
				GOWIN Phy			
				GOWIN Pow GOWIN Tim			
				HTML Docu			
				Memory Init	tialization		
				Plain Text Fil VHDL File (*.			

#### Note!

Alternatively, you can also open the "Open File" dialog box via the following two ways:

- Using the shortcut: Ctrl + O.
- By clicking on the "Open" icon on the toolbar.
- Select the directory in which the physical constraint file is located and open the selected file.

## 3.4 FloorPlanner View

Create or open the FloorPlanner (including the netlist file), as shown in Figure 3-9.

The interface includes the menu, List, Chip Array, Package View, Message, and various constraint editing views, etc.

#### Figure 3-9 FloorPlanner View

🐝 FloorPlanner									x
<u>File Tools Vie</u>	w <u>H</u> elp								
i 🗋 📂 🔗 I	ピ 🥥 🔇	<u>&gt;</u> 🔍							
List	ē×	Chip Array 🔀	Package View 🖂						
Project Netli	st								
4 🗟 test									
<ul> <li>Ports (</li> <li>Primitiv</li> </ul>									
Nets (8)				-000000000					
Module				00000000		00000000000			
🖻 📄 Timing	Paths			o]]]]]]]]]] o]]]]]]]]]]]]]]]]]]	*********				
				00000000	*********				
				-00000000	505055566 55555555				
					**********				
				°000000000	*******				
				-000000000		0000000000			
I/O Constraints									₽×
Port	Direction	Diff Pair	Location	Bank	Exclusive	ІО Туре	Drive	Pull Mode	^
din1[0]	input		F8,D9	3	False	LVCMOS18	N/A	UP	
din1[1]	input		drag or type t		False	LVCMOS18	N/A	UP	
	· · ·	III			e 1	11/21/0212	N174	115	•
Tel C···· I/O	Const Pr:	imitive Const…	Group Const	Resource Reser…	Clock Assi	Quadrant Const	Helk Co	nst… Vref Cor	ist
									d

#### 3.4.1 Menu

The FloorPlanner includes the "File", "Tools", "View", and "Help" options.

#### File

The file view is shown in Figure 3-10.

#### Figure 3-10 File

🐝 F	loorPlanner		
<u>F</u> ile	<u>T</u> ools View	<u>H</u> elp	
	<u>N</u> ew	Ctrl+N	
	Open	•	Design
1	Save	Ctrl+S	Constraint
1	Save As		Posp
0	Reload		Timing Paths
	Vfc Configuration	'n	
	Quit		

- New: New project, add user design, constraint and set chip information, etc., as shown in Figure 3-6.
- Open: Open the user netlist file, constraint file, device layout information file, or timing path file.
- Reload: Reload the cst file after modifying and saving it to a disk or project.
- Save: Save the modified information of the current constraint and override the original constraint file information.
- Save As: Output the modified information of the current constraint information to the file specified by the user. The file name of the network list is used as the constraint file name by default; however, this can be modified by the user;
- Quit: Close the Gowin FloorPlanner.

#### Note!

The Select Part option is used to select the chip, package information to support all Gowin FPGA devices, as shown in Figure 3-11.

#### **Figure 3-11 Select Device**

🐝 Select Part	2 X
Family:	GW1N 💌
Device:	GW1N-4 💌
Package:	PBGA256 💌
SpeedGrade:	5 🔹
	OK Cancel

#### Tools

The tools view is shown in Figure 3-12. The tools functions are as follows:

- Input constraint information.
- Support One Hit Drag, Find, and Constraint Legality Check functions.

Display various constraint information in real time in the constraint editor, and the corresponding location information will be displayed in the Chip Array and Package View. The menu functions are as follows:

🐳 FloorPl	anner		
File Too	ls View Help		
	Select Primitives Select Nets		
List Proj	Create Group	۲	Create Primitive Group
	Select Utilization Reserve Resources		Create Relative Group
C	Clock/Control Assignment		
C	Select Dcs/Dqce Select Hclk		
C	Define Vref Driver		
	One Hit Drag	F3	
	Find Constraint Check		
	Update Constraints Using Posp		

Figure 3-12 Tools

• Select Primitives.

- a) Select the Primitive option to create the corresponding constraint, click menu, and the dialog box will open, as shown in Figure 3-13.
- b) Find primitives by "Name" or "Type", and select the corresponding primitive.
- c) Click "OK" to generate the constraint information.

#### Note!

- The constraint information is displayed in the "Primitive Constraints" area at the bottom of the main interface.
- The user can set the location information by entering or dragging in the editing view.
- The location is highlighted in yellow in the Chip Array.

#### **Figure 3-13 Primitive Finder View**

_	. Primitive		
Fi	lter		
Na	me: *		
Type: *			•
_	Name	Туре	
1	cnt8_13_cZ	INS_LUT4	E
2	cnt8_14_cZ	INS_LUT4	
3	cnt8_15_cZ	INS_LUT4	
4	cnt8_17_cZ	INS_LUT4	
5	cnt8_18_cZ	INS_LUT2	
6	cnt8_18_sx_cZ	INS_LUT4	
7	cnt8_1_cZ	INS_LUT4	
8	cnt8_22_cZ	INS_LUT4	
9	cnt8_22_sx_cZ	INS_LUT4	

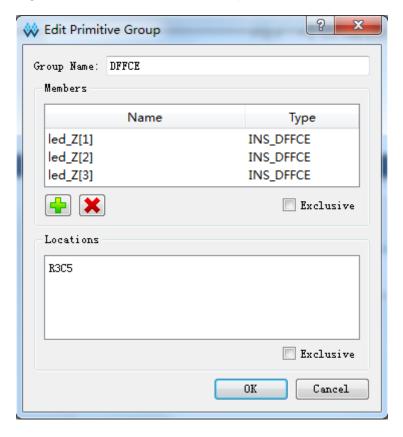
The Create Group menu includes the Create Primitive Group and Create Relative Group options. The functions are as follows:

- Create Primitive Group:
  - a) Click "Create Primitive Group", and the dialog box will open, as shown in Figure 3-14.

Wew Primitive Group	8 23
Group Name:	
Members	
Name	Туре
	Exclusive
Locations	
	Exclusive
(	OK Cancel

#### Figure 3-14 New Primitive Group

- b) Set the group name, primitive, location, and group exclusive information according to your needs.
- c) Add and remove the primitives by clicking on the "🛨" and "💌" icons respectively. The primitive group information is shown in Figure 3-15.



#### Figure 3-15 Correct Primitive Group Interface

#### Note!

- Group name, primitive, and group location are required.
- The location information for the group can be inputted in the following ways:
  - Input manually.
  - Before creating the group constraint, copy the location and paste it into "New Primitive Group > Location" in the "Chip Array".
- 3. After creating the primitive group, click"OK", and the tool will check the syntax of the group location information.
  - a) If the location information is not acceptable, a prompt dialog box will appear, as shown in Figure 3-16 and Figure 3-17. Modify the location information accordingly.

**Figure 3-16 Invalid Position** 

🐝 Synta	ax Error
<b></b>	Invalid locations: "R3C", Refer to: (R)(\d+ \[\d+:\d+])(C)(\d+ \[\d+:\d+])([0-3][AB]?)? (IO[TBLR])(\d+ \[\d+:\d+])([AB])? [a-zA-Z]+[0-9]+ [0-9]+
	ОК

**Figure 3-17 Invalid Position** 

🐝 Error	×
	Location is invalid
	ОК

b) If no error exists, click "OK", and the available location will be displayed in the chip array.

See the newly generated group constraint in the Group Constraints area that is located at the bottom of the main view.

Double click on the group constraint and the dialog box will open for further editing, as shown in Figure 3-14.

- Create the relative group:
  - a) Create a group constraint with a relative position. Click on the menu option and the dialog box will open, as shown in Figure 3-18.
  - b) Set the group name, primitive, and the relative position information that corresponds to each primitive.
  - c) Add and remove the primitive by clicking on the "+" and " =" icons respectively. The Relative Group Constraints that are created successfully will be displayed, as shown in Figure 3-19.

Figure 3-18 New Relative Group

🐝 New Relative Group	? ×
Group Name: Members	
Primitive Name	Relative Location
📥 🗶	
	OK Cancel

Figure 3-19 Group Interface in Correct Relative Position

Edit Relative Group		
Group Name: LUT2 Members		
Primitive Name	Relative Location	
cnt8_18_cZ cnt_3_cZ[13]		
🛖 🗶	OK Cancel	

#### Note!

- Group name, primitive, and relative location are required.
- The location information for the group can be inputted in the following ways:
  - Input manually.
  - Before creating the group constraint, copy the location and paste it into "New Relative Group > Relative Location" in the "Chip Array".
  - d) Click the "OK" button after configuring the group. The constraint information will appear in the "Group Constraints" area at the bottom of the main interface.

Double click the group constraint, and the dialog box will appear for further editing, as shown in Figure 3-19.

#### **Reserve Resources:**

- 1. Click to create a new constraint in the "Resource Reservation" at the bottom of the main interface.
- 2. Double-click "Position" to input the constraint position.
- 3. Double-click "Attribute" to set the properties for the reserved position, as shown in Figure 3-20.

#### Note!

The "Name" attribute is used to distinguish between different constraints. The name cannot be modified.

#### Figure 3-20 Reserved Constraint

R	esource Reservation		
	Name	Locations	Attribute
	reserve_0	type location	GENERAL 🔻
			GENERAL LUT REG WLUT

#### **Clock/Control Assignment**

Create a Clock constraint that limits the number of constraints, and check it via a constraint validity check. Click the menu, and the dialog box will open, as shown in Figure 3-21. The operations are as follows:

- 1. Click "<sup>1</sup> to select the corresponding net.
- 2. Select "BUFG", "BUFG[0]~[7]", and "BUFS" via the "Type" drop-down list.
- 3. Configure the signal via the "CE", "CLK". After configuring, click "OK" to generate the constraint information in the "Clock Assignment". Double click in the edit view. The dialog box for editing will open, as shown in Figure 3-21.

Figure 3-21 Timing Constraint

🐝 Cloo	k Assignment	? ×
Net		
Туре	BUFG	•
-Sign	al	
🗖 C	E	
🗖 🗖 C	LK	
	OGIC	
🗖 S	R	
		OK Cancel

#### Select Dcs/Dqce

Create quadrant constraints for DCS and DQCE. Specify the instance to the specific quadrant according to the chip quadrant distribution, as shown in Figure 3-22 and Figure 3-23.

The related operations are as follows:

- Select the corresponding DCS/DQCE devices by clicking on the "
   icon;
- 2. Configure the quadrant positions via the checkboxes under "Position";
- 3. Click "OK", the constraint information is generated in the "Quadrant Constraints" constraint editing window at the bottom of the main interface. In the editing window, double-click to open the constraint dialog box again for editing.

#### Note!

Quadrant constraints cannot be added if there is no existing DCS/DQCE device.

The constraint editing window of "Quadrant Constraints" of GW2A-18, GW2AR-18, and GW2A-55 series is shown in Figure 3-23.

Figure 3-22 Quadrant Constraints (GW1N)

🐳 Quadrant Constraints	5 ×
Instance Position	
🔲 LEFT	RIGHT
	OK Cancel

#### Figure 3-23 Quadrant Constraints (GW2A)

🐝 Quadrant Co	onstraints		? ×
Instance Position			
TOPLEFT	TOPRIGHT	BOTTOMLEFT	BOTTOMRIGHT
		ОК	Cancel

#### Select Hclk

Create HCLK constraints on primitives and specify the constraint locations around the chip, as shown in Figure 3-24.

The related operations are as follows:

- 1. You can select the device by clicking on the "
- 2. Configure the quadrant positions using the checkboxes that appear under the "Position" heading.
- 3. Click "OK", the constraint information is generated in the "Quadrant Constraints" constraint editing window at the bottom of the main interface. In the editing window, double-click to open the constraint dialog box again for editing.

#### Note!

- HCLK constraints cannot be added if there is no existing compatible device.
- The positions that are available vary according to the different devices in the project.

#### Figure 3-24 Hclk Constraints

W Hclk Constraints	? <b>**</b>
Instance	
Position	
TOPSIDE[0]	TOPSIDE[1]
BOTTOMSIDE[0]	BOTTOMSIDE[1]
LEFTSIDE[0]	LEFTSIDE[1]
RIGHTSIDE[0]	RIGHTSIDE[1]
	OK Cancel
	UN Cancel

#### Define the Vref Driver

Create a new Vref Driver for configuring the Vref of the I/O port, and click the menu option to create a new constraint in "Vref Constraints" at the bottom of the main interface, as shown in Figure 3-25.

#### Figure 3-25 Vref Constraints

	ref Constraints									
	Name		Locations		IO TYPE					
l	vref_driver_0		drag to set		VREF1_DRI\	/ER				
	Tcl Console	I/O Constraints	Primitive Constraints	Group C	onstraints	Resource Reservation	Clock	k Assignm		

#### Note!

- Specify the Vref constraint position by dragging.
- Modify the Vref name by double-clicking.

#### One Hit Drag

Quickly create primitive and I/O constraint information using the following steps:

1. In Netlist, select primitive or port, as shown in Figure 3-26.

List		₽×				
Project	Netlist					
🔺 🔡 led	🛚 📴 led_water_1hz					
Þ 🚞	Ports (6)					
⊿ 📄	Primitives (81)	E				
	Cnt8_13_cZ (INS_LUT4)					
	Cont8_14_cZ (INS_LUT4)					
	Cont8_15_cZ (INS_LUT4)					
	Cont8_17_cZ (INS_LUT4)					
	Cont8_18_cZ (INS_LUT2)					
	Cnt8_18_sx_cZ (INS_LU	T4)				
	Cnt8_1_cZ (INS_LUT4)					
	Cnt8_22_cZ (INS_LUT4)					
	Cnt8_22_sx_cZ (INS_LU	T4)				
	Cnt8_cZ (INS_LUT4)					
	Cnt8_sx_cZ (INS_LUT4)					
	Cnt8_sx_sx_cZ (INS_LU	T4) 👻				
•		•				

#### Figure 3-26 Select One Hit Drag Primitive

2. In Chip Array, select one or more location information, as shown in Figure 3-27.

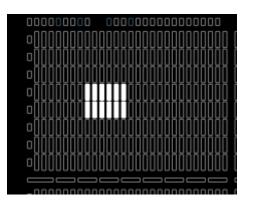


Figure 3-27 Select One Hit Drag Location

3. Click "Tools > One Hit Drag" or press "F3" to generate the constraint information directly, as shown in Figure 3-28.

## Figure 3-28 One Hit Drag Constraints

### Note!

Select the rectangle location by pressing "Ctrl" and clicking the left mouse button.

## Find

Quickly find primitives, the I/O port, and edit the corresponding constraint information on the primitive or port by accessing the right-click menu. Click the menu, and the dialog box will appear, as shown in Figure 3-29.

The related operations are as follows:

- 1. Find by selecting "All Primitive" or "All Port".
- 2. Select the corresponding item, right-click "Edit \* \* \* Constraint" to edit the constraint information at the bottom of the main interface.

Find	nitives	? <mark>×</mark>							
All Prim All Port									
Name:	*	]							
Type:	*								
	Name	Туре							
1 clkdiv	/_inst	INS_CLKDIV							
2 dqce	inst	INS_DQCE							
3 ides4	L_inst	INS_IDES4							
4 oser4	4_inst	INS_OSER4							
5 sub_i	nst/clkdiv_inst	INS_CLKDIV							
6 sub_i	nst/dqce_inst	INS_DQCE							
7 sub_i	nst/ides4_inst	INS_IDES4							
8 sub_i	nst/oser4_inst	INS_OSER4							
		,							
		Close							

### Figure 3-29 Find View

## Update the Constraints Using Posp

Update the constraint information as the location information in the device layout information file.

## View

As shown in Figure 3-30, the view option contains commands that can be used to control the toolbars, display windows, zoom in/out/fit chip array, and package view. A brief introduction to the sub-menus is presented below:

- Toolbars: Display shortcut buttons in toolbars.
- Windows: Display different windows, as shown in Figure 3-31.
- Zoom Out: Zoom out on the chip array or package view.
- Zoom In: Zoom in on the chip array or package view.
- Zoom Fit: Zoom in on the chip array or package view according to the view size.

### Figure 3-30 View

Viev	w Help				
	Toolbars	•	~	File	
	Windows	•	✓	Tools	
۹	Zoom Out	F7			
e,	Zoom In	F8			
٩	Zoom Fit	F6			

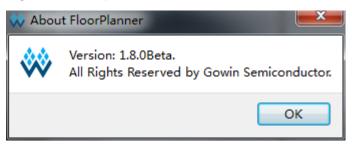
## Figure 3-31 Windows

<u>File</u> <u>T</u> ools View	w <u>H</u> elp			
	Toolbars	•	1	
Image: Second	Chip Array			
	F7	<	Package View	
_	Zoom In	F8	$\checkmark$	List
		F6	<	Tcl Console
			$\checkmark$	I/O Constraints
🛛 🗅 🛅 Module	e		$\checkmark$	Primitive Constraints
			$\checkmark$	Group Constraints
			$\checkmark$	Resource Reservation
			$\checkmark$	Clock Assignment
			$\checkmark$	Quadrant Constraints
			$\checkmark$	Vref Constraints
			✓	Hclk Constraints

## Help

The help section contains information about the software version and associated copyright information. Click on the "About" option and the prompt box will appear, as shown in Figure 3-32.

### Figure 3-32 Help



# 3.4.2 List View

The list view includes project and netlist, which can display device information, user design, and path information for the constraint file, Netlist information, etc.

## Project

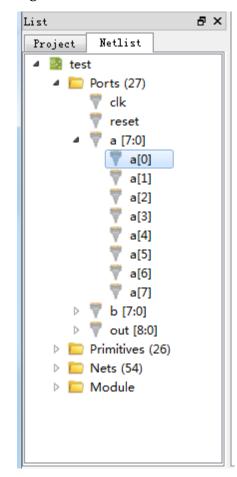
The project interface is shown in Figure 3-33. This displays the chip information related to the project, such as Series, Device, Package, and the design files constraint files, device layout information file, and timing path file entered by the user.

Figure 3-33 Project View

List		₽ ×
Project	Netlist	
Series:	GW1N	
Device:	GW1N-4	
Package:	PBGA25	6
Design:	C:/Vse	ers/root/Documents/f***
Constraint	t: C:∕Vse	rs/root/Documents/f***
Posp:		
Timing Pat	ths:	

## **Netlist View**

As shown in Figure 3-34, the netlist displays the Ports, Primitives, Nets, Module, Timing paths, and corresponding quantity in the tree structure.



### Figure 3-34 Netlist View

### Note!

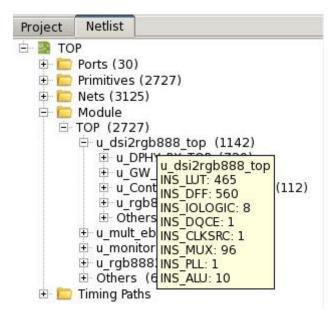
- The names of the port, primitives, etc. in the full path and order are displayed alphabetically default.
- Display port and net via combining bus and non-bus, as shown in Figure 3-35.

List		₽×
Project	Netlist	
🔺 🗟 tes	t	
🔹 🔺 📄	Ports (27)	
	🔻 clk	
	🔻 reset	
▲	🔻 a [7:0]	
	🔻 a[0]	
	🝸 a[1]	
	🝸 a[2]	
	🝸 a[3]	
	🔻 a[4]	
	💙 a[5]	
	▼ a[6]	
⊳	▼ a[7]	
	♥ b [7:0] ♥ out [8:0	
	Primitives (	
	Nets (54)	20)
	Module	
	Woodle	

Figure 3-35 Show Bus and Non-Bus Conjunctively

• Display the module in the hierarchy and show the number of instances in each module when placing the mouse on a module in the module list, as shown in Figure 3-36.

Figure 3-36 Hierarchical View



 The sequential path is listed in the slack time from small to large, as shown in Figure 3-37.

Figure 3-37 Timing Path

List		6	×
Project	Netlist		
	V_PLL Ports (12) Primitives Nets (22) Module GW_PLL ( Timing Pai Setup	(9) 9)	
	<ul> <li>Path 1</li> <li>Path 2</li> <li>Path 3</li> <li>Path 4</li> <li>Path 5</li> <li>Path 6</li> <li>Path 6</li> <li>Path 7</li> <li>Path 8</li> <li>Hold</li> <li>Path 1</li> <li>Path 2</li> <li>Path 4</li> <li>Path 4</li> <li>Path 5</li> <li>Path 6</li> </ul>	(Slack:5.515 Arrive:9.108 Require:14.6 (Slack:5.745 Arrive:8.878 Require:14. (Slack:5.927 Arrive:8.696 Require:14. (Slack:5.933 Arrive:8.691 Require:14.6 (Slack:6.163 Arrive:8.46 Require:14.6 (Slack:6.787 Arrive:7.836 Require:14.6 (Slack:7.181 Arrive:7.443 Require:14.4) (Slack:7.181 Arrive:7.212 Require:14.4) (Slack:7.411 Arrive:7.212 Require:14.4) (Slack:1.266 Arrive:5.381 Require:4.1) (Slack:1.323 Arrive:5.438 Require:4.1) (Slack:1.661 Arrive:5.776 Require:4.1) (Slack:1.928 Arrive:6.043 Require:4.1) (Slack:1.942 Arrive:6.057 Require:4.1) (Slack:1.999 Arrive:6.114 Require:4.1)	

## **Netlist View**

Netlist incorporates a right-click menu with the following functions:

- 1. Highlight the corresponding constraint location in the chip array.
- 2. Edit the corresponding constraint information.

## Note!

If the current primitive, net or port has no position constraint, the highlighted function is not available, as shown in Figure 3-38.

Figure 3-38 Netlist Right-Click Functions

List		É	×
Project	Netlist		
🔺 🗟 DV	/I_gen		*
D 📄	Ports (48)		
⊿ 📄	Primitives (	(310)	
	🗋 clk_50N	1.un2_clkcnt2lto11 (INS_LUT4)	
	🗋 clk_50N	1.un2_clkcnt2lto11_1_cZ (INS_LUT2)	
	🗋 clk_50N	1.un2_clkcnt2lto15_2_cZ (INS_LUT4)	
	🔁 clk_50N	1.un2_clkcnt2lto17 (INS_LUT4)	
	🗋 clk_50N	1.un2_clkcnt2lto23_1_cZ (INS_LUT2)	
	🚺 clk_50N	1.un2 clkcnt2lto23 3 cZ (INS_LUT4)	
	🗋 clk_50N	A. Highlight IT4)	
	Clkcnt1	3	
	Clkcnt1	3 Edit Constraint	
	Clkcnt1	_3_cZ[13] (INS_LUT3)	
	Clkcnt1	_3_cZ[14] (INS_LUT3)	
	Clkcnt1	_3_cZ[15] (INS_LUT3)	
	Clkcnt1	_3_cZ[17] (INS_LUT3)	
	Clkcnt1	_3_cZ[19] (INS_LUT3)	Ŧ
•		4 III	

# 3.4.3 Package View

As shown in Figure 3-39, the package view displays the package information for each chip based on the chip package information, displaying pins for I/O, power supply, ground, etc. When placing the mouse in a position, the I/O information of the location will be shown, including I/O type, bank, and LVDS.

Figure 3-39 Package View

Array 🔀	Package	View	×																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	<b>15</b>	<b>16</b>		
	Α	≯	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	≱	А	
	В	₽	Ŧ	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	Ŧ	₽	В	
	С	₽	Þ	Ŧ	₽	₽	₽	₽	₽	₽	Þ	₽	Þ	₽	Ŧ	Þ	Þ	с	
	D	₽	₽	₽	Ŧ	≱	₽	₽	₽	₽	₽	₽	≱	Ŧ	Þ	٩		D	
	Е	₽	₽	₽	≱	Ŧ	₽	₽	₽	₽	₽	₽	Ŧ	≱	Þ	Þ	₽	E	
	F	₽	₽	₽	₽	₽	Ŧ	₽	₽	₽	₽	Ŧ	Þ		٩	⊅		F	
	G	₽	₽	₽	₽	₽	₽	≉	≱	≉	≯		٩			⊅		G	
	Н	₽	₽	₽	₽	₽	₽	≱	Ŧ	Ŧ	≯		٩		٩	٩		н	
	J	₽	₽	₽	₽	₽	₽	≱	Ŧ	Ŧ	≯		٩			٩		J	
	К	₽	⊅	₽	₽	₽	₽	≯	≯	≱	≯		٩	₽	٩	٩		к	
	L	₽	₽	₽	₽	₽	÷	₽	₽	₽	₽	Ŧ	٩		₽		$\blacksquare$	L	
	М	₽	₽	€	≱	Ŧ	₽	₽	₽	₽	₽	₽	Ŧ	≱				М	
	Ν	₽	₽	₽	Ŧ	≯	₽	₽	₽	₽	Þ	Þ	≯	Ŧ	Þ	٩	٩	N	
	Р	₽	₽	Ŧ	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	Ŧ			Р	
	R	₽	Ŧ	₽	₽	₽	₽	Þ	₽	₽	₽	₽	₽	₽	₽	÷	٩	R	
	т	≯	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	₽	≯	т	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		

User I/O and multiplexing I/O, power, ground, and dedicated pins are also marked with different symbols and colors. The different symbols and colors used for the various pins are defined below:

- Red means BANK0 pins; Green means BANK1 pins.
- Blue means BANK2 pins; Yellow means BANK3 pins.
- "Data means single-ended and differential I/O in BANK0; the filling color changes according to the BANK.
- "Diamond and a multiplexing I/O in BANK1; the filling color changes according to the BANK.
- "
   <sup>\*</sup>
   <sup>\*</sup>
- "📥" means VSS; the filling color does not change.
- "<sup>1</sup><sup>2</sup>" means dedicated pins.
- "환" means NC.

As shown in Figure 3-40, the package view incorporates a right-click menu. The relevant functions are as follows:

- Support zoom in/out view, and show differential IO pairs.
- Switch to display between the Top and Bottom View. The Top View is the default view.

Figure 3-40 Right-Click Function of Package View

	Zoom In	F8
	Zoom Out	F7
	Zoom Fit	F6
	Show Differential IO Pairs	
۲	Top View Bottom View	

In the package view, select "Show Differential IO Pairs" by right-clicking and selecting "Display difference pair" from the menu. As shown in Figure 3-41, the items in a differential pair are connected by a red line.

Figure 3-41 Differential Pair Display

Chip Array 🗵	Pac	kage V	iew 区	3															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		*
	А	≯	N	R	R	R		R		₽	₽	₽	N	8	N	N	≯	Α	
	В	R	÷	P	N		N	₽	i P	¢		ø	6	ø	P	Ŧ		В	
	С	R	N	÷	P	N	P	N	•		P		ø	ø	Ŧ	Ø	P	С	
	D	₿	₩	₽	Ŧ	≯	R	₽	R	ø	₽	ø	≯	Ŧ	R		₽	D	
	Е	R	₽	₿	≯	Ŧ	ø	N	₽				÷	≯			₽	Е	
	F	₿	₩	₽	B	R	÷	B	ø	P	P	÷	R	₽	₿	₿		F	
	G	R	₽	₿	₿	₽	N	≯	≯	≱	≉	R				₿	₽	G	
	Н	N	₩	₽	B	P	₿	≱	÷	÷	≯	₿		₽	₿	♥		н	=
	J	₽	₽	₿	N	R	N	\$	÷	÷	≯	R		1		₽	₽	J	
	K	N	₿	₽	R	₽	N	≱	≱	≯	≯	₿		₽	₽			К	
	L	₽	₽	₿	N	N	÷	ø		ø	₿	÷		1		R	₽	L	
	Μ	N	R	₽	≯	÷	P	7 <b>P</b>	R	Ø	R	₿	÷	≯	₽			М	
	Ν	N	P	N	÷	≱	ø		۲	N	ø	N	≱	÷		R	₽	Ν	
	Ρ	N	₽	÷	₽	N	₽	8	₽	8	₽	8	R	₿	÷	R		Р	
	R	P	÷	N	ø	•	۶	•	Þ	R		R	Ø	R	R	÷		R	
	Т	≯	B	P		P		P		•	N	•	N		N	9	\$	Т	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		-
•										11									 •

The package view displays the I/O port constraint position, which can be set in the following two ways:

- Drag the constraint location of the I/O port in the package view.
- Drag the I/O Port to the package view from the "Netlist" or "I/O Constraints" at the bottom.

Note!

- When dragging, the mouse will display the name of the port that is being dragged.
- "Similar indicates that the dragged port cannot be placed at the desired location.

# 3.4.4 Chip Array View

The chip array view of the FloorPlanner is shown in Figure 3-42. The chip array shows the distribution of IOB, CFU, and DSP according to the chip ranks and real-time displays all constraint locations and support functions of enlarging, reducing, repeating location, suspending, dragging and dropping, etc.

The IOB is all IOB locations of the bare chip, and is distinguished by different colors:

- White corresponds to the I/O location.
- Red is the position of an unpackaged I/O.
- The blue IOB in the for GW2AR-18, GW1NR-4, GW1NR-4B, GW1NR-9, GW1NR-9ES, and GW1NSR-2C series indicates that the configure embedded SDRAM at I/O.

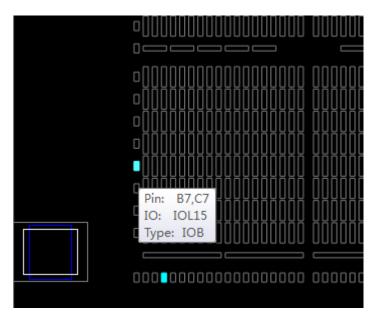
Chip Array 🔀	Package View 🗵	
	◦]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]	
	□0000000000000000000000000000000000000	
		L

Figure 3-42 Chip Array View

Chip array includes grid mode, macro element mode, and primitive mode.

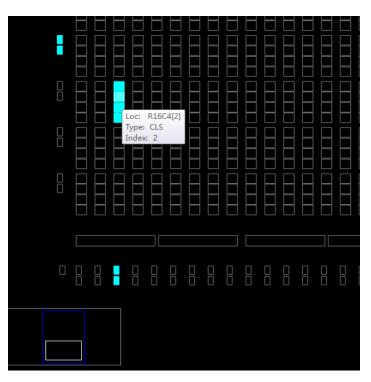
• Grid mode: Macroscopically display constraint position in grid scale, as shown in Figure 3-43.

## Figure 3-43 Grid Constraint



 Macro element mode: Display constraint position in CLS, block, etc., as shown in Figure 3-44.





• Primitive mode: Display constraint position in reg, lut, etc., as shown in Figure 3-45.

		Loc: Type: Index:	R16C4[2][A] LUT	
_				

## Figure 3-45 Primitive Constraint

The chip array supports the following drag/drop functions:

- Drag from array: Applies to the constraint position occupied by the particular primitive. Can be dragged optionally in Array.
- Drag/drop from Netlist to Array to generate and specify constraints.
- Drag/drop from Edit to Array to specify constraints.

The Built-in Chip View is used to display the current view position relative to the entire chip in real time. The Chip Array view follows when dragging the white box in the chip. The Chip Array distinguishes between constraint type and displays the constraint position in different colors. The meaning of each color is as follows:

- White: Display the constraint position in the selection state or highlighted state.
- Dark blue: Display the position information for the reserved constraints, indicating that the position cannot be occupied again.
- Yellow: Display the constrained position in the primitive constraint occupied by the fix position.
- Light blue: Display the I/O and Primitive locations in the range.

The chip array incorporates a right-click menu. The functions are as follows:

- Zoom in/out view.
- Show constraints in view, place view, and multi-view.
- Show I/O connection.
- Convert place to constraints.
- Eliminate highlighting.
- Remove and copy location information.

### Note!

- If grid, block, reg, or lut, etc. are selected in the view, the "Copy Location" is available on the right-click menu.
- If no grid is selected, the function is not available, as shown in Figure 3-46.
- Select an area by holding "Ctrl" and clicking on the left mouse button. Copy the location of the selected area by right-clicking and selecting the "Copy the Location" option. The replicated location can be pasted directly into any constraints in the editor.

Chip Array 🔀 🛛 Package View 🛛	×	
<u>e</u>	Zoom In	F8
	Zoom Out	F7 000000
0	Zoom Fit	F6 1000000
$\checkmark$	Show Constraints View	
	Show Place View	► jj000000
	Show Multi-View	
	Show In-Out Connection	
	Show In Connection	
	Show Out Connection	<u>]000000</u> 0
	Convert Place To Constraints	
	Unhighlight All	
	Move	)0000000
	Copy Location	000000

### Figure 3-46 Right-Clicking on the Chip Array

The Show Place View also shows the Lut and Reg density, as shown in Figure 3-47:

- ALL Instance: Shows all Instance places. Light green indicates less than five, green indicates six to ten, and dark green indicates more than ten.
- Only Lut: Shows all lut places. Light green indicates less than two, green indicates three to four, and dark green indicates more than four.
- Only Dff: Shows all reg places. Light green indicates less than two, green indicates three to four, and dark green indicates more than four.

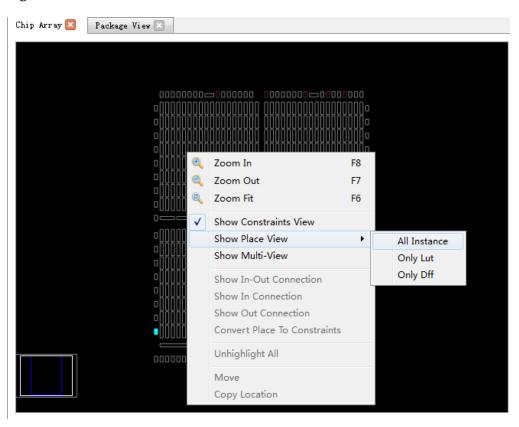


Figure 3-47 Show Place View

The chip array view also highlights the timing paths, as shown in Figure 3-48.

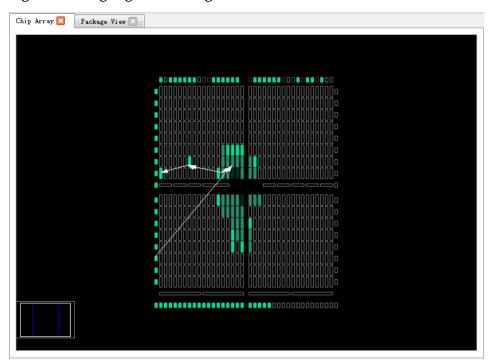


Figure 3-48 Highlighted Timing Path

# 3.4.5 Constraint View

The constraint view includes I/O constraints, primitive constraints, group constraints, etc., which are used to display the detailed information of each constraint and provide the constraints editor function and the drag/drop position function. A brief introduction to each view is provided below.

## I/O Constraints

The I/O constraint view is as shown in Figure 3-49. The available functions are as follows:

- Display all properties and constraint information of the I/O port in the user design, such as direction, bank, I/O type, pullMode of port, etc.
- Provide the functions for editing constraint location, attribute, etc.
- Change the constraint information by dragging, double-clicking, etc.

## Note!

- Set the I/O location information by dragging or double-clicking.
- Display the I/O name when dragging/dropping the I/O.
- When dragging the I/O into the chip array, the placed position lightens, and the color of the implacable position brightens.
- After setting, the constraint position in the chip array is highlighted in yellow or light blue, and the constraint position in the package view is highlighted in orange.

The functions available on the right-click menu are as follows:

- Unplace.
- Reset port properties.
- Highlight the constraint position.
- Update the constraint according to the device layout file.
- Set the I/O type, turnover rate, pull mode, drive mode, BANK voltage, etc.
- The right-click menu supports bulk modification of the port attributes.

## Note!

Users can select multiple ports. If multiple ports have the same property values, they can be configured uniformly via the right-click menu.

## Figure 3-49 I/O Constraint View

Port	Direction	Diff Pair	Location	Bank	Exclusive	ІО Туре	Drive	Pull Mode	PCI Clamp	Vref	
E	input	dr	unp	1 laca	1	LVCMOS18	N/A	UP	N/A	N/A	
N	input	dr	'an	et Properties		LVCMOS18	N/A	UP	N/A	N/A	
K	input	dr	200	nlight		LVCMOS18	N/A	UP	N/A	N/A	
	input	dr	rag	ate Constraints l	Ising Posp	LVCMOS18	N/A	UP	N/A	N/A	
)	output	dr	rag	ate constraints c	osing Posp	LVCMOS18	8	UP	N/A	N/A	
L	output	dr	ag IO T	ype	•	LVCMOS18	8	UP	N/A	N/A	
2	output	dr	ag Pull	Mode	•	LVCMOS18	8	UP	N/A	N/A	
3	output	dr	ag Hyst	eresis	•	LVCMOS18	8	UP	N/A	N/A	
	input	dr	ag Ban	k Vccio	+	LVCMOS18	N/A	UP	N/A	N/A	

## **Primitive Constraints**

The primitive constraint view is shown in Figure 3-50. The functions as follows:

- Display the name, type, location, and exclusive information for all primitive constraints.
- Editing.

Note!

- Modify the location information by dragging or double-clicking the input.
- Select exclusive information by double-clicking.
- Right-clicking on the menu allows the user to highlight constrained positions; remove, add, and update constraints; and set the parameter value.

The grammar and acceptability of the location will be checked when the primitive constraints are manually inputted. The error message dialog box will be as shown in Figure 3-16 and Figure 3-17.

### Figure 3-50 Primitive Constraints View

Primitive	Туре	Locations	Exclusive
slu_inst	INS_ALU	R3CR(2)(R) Select Primitives Hightlight Update Constraints Using Posp Set Parameter Remove	False

## **Group Constraints**

The group constraint view is shown in Figure 3-51. The functions are as follows:

- Display the name, type, number of primitive, location, and exclusive information, including primitive and relative.
- As shown in Figure 3-15 and Figure 3-19, double-click on the corresponding group, open the dialog box and edit the constraint information.
- The right-click menu allows the user to highlight constrained positions; remove, add, and update constraints; and set the parameter value.

Group Constraints				5 ×
Group	Туре		Members Number	Members Exclusive
grp1	Primitive	2	(ref grid pumber 1) New Primitive Group	false
			New Relative Group	
			Highlight	
۲ III			Update Constraints According Place	- F
Tcl I/O Cons Primitive Cons Grou	p Cons… Res	ource Rese"	Remove	Cons… Vref Cons…

### Figure 3-51 Group Constraints View

## **Resource Reservation**

The reserved constraint view is shown in Figure 3-52. The functions are as follows:

- Display location information for all the currently reserved constraints.
- The right-click menu allows the user to highlight constrained positions, and remove, add, and update constraints.
- The name is used to distinguish between the usage constraints. It cannot be modified.

### Note!

Users can modify the location information by dragging or double-clicking on the input.

#### Figure 3-52 Resource Reservation View

eserve_0 type location GENERAL  GENERAL LUT REG WLUT	Name	Locations	Attribute	
LUT REG	eserve_0	type location	GENERAL -	
REG			GENERAL	
			LUT	
WLUT			REG	
			WLUT	

## **Clock Assignment**

The clock constraint view is shown in Figure 3-53. The functions are as

follows:

- Display information on current clock constraints.
- The right-click menu allows users to add and remove clock constraints.

#### Note!

-- - -

- Double-click to edit.
- If no location information is available, the clock constraint does not support the drag/drop function.

The create clock constraint view is shown in Figure 3-21.

### Figure 3-53 Clock Constraint View

CTOCK WZZIŚDWEJ	at					
	Net		Туре	Signal		
a_c[0]		Clock/C Remove	RUEG Control Assignment	LOGIC		
Tcl Console	I/O Constraints	Primitive Constr	aints Group Const:	raints Resource	Reservation Clock As	signment

## **Quadrant Constraints**

The quadrant constraint view is shown in Figure 3-54. The functions are as follows:

- Display all quadrant constraints, including instance name, type, and quadrant location.
- The right-click menu allows the user to add new quadrant constraints and remove the existing constraints.

### Note!

The quadrant constraints are valid only for DCS and DQCE devices.

The create quadrant constraint is shown in Figure 3-22 and Figure 3-23.

## Figure 3-54 Quadrant Constraints View

Quadrant Constr	aints						
	Instance	Туре	F	Position			
dqce_inst		INS_DQC	e RIGHT				
				Select Dcs/Dqce			
				Remove			
Tcl Console	I/O Constraints	Primitive Constraints	Group Constraints	Resource Reservation	Clock Assignment	Quadrant Constraints	Helk Constraints

## Hclk Constraints

The hclk constraints view is as shown in Figure 3-55. The functions are as follows:

- Display the location constraints for each instance for hclk, including instance name, type, and quadrant location.
- The right-click menu allows the user to add new constraints and remove the existing constraints.

The create clock constraint is shown in Figure 3-24.

### Figure 3-55 Hclk Constraints View

Helk (	Constraints						
	Instance	Туре		Position			
clko	liv_inst	INS_CLKDIV		Select Hclk			
				Remove			
Tcl	Console I/O Constraints Pr	imitive Constraints Group	Constraints F	Resource Reservation	Clock Assignment	Quadrant Constraints	Hclk Constraints

## **Vref Constraints**

The Vref constraints view is shown in Figure 3-56. The functions are as follows:

- Display Vref driver information defined by the user. The user can customize the Vref name and location.
- The right-click menu allows the user to add and remove constraint information.

### Note!

Set location information by dragging.

## Figure 3-56 Vref Constraints View

Name     Locations     IO TYPE       ref_driver_0     drag to set     VREF1_DRIVER       Highlight     Remove	Name     Locations     IO TYPE       ref_driver_0     drag to set     VREF1_DRIVER         Define Vref Driver       Highlight	f Constraints							
Define Vref Driver Highlight	Define Vref Driver Highlight	٦		Locations		IO TYPE			
Highlight	Highlight	vref_driver_0		drag to set	VREF1_DRIV				
Remove	Remove								
						Remove			
		Tcl Console	I/O Constraints	Primitive Constraints	Group Constraints	Resource Reservation	Clock Assignmen	t Quadrant Constraints	Helk Constraints

# 3.5 Tcl Console View

The Message View is shown in Figure 3-57. The output result is displayed.

Figure 3-57 Message View

lessage									8
> into	(FF0001) : run fl	porpian flow.							
> Info	(FP0002) : buildi:	ng chip graphic							
> Info	(PA0001) : "D:/ID:	E_test/CHUNXIANG/floorplan	n/fp_test/src/dqce.vm°	'   Reading netlist file.					
> Info	(PA0004) : Parsin	g netlist file 'D:/IDE_tes	st/CHUNXIANG/floorplar	/fp_test/src/dqce.vm/_co	ompleted.				
> Info	(PA0006) : Proces	sing netlist completed.							
> Info	(MEOOO2) : readin	z device GW1N-4 packaze PH	GA256						
> Info	(ME0003) : buildi:	ng chip array							
> Info	(FP0001) : run fl	oorplan flow.							
> Info	(FP0002) : building chip graphic								
> Info									E
> Info									
> Info	(PA0006) : Proces	sing netlist completed.	· · ·		•				-
Message	I/O Constraints	Primitive Constraints	Group Constraints	Resource Reservation	Clock Assignment	Quadrant Constraints	Hclk Constraints	Vref Constraints	

# 3.6 Create Constraints - Drag Mode

The constraints editor supports the creation of I/O constraints, primitive, group, resource, clock, quadrant, Hclk, and Vref. Users can create constraints on the tools. See <u>3.4.1</u> Menu for details.

## Note!

You can also create constraints in other ways. Take "drag/drop", for instance. This section describes how to generate a onstraint by dragging and dropping.

# 3.6.1 Set I/O Constraints Position

Users can set the I/O constraints position via the following process:

After initiating the FloorPlanner and reading the design file, the constraints for all ports in the Netlist are automatically loaded into the I/O constraints editor.

You can drag and drop in the following two ways:

- 1. In Netlist (on the left), select port and drag it to the chip array.
- 2. In I/O constraints editor (located below), select constraints and drag it to the chip array. Place it if permitted. The constraints location will then be changed to the IOB Location, as shown in Figure 3-58.

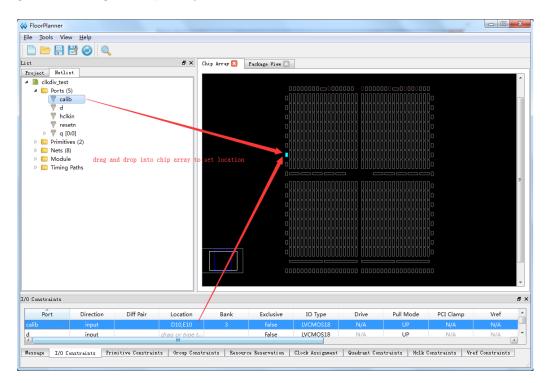


Figure 3-58 Drag to Chip Array to set I/O Constraints

Drag to package view to set the I/O constraints. The steps are as follows:

- 1. In Netlist, drag port to package view and place it in a permissible location.
- 2. Change the location of the constraints to the pin in the package view, as shown in Figure 3-59.

### Note!

You can also set I/O constraints by dragging the constraints in the I/O constraints editor to the package view.

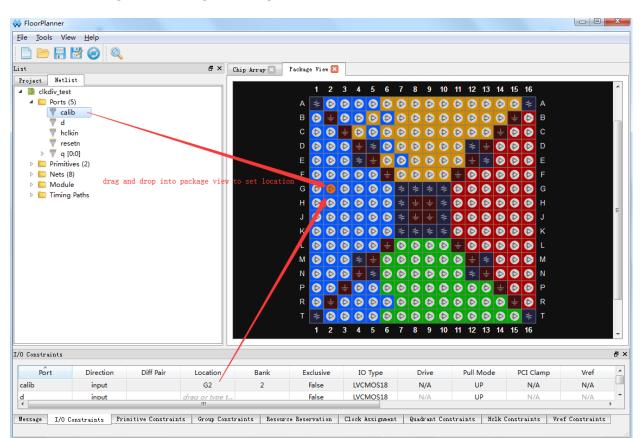


Figure 3-59 Drag to Package View and Set I/O Constraints

As shown in Figure 3-60, users can set the constraint location for the port in the constraints editor. The steps for doing this are as follows:

- 1. Select port in Netlist.
- 2. Select IOB in chip array.
- 3. Press "F3".

### Note!

You can also select IOBLOCK by zooming out on the chip array.

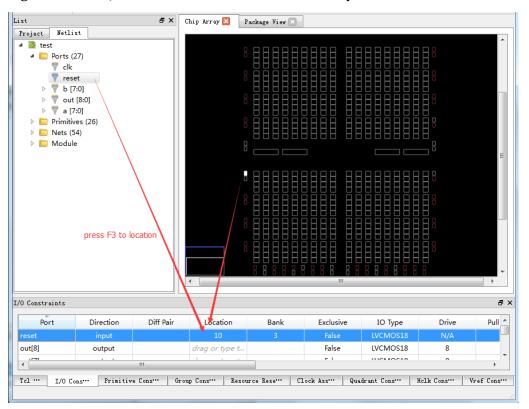


Figure 3-60 Set I/O Constraints Location Constraint by F3

In addition to the three methods described above, users can also enter the required constraint location by directly double-clicking on the I/O constraints location.

# 3.6.2 Create Primitive Constraints

Primitive constraints can be created in two ways. The steps are outlined below.

## Set primitive constraints by dragging/dropping to chip array

- 1. In Netlist, double-click or right-click on the menu. Select "Edit Constraint", generate primitive constraints, and add the corresponding primitive constraints to the primitive constraints editor.
- 2. After generating the constraints, set the constraint location, select the corresponding constraints in the primitive constraints editor, and drag it to the chip array, as shown in Figure 3-61.

## Note!

- You can also right-click the menu in the primitive constraints editor to select "Select Primitives". The Primitive Finder is displayed. Select the primitive and click "OK". Add the corresponding primitive constraints to the primitive constraints editor.
- Alternatively, select the corresponding primitive in Netlist, drag it to the chip array, and set the location of the corresponding constraints.

List & X	Chip Array 🔀	Package View 🖂		
Project Netlist				
<pre>Iterative Iterative I</pre>	to chip array to			E
Primitive Constraints		/		8×
Primitive	Туре	Locations	Exclusive	
alu_inst	INS_ALU R6C5		False	
Mes I/O Constra Primitive Constra	Group Constra	Resource Reserva*** Clock	Assign… Quadrant Constra… Helk Constra…	Vref Constra

Figure 3-61 Drag to Chip Array and Set Primitive Constraints

## Set the Primitive Constraints Location by clicking F3

- 1. Select primitive in the Netlist;
- 2. In the chip array, select the grid and press "F3" to set the primitive constraints location, as shown in Figure 3-62.

### Note!

If no primitive constraints exist, FloorPlanner creates primitive constraints and restricts them to the selected grid location.

List 🗗	Chip Array Z Package View Z
List Project Netlist Project Netlist Ports (27) Primitives (26) Out_reg_4_cZ[0] (INS_LUT4) Out_reg_4_cZ[1] (INS_LUT4) Out_reg_4_cZ[2] (INS_LUT4) Out_reg_4_cZ[3] (INS_LUT4) Out_reg_4_cZ[4] (INS_LUT4) Out_reg_4_cZ[6] (INS_LUT4) Out_reg_4_cZ[6] (INS_LUT4) Out_reg_4_cZ[7] (INS_LUT4) Out_reg_4_cZ[7] (INS_LUT4) Out_reg_4_cZ[1] (INS_DFF) Out_reg_Z[1] (INS_DFF) Out_reg_Z[2] (INS_DFF) Out_reg_Z[3] (INS_DFF) Out_reg_Z[4] (INS_DFF) Out_reg_Z[5] (INS_DFF) Out_reg_Z[6] (INS_DFF) Out_reg_Z[6] (INS_DFF) Out_reg_Z[6] (INS_DFF)	
Primitive Constraints	
Primitive	Type Locations Exclusive
out_reg_4_cZ[0]	INS_LUT4 R13C8 False
Tcl ··· I/O Cons··· Primitive Cons···	Group Cons… Resource Rese… Clock Ass… Quadrant Cons… Hclk C

Figure 3-62 Set Location of Primitive Constraints by Pressing F3

# 3.6.3 Create Group Constraints

As shown in Figure 3-63, create primitive group and relative group by right-clicking in the group constraints.

Figure 3-63 Right-Click Menu of Group Constraints Editor

Group	Туре	Members Number	Members Exclusive	Loc
	New Primitive Group New Relative Group			

## **Create Primitive Group Constraints**

- 1. Right-click on the menu and click "New Primitive Group", the Edit Primitive Group menu will open.
- 2. Input Group Name, click "+", and the Primitive Finder window will open.
- 3. Select the primitive, click "Primitive Finder" to select "OK", and add it into Members.
- 4. Input the constrained Location in Locations.
- 5. Select "OK" in the Edit Primitive Group and add Group Constraints in

Group Constraints, as shown in Figure 3-64.

- 6. Right-click "Timing Paths > Path" in Netlist.
- 7. When the menu opens, click "Edit Constraints" to add the group constraint that corresponds to path to group constraints editor.

Name out_reg_Z[4] out_reg_Z[5]	Type INS_DFF INS_DFF		Filter Name: * Type: *			••••
				Name	Type INS LUT4	
press "+" then pop u	p Primitive Finder to choose	ve	34 out_reg_4_cZ[7] 35 out_reg_Z[0]		INS_LUT4	
Locations			36 out_reg_Z[1]		INS_DFF	
RSC6, RSC8			37 out_reg_Z[2]		INS_DFF	*** ***********
input location			38 out_reg_Z[3]		INS_DFF	
			<b>39</b> out_reg_Z[4]		INS_DFF	
			40 out_reg_Z[5]		INS_DFF	
			41 out_reg_Z[6]		INS_DFF	
Dres	s "ok" then creat a group consu	äht	42 out_reg_Z[7]	press "ok" then add primitives into Members I	INS_DFF	-
	OK Cance		-	press ok chen add primitives into Members i	OK Cancel	

Figure 3-64 Create Primitive Group Constraints

## **Create Relative Group Constraints**

- 1. Right-click on the menu and click "New Relative Group." The Edit Primitive Group menu will open.
- 2. Input group name, click "+", and the Primitive Finder window will open.
- 3. Select primitives.
- 4. Select "OK" and add it to the Member.
- 5. Add a relative position for each primitive.
- 6. Select "OK" in the Edit Primitive Group, as shown in Figure 3-65.

Figure 3-65 Create Relative Group Constraints

<ul> <li>▼ reset</li> <li>▶ ▼ b [7:0]</li> <li>▶ ▼ out [8:0]</li> <li>▶ ▼ a [7:0]</li> <li>▶ ■ Primitives (26)</li> <li>▶ ► (54)</li> <li>▶ Module</li> </ul>		**	/ Primitive Finder All Primitives Filter Name: * Type: *		▼   ? <mark>▼</mark>
🗱 Edit Relative Group		? ×		Name	Туре
Group Name: grp_rel input group	p name		10 out_reg_Z[1]		INS_DFF
Hembers			11 out_reg_Z[2]		INS_DFF
	Relative Location		12 out_reg_Z[3]		INS_DFF
out_reg_Z[6] R0C1 out_reg_Z[7] R5C0 input	relative location		13 out_reg_Z[4]	INS_DFF	
			14 out_reg_Z[5]		INS_DFF
			15 oct_reg_Z[6]		INS_DFF
press "+" then pop up Primitive Finder to choose pri	mitives		16 out_reg_Z[7]		INS_DFF
			17 out_reg_Z[8]		INS_DFFR
			18 un7_out_reg_s_8		INS_LUT1 -
press "OK" then create a relative grou	up constraint	Cancel		press "ok" then add primitives into Memb	ers ist OK Cancel
Group Constraints					
Group	Туре	Mem	bers Number	Members Exclusive	Locations
grp_rel	Relative	2		N/A	N/A

# 3.6.4 Resource Reservation Creation

## **Create Resource Reservation**

- 1. Right-click on Resource Reservation and the menu will open.
- 2. Click "Reserve Resources" to add the Resource Reservation constraint to the editor, as shown in Figure 3-66.

### Figure 3-66 Create Resource Reservation

Resource Reserv	esource Reservation									
	Name	Locations	Attribute							
reserve_0			GENERAL	Reserve R						
				Highlight						
				Remove						
Tcl Console	I/O Constraints	Primitive Const	raints Groun	Constraints	Resource Reservation	Clock Assignment	Quadrant Constraints	Hclk Constraints	Vref Constraints	

## Set Resource Reservation Location

- 1. Select a constraint from the resource reservation and drag it to the target location in the chip array.
- Modify constraints location, as shown in Figure 3-67. In chip array, press "Ctrl" and select the area. Right-click "Copy Location" and copy the location to "Resource Reservation > Location" to modify. Then drag to set the resource reservation constraint location.

### Note!

Or manually input the location by double-clicking "Constraints > Location" to modify the position.

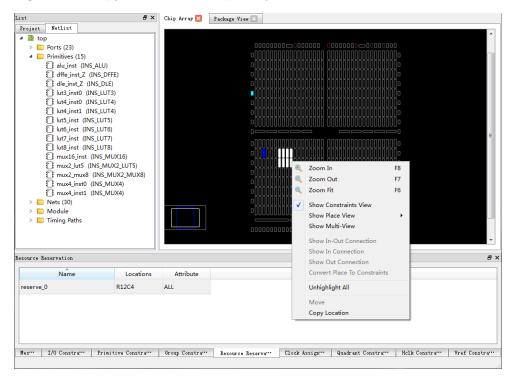


Figure 3-67 Copy Location in Chip Array to Set Resource Reservation Constraint

# 3.6.5 Create Clock Assignment

## **Create Clock Assignment Constraints**

- 1. Right-click the menu in clock assignment, select "Clock/Control Assignment". The "Clock Assignment" dialog box will open.
- 2. Click "+" and the "Find" dialog box will open. Select Net and click "OK" in Find.
- 3. Set net, select type from the drop-down list and set position and signal. Click "OK" to add the constraint to Clock Assignment, as shown in Figure 3-68.

	Find All Net Filter Name: Type:				×
		Name	Fanout	Instance Type	•
	10 b_c	[1]	2	INS_IBUF	
	11 b_c	[2]	2	INS_IBUF	=
🔅 Clock Assignment 🔹 ? 🗙	12 b_0	[3]	2	INS_IBUF	
Net clk_c	13 b_0	[4] press "+" pop up Find to select net	2	INS_IBUF	
Type BUFG -	14 b_0	[5]	2	INS_IBUF	
Signal CE	15 b_0	[6]	2	INS_IBUF	
CLK select signal	16 b_0	67	2	INS_IBUF	
LUGIC	17 clk	c	9	INS_IBUF	
SR SR	18 out	_c[0]	1	INS_DFF	-
press "ok" creat a clock Assignent OK Cancel		press "ok" add selected net into o	lock Assigment	OK	ancel

### Figure 3-68 Create Clock Assignment Constraints

## Modify Clock Assignment Constraints

In the Clock Assignment window, double-click on the name of the constraints to be modified. The clock assignment dialog box will open.

All properties except net, type, and the signal can all be modified, as shown in Figure 3-69.

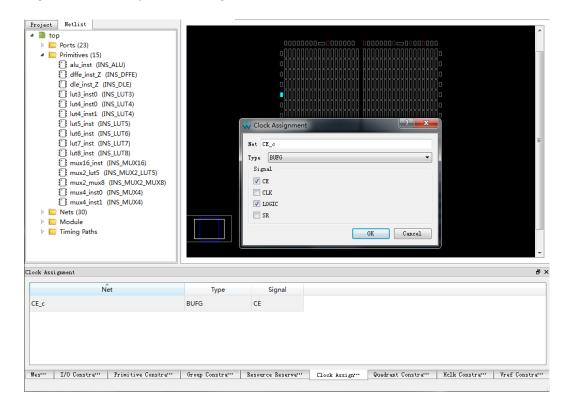


Figure 3-69 Modify Clock Assignment Constraints

# 3.6.6 Quadrant Constraints Creation

The quadrant constraints only apply to the following instance types:

- 1. Dcs
- 2. Dqce

## **Create Quadrant Constraints**

- 1. In the Quadrant Constraints editor, right-click the menu, select "Select Dcs/Dqce", and the quadrant constraints dialog box will open.
- Click "+" and the Dcs/Dqce will open. Select "Instance" and click "OK" in Dcs/Dqce to finish setting the instance. Select the constraint location in "Position". Click "OK" to add the constraint to the quadrant constraints view, as shown in Figure 3-70.

		_					
List 🗗 🛪	Chip Array 🔀	Package View 🖂					
et Netlist	]						
🔺 💁 top							
Ports (6)							
Primitives							
Nets (22)							
D Module							
					10 <sub>0</sub>		
			U	<u> </u>	JŨn		
			01	🗱 Dcs/Dqce		? <mark>×</mark>	
				Filter			
			1	Name			
			<b>-</b>	Type *		•	
			Ľ	Nam	e	Туре	
💸 Quadrant Co	nstraints		? ×	1 dqce_inst		INS_DQCE	
press "+" p	op up Dcs/Dqce dia	log to select instance					
Instance dqce_	inst	-		2 sub_inst/dqce_inst		INS_DQCE	
Position							
✓ LEFT	[	RIGHT					
set position				< m		4	
		OK C	ancel		OK	Cancel	
press "	OK" creat a Quadrar	t Constrants		press "OK" add selected instan	ce into Quadrant Cons	traints	
			-				
Quadrant Constraints							₽×
	Instance	Туре		Position			
dana taat							
dqce_inst		INS_DQCE	LEFI				

### Figure 3-70 Create Quadrant Constraints

## **Modify Quadrant Constraints**

Double click on the name of the constraint you would like to modify in the Quadrant Constraints. The Quadrant Constraints window will open. The instance cannot be changed; only the position property can be modified, as shown in Figure 3-71.

List  Xetlist  Froject Netlist  Project Op  Primitives(8)  Primitives(8)  Nets(22)  Module	Chip Array 🛛		■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■				
Quadrant Constraints							8×
Instance		Туре	Position				
dqce_inst		INS_DQCE	LEFT				
Tel C I/O Const Pr	imitive Const…	Group Const	Resource Reser	Clock Assi	Quadrant Const	Helk Const…	Vref Const…

Figure 3-71 Modify Quadrant Constraints

# 3.6.7 Create Hclk Constraints

## **Create Hclk Constraints**

The steps required to create Hclk Constraints are as follows:

- 1. In the Hclk Constraints window, right-click to open the menu and select "Select Hclk." Hclk Constraints will open.
- 2. Click "+" and Hclk will open.
- 3. Select the instance, and click "OK" in Hclk; set the instance, select the constraint position in position, and click "OK" to add the constraint to Hclk Constraints. As shown in Figure 3-72.

### Note!

Hclk constraints only constraint the Instance of Clkdiv and dlltype types.

List 🗗 🛪	Chip Array 🔀 🛛 Pack	tage View 🗵		
Project Netlist				
▲  age top ▷  in Ports (5)				
Primitives (6)				
Nets (19)				
Module				
			W Hclk	?
			Filter	
			Name	
			Type *	•
🐝 Hclk Constraints		? ×		
	ress "+" pop up Hclk to se	elect instance	Name	Туре
Instance clkdiv_inst			1 clkdiv_inst	INS_CLKDIV
Position			2 sub_inst/clkdiv_inst	INS_CLKDIV
TOPSIDE[0]	TOPSIDE[1]		2 sub_inst/clkdiv_inst	INS_CLKDIV
TOPSIDE[0]		-		INS_CLKDIV
TOPSIDE[0]	BOTTOMSIDE [		press "OK" add select	
TOPSIDE[0]	position EFTSIDE[1]     RIGHTSIDE[1]	]		ted instance into Hclk Constraints
TOPSIDE[0]	position EFTSIDE[1]     RIGHTSIDE[1]		press "OK" add select	
TOPSIDE[0]  TOPSIDE[0]  TOPSIDE[0]  LEVISIDE[0]  RIGHTSIDE[0]  press "OK" creat a Ho	position EFTSIDE[1]     RIGHTSIDE[1]	]	press "OK" add select	ted instance into Hclk Constraints
TOPSIDE[0] BOTTOMSIDE[0] LEFISIDE[0] RIGHTSIDE[0] press "OK" creat a Ho Helk Constraints	position EFTSIDE[1]     RIGHTSIDE[1]	]	press "OK" add select	ted instance into Hclk Constraints
TOFSIDE[0] BOTTOMSIDE[0] EFTSIDE[0] RIGHTSIDE[0] press "OK" creat a Ho Helk Constraints	bosition bottomside [1] bottomside [	] Cancel	Position	ted instance into Hclk Constraints

### Figure 3-72 Create Hclk Constraints

## Modify Hclk Constraints

The steps required to modify the Hclk constraints are as follows:

Double-click on the constraint name in Hclk Constraints. The "Hclk Constraints" dialog will open, as shown in Figure 3-73.

## Note!

The instance cannot be changed. Only the position property can be modified.

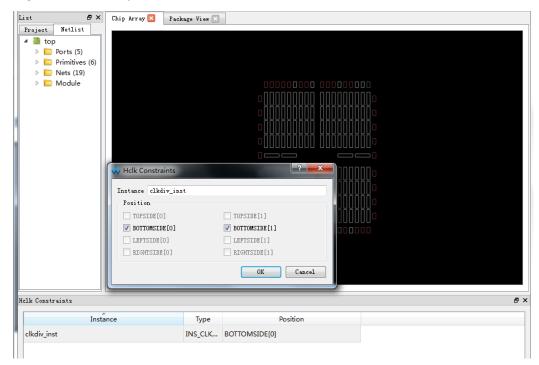


Figure 3-73 Modify Hclk Constraints

# 3.6.8 Vref Constraints Creation

## **Create Vref Constraints**

In Vref constraints, right-click the menu and select "Define Vref Driver", add the Vref Constraints into Vref Constraints Editor, as shown in Figure 3-74.

Figure 3-74 Create Vref Constraints

Vref Constraint	ts		, L					
	Name	Locations		IO TYPE				
vref_driver_0		drag to set	VREF1_DRIV	/ER				
vref_driver_1		drag to set	VREF1_DRIN	/ER				
			κ		Define Vref Drive Highlight	er		
					Remove			
Tcl Console	I/O Constraints	Primitive Constraints	Group Constraints	Resource Reservation	Clock Assignment	Quadrant Constraints	Helk Constraints	Vref Constraints

Customize the Vref constraint name. The Vref name cannot be duplicated. If the user attempts to reuse an existing Vref name, a prompt will be displayed, as shown in Figure 3-75.

## Figure 3-75 Vref Rename Check

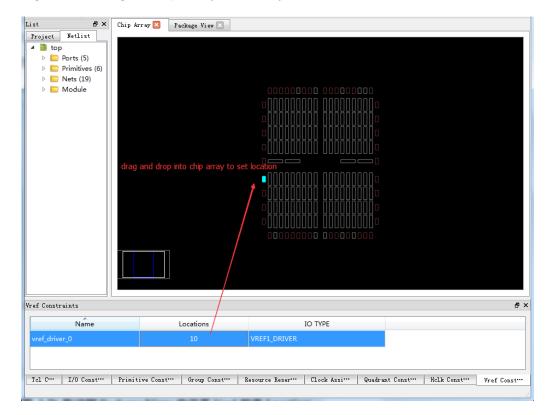


## Set Location for Vref Constraints

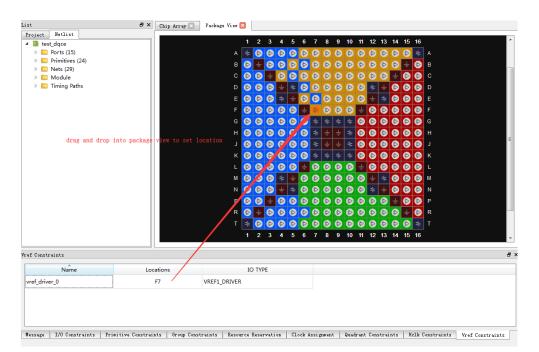
Select a constraint in Vref Constraints, drag it into the chip array, and place it in a placeable grid, as shown in Figure 3-76.

## Note!

- Users can also set the Vref constraints location in Vref Constraints;
- Select a constraint and drag it into the package view. If placeable, put it in an available place, as shown in Figure 3-77.



## Figure 3-76 Drag to Chip Array to Modify Location



#### Figure 3-77 Drag to Package View to set Vref Constraint Location

## **4**<sub>Timing Constraint</sub>

## 4.1 Static Timing Analysis (STA) Overview

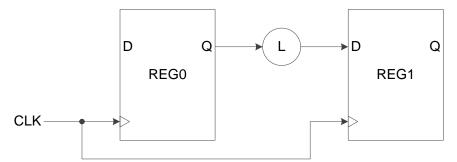
Static timing analysis (STA) comprehensively analyzes the electric network list, calculates the establishment and maintenance time of the timing elements in the circuit, and determines whether it meets the requirements. The designer provides constraint excitation, and the software completes the computational analysis process automatically. Compared with the traditional analysis method, STA offers a short verification time and high coverage. STA starts with user constraints and analyzes specific timing models. Gowin software verifies the circuit performance, identifies possible timing violations, and generates timing analysis feedback to the user by analyzing data required time, data arrival time, and clock arrival time. Users can further adjust the circuit design as required to improve system working rate and stability.

Before conducting STA, please refer to the basic models, terms, and concepts described below.

## 4.1.1 Basic Model of Static Timing Analysis

STA is used for a timing analysis model that starts from the timing element to the timing component. The basic model is shown in Figure 4-1. The REG0 trigger synchronizes data from D to Q at the clock effective edge. The data triggers REG1, which collects data from the REG0 trigger at the clock effective edge. STA is employed to verify whether REG1 can collect the data from the REG0 trigger correctly.

#### Figure 4-1 Basic Model of Timing Analysis



The effective clock of the REG0 trigger is launch edge; the REG1 trigger effective clock edge is the latch edge. If we do not take into account the effect of the path constraints, the interval time of the two edges is usually one clock cycle or half a clock cycle.

## 4.1.2 Timing Analysis Terminology

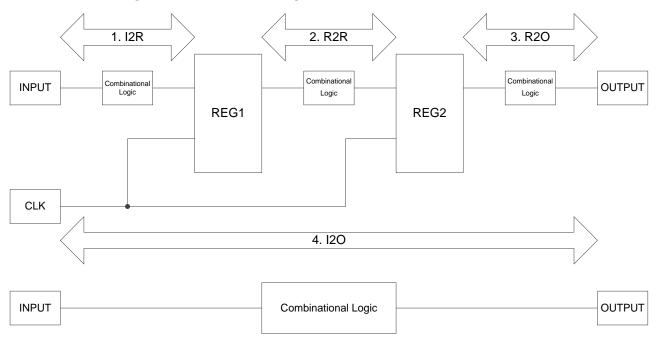
The basic timing units involved in the timing model are as follows:

- Cells: Include LUT, DFF, MUX, DL, DDR, etc;
- Pins: Cells I/O port;
- Ports: I/O ports of the top-level module, usually the external pins of the device;
- Nets: Connection between pin and pin;
- Clocks: The clock set in timing constraint.

## 4.1.3 Timing Analysis Path

Usually, static timing analysis (STA) analyzes four types of paths and classifies them according to different start and end points, as shown in Figure 4-2:

- I2R: From input port to register;
- R2R: From register to register;
- R2O: From register to output port;
- I2O: From input to output.



**Figure 4-2 STA Four Timing Paths** 

STA calculates the data arrival time and data required time through four types of paths.

The data arrival time refers to the time required from the beginning of the timing path to the end point. The data required time refers to the time required for the data to arrive. When calculating the data arrival time, the clock path has a clock skew, which refers to the time difference between the clock and the clock port.

## 4.1.4 Common Timing Checks

#### Setup Time and Hold Time Check

- 1. Setup time: The shortest time for data stability before the clock effective edge. If the time is not met, the next level register cannot collect data properly.
- 2. Hold time: The shortest time for data stability after the clock effective edge. If the time is not met, the data will be overwritten by the new data transmitted by the superior register.

#### **Recovery Time and Removal Time Check**

- 1. Recovery time: Before clock effective edge; the shortest time for signal removing asynchronous reset to keep stable. If the time is not satisfied, the register may not enter the normal working state.
- 2. Removal time: After clock effective edge; the shortest time for signal removing asynchronous reset to keep stable. If the time is not satisfied, the register may not enter the normal working state.

#### MPW Check

MPW: Min. width of high and low level recognized by internal chip components. The clock will not normally be recognized if the MPW is lower than the width.

STA usually checks the above three types of tests and recommends the layout process to better meet the user's requirements for timing.

## 4.2 Timing Constraint Editor

### 4.2.1 Overview

Gowin STA supports multiple timing commands, such as constraints for clock, I/O, path, and clock report command. The user can add timing constraints using the GUI provided by the STA.

STA supports default timing analysis. The default clock rising edge is 0 ns, falling edge is 5 ns, and the period is 10 ns.

STA provides timing analysis in the cross-clock domain for all clocks by default. If cross-clock domain analysis is not selected, the relationship between clocks can be set via specific timing constraints.

STA provides two timing reports: HTML and text. HTML is the default setting. The default content of the timing report includes establish time check, keep time check, maximum frequency calculation, and minimum clock pulse check. A custom report based on the specific requirements is also supported.

## 4.2.2 Open Editor

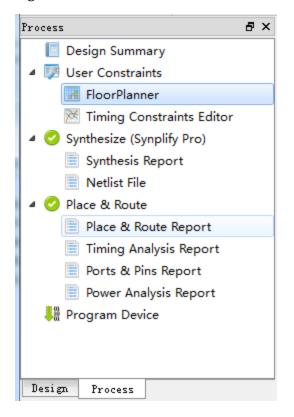
#### **Start Timing Constraints Editor**

After setting RTL software engineering, to open the timing constraints editor, select "Process > Timing Constraints Editor" from the menu, as shown in Figure 4-3.

#### Note!

The network list of the project is automatically loaded into the timing constraints editor.

**Figure 4-3 Process View** 



### 4.2.3 Create and Open the Constraint File

The usage of the timing constraints editor is introduced below with a set of examples.

#### **Create Constraint File**

The steps required to create a constraint file are as follows:

- 1. Click "File > New", the "Open File" dialogue will open.
- 2. Select "Timing Constraints File", as shown in Figure 4-4.

#### Note!

You can also open a new timing constraints file by taking the following steps:

- Click the "New" icon on the toolbar.
- Use the "Ctrl+N" shortcut.

🐳 New	? <mark>x</mark>
Rost-Synthesis Project	•
▲ Files	
📘 🔂 Verilog File	
VHDL File	E
Physical Constraints File	
Timing Constraints File	
GAO Config File	
Create a Timing Constraints file.	
ОК	Cancel

Figure 4-4 Open New Timing Constraints File

Click "OK", and a new timing constraints file will open, as shown in Figure 4-5.

Figure 4-5 Create Timing Constraint File

🐝 New Timi	ng Constraints File	? ×
Name:	Enter a name	. sdc 🔻
Create in:	C:\Users\root\Desktop\1.8IDE\test\src	.sdc .scf prowse
	🗹 Add to current project	
	_	OK Cancel

- Name: Name of the new sequential constraint file. The .sdc and .scf. suffixes are supported.
- Create in: Select the constraint file location by clicking on the "Browse" button. New constraint files are stored in the src folder of the project by default.
- Add to current project: If this is checked, the constraints file will be added to the project automatically.

#### **Open Constraint File**

The steps required to open a constraint file are as follows:

1. Click "File > Open".

## 2. Open "Open Timing Constraint", as shown in Figure 4-6. Note!

You can also open a constraints file by taking the following steps:

- Click the "Open" icon on the toolbar.
- Use the "Ctrl+O" shortcut.

#### Figure 4-6 Open Timing Constraint File

Open File	→ test → src			<ul> <li>- 4y 搜索 51</li> </ul>	rc	×
组织 ▼ 新建文件夹					= •	0
☆ 收藏夹	名称	修改日期	类型	大小		
▶ 下载	itest.cst	2017/6/30 17:38	CST 文件	1 KB		
	test.gao	2017/6/22 16:49	GAO 文件	0 KB		
3 最近访问的位置	test.gpa	2017/6/22 16:29	GPA 文件	0 KB		
	test.sdc	2017/6/29 9:42	SDC 文件	0 KB		
言 库	test.v	2017/6/20 13:39	V 文件	1 KB		
■ 単 目 目 1 目 目 目 目 目 目 目 目 目 目 目 目 目 目 目 目	test1.cst	2017/6/22 16:40	CST 文件	0 KB		
■ 视频						
👌 音乐						
🖳 计算机 📃						
🏭 本地磁盘 (C:)						
📻 本地磁盘 (D:)						
🕞 本地磁盘 (E:) 🔻						_
文件	名(N):			✓ All Files	(*) (*.*)	-
				All Files		
					Design Constraints	
					esign Constraints File FPGA Designer Proj	
Start	Page		Design	Sunnary GOWIN	Firm Core File (*.vfc)	
					Oscilloscope Config Physical Constraint I	
					Power Analyzer Con	
				GOWIN	Timing Constraint Fi	le (*.sdc *.scf)
					ocument File (*.htm / Initialization File (*.ı	
					/ Initialization File (".i xt File (*.txt)	TII)
					le (*.vhd *.vhdl)	
					HDL File (*.v *.sv *.vn	

Select the location in which the timing constraint file is stored, and select the file you wish to open.

## 4.2.4 Editor

After creating or opening the constraint file, the editor interface will be displayed, as shown in Figure 4-7.

Eile <u>C</u> onstraints <u>R</u> eports <u>V</u> iew <u>H</u> elp								
	<ul> <li>Timing Constraints</li> <li>Clocks</li> <li>Clock Latency</li> <li>Clock Uncertainty</li> <li>Clock Group</li> <li>I/O Delay</li> <li>Path</li> <li>False Path</li> <li>Max/Min Delay</li> <li>Multicycle Path</li> <li>Report</li> <li>Report Timing</li> </ul>	11	Clock Name cik1	Type Base	Period 20ns	Frequency 50MHz	Rise 0	Fall 10
nsole	Report High Fanout Nets Report Route Congestion	-	•					
reate_clock -name clk1 -period 20 -v	weform {0 10} [get_ports {din1[1]	}]						

Figure 4-7 Timing Constraint Editor

The Netlist Tree view is as shown in Figure 4-8.

**Figure 4-8 Netlist Tree View** 

Netlist Tree	₽×
۱	Q
▲ (test)	
I/O Ports (27)	
Nets (81)	
🖻 🚞 Primitives (56)	

The Netlist Tree window contains various elements of the current network table file, including top module, I/O ports, nets, and primitives.

- "<sup>"</sup> : check flatten.
- "<sup>1</sup> : check hierarchy.

The middle and right area of the main interface is the constraint editing area, as shown in Figure 4-9. The left list is the timing constraint list, and the right side is the editing list. Click on a constraint type from the type list, and the constraint editing list will be displayed in the edit area list.

Timing Constraints	Clock Name	Туре	Period	Frequency	Rise	Fall	Divide by	Multip
Clocks	clk1	Base	10ns	100MHz	0	5	N/A	N,
Clock Latency								
Clock Uncertainty								
Clock Group								
I/O Delay								
▲ Path								
False Path								
Max/Min Delay								
Multicycle Path								
<ul> <li>Report</li> </ul>								
Report Timing								
Report High Fanout Nets								
Report Route Congestion								
Report Min Pulse Width								
Report Max Frequency								
Report Exception								
Set Operating Conditions								
	•							

#### Figure 4-9 Constraint Edit View

## 4.2.5 Timing Constraint

There are two GUI interfaces for timing constraints:

1. Click "Constraints" in the menu. Select the timing constraint command and open the GUI by selecting the corresponding constraint command, as shown in Figure 4-10.

Figure 4-10 Open Timing Constraint View in Menu

Cor	nstraints <u>R</u> eports <u>V</u> iew								
	Create Clock								
	Create Generated Clock								
	Set Clock Latency								
	Set Clock Uncertainty								
	Set Clock Groups								
	Set I/O Delay								
	Set False Path								
	Set Max/Min Delay								
	Set Multicycle Path								
	Set Operating Conditions								

2. Right-click the table on the right side of the STA to select different timing constraint commands based on the different options in the middle, as shown in Figure 4-11.

Timing Constraints	Clock Name	Туре	Period	Frequency	Rise	Fall	Divide by	Multiply by
<ul> <li>Clocks</li> </ul>	clk1	Base	10ns	100MHz	0	5	N/A	N/A
Clock Latency		Remove	1 10/13	10010112		, in the second s		17/6
Clock Uncertainty								
Clock Group		Set Clock Laten	·					
I/O Delay		Set Clock Unce	rtainty					
✓ Path		Set I/O Delay						
False Path		Set Clock Grou	ps					
Max/Min Delay		Create Clock						
Multicycle Path		Create Generat	ed Clock					
Report		create Generat	eu clock					
Report Timing								
Report High Fanout Nets								
Report Route Congestion								
Report Min Pulse Width								
Report Max Frequency								
Report Exception								
Set Operating Conditions								
	•							

Figure 4-11 Right Click to Open the Timing Constraint View

## 4.3 Edit the SDC File

You can open, read, and manually modify the SDC file in the timing constraints editor, as shown in Figure 4-12.

Figure 4-12 Edit SDC File

🐳 GOWIN FPGA Designer - [D:\user-bak\Users\gowin\Desktop\1.8IDE\test\src\test.sdc]	- • ×
Die Edit Project Jools Window Help	_ & ×
🗋 🖮 🖩 🖷 🖛 🔺 🧏 🖻 🖷 👪 🥌	
Design <b>B</b> X 1 create_clock -name clk1 -period 10 -waveform (0 5) [get_ports (cl.	k}]
▲ 🧰 test - [D:\user=bak\Users\gowin\Desktop\1.8IDE\t) 2	
😡 GW1N-4-PBGA256-6	
4 📂 Verilog Files	
test.v	
4 📄 Timing Constraints	
i test.sdc	
<	۱.
Design Process 🤤 Start Page 🗵 📘 Design Summary 🔝 📡 test.sdc 🔯	
Output	8 ×
Output Error Warning Info	
	In: 1 Col: 0

## 4.4 Edit Timing Constraint

## 4.4.1 Clock Constraint

#### **Create Clock**

- Typically used to create the system master clock. The clock name, period, waveform, duty cycle, and clock source can be specified;
- STA provides a default clock. The period of the default clock is 10 ns, which takes up 50% of air ratio, and the rising edge reaches;
- STA can create multiple clocks, which form multiple clock domains and support cross-clock domain analysis.

Users can add the clock constraint in two ways. The operations are as follows:

- 1. Add the clock constraint through the "Constraints":
  - a) In "Constraints", select "Create Clock...", and the "Create Clock" dialog box will open, as shown in Figure 4-13;

#### Figure 4-13 Create Clock Constraint

Clock name: - Waveform	clki								
Period:	20	ns				1			
Frequency:	50	MHz							
Rising:	5	ns					_		
Falling:	15	ns	0	5		15	20		
Objects: [get_ports {clk}]									

b) Complete the Clock information, including the clock name, waveform, and objects.

#### Note!

- The figure on the right of the waveform is the clock waveform based on clock information;
- Click " ......" on the right of Objects, and the "Select Objects" dialogue will open, as shown in Figure 4-14;

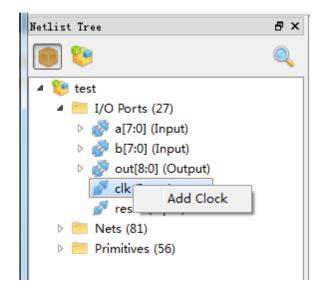
- When the target object has a clock, select "Add" and add the clock to the target object, or the new clock will be ignored;
- The default Waveform period is 10 ns, 0 ns is the uplift edge, and the half cycle is the descending edge.

#### Figure 4-14 Objects

🐝 Select Objects	? ×
Collection: get_ports V Filter: *	Search
Matches & Selected	
27 matches found 1 selected	d names
clk ^ clk	
reset	
a[7]	
a[6]	
a[5] =	
a[4]	
a[3]	
a[1]	
a[0] <<	
b[7]	
b[6]	
b[5] b[4]	
b[3]	
b[2]	
	OK Cancel

- The operations and functions represented in Figure 4-14 are as follows:
  - "Collection" specifies the searched object type.
  - "Filter" is filtered by wildcard.
  - Select "Objects" after searching, and the selected list will be displayed on the right.
  - ">" Add the selected items from the left list to the right list.
  - "> >" Add all items from the left list to the right list.
  - "<" Remove the selected item from the right list.
  - "< <" Remove all items from the right list."
  - Click "OK" to add objects.
- 2. Users can add clock constraints via the Netlist Tree:
  - a) Select I/O Port or Net in the Netlist Tree;
  - b) Right-click and select "Add Clock" to create a new clock, as shown in Figure 4-15.

Figure 4-15 Create Clock Constraint



3. After creating the clock, add the corresponding constraint to the clock, as shown in Figure 4-16.

Figure 4-16 Clock Constraint List

Clock Name	Туре	Period	Frequency	Rise	Fall	Divide by	Multiply by	Duty cycle	Phase	Offset
clk1	Base	10ns	100MHz	0	5	N/A	N/A	N/A	N/A	N/A
clk2	Base		50MHz							
								'		

The related operations are as follows:

- 1. To edit the clock, double-click on the corresponding constraint in "Clocks", open the clock and edit the clock information.
- 2. To remove a clock, select the clock you want to delete from the available list, right-click and select "Remove".
- 3. To select a clock, right-click on the menu, and quickly set the clock latency, clock uncertainty, or I/O delay information for the clock, as shown in Figure 4-17.

Clock Name	Туре	Period	Frequency	Rise	Fall	Divide by	Multiply by	Duty cyc
clk1	Base	10ns	100MHz	0	5	N/A	N/A	N/A
clk2 Base		20pc 50MHz Remove		0	10	N/A	N/A	N/A
		Set Clock L Set Clock U Set I/O Del Set Clock G	Jncertainty ay					
		Create Cloo Create Gen	ck erated Clock					

#### Figure 4-17 Right-click Contents of Clock List

#### **Generated** Clock

- Create a derived clock based on the master clock, and the master clock and derived clock will be in the same clock domain.
- The constraint conducts frequency division, frequency multiplication, phase shifting, and adjusting duty cycle based on the master clock to create a derived clock.

Users can create a generated clock in the following two ways:

- 1. Create in "Constraints"
  - a) In "Constraints", select "Create Generated Clock" and "Create Generated Clock" will open, as shown in Figure 4-18.
  - b) Select "Source", add the clock associated with the source to "Master Clock", and select the master clock.
  - c) In Objects, click the "...." icon that appears on the right, and the "Select Objects" window will open. Select the target object.

#### Note!

- If no clock exists in the selected source, the master clock has no options, and you will need to select the source again.
- In waveform, the graphic on the left displays the master clock waveform figure, the graphic on the right displays the generated clock waveform figure based on the generated options (such as double frequency, frequency division). The rising edge, falling edge, and cycle information for the Clock are shown in the figure.

🗱 Create Generated Clock	? ×
Clock Name:	
Source:	
Master Clock:	•
Relationship to source	
Based on frequency	
Divide by:	Phase:
Multiply by:	Offset:
Duty cycle:	
Based on waveform	
Edge list:	
Edge shift list:	ns ns ns
Invert waveform	Add Add
Objects:	
Source clock	Generated clock
	OK Cancel

Figure 4-18 Create Generated Clock Constraints

- 2. Create generated clock from clocks:
  - a) In Clocks, right-click the menu in the blank cell.
  - b) Select "Create Generated Clock" to create a new Generated Clock, as shown in Figure 4-19.

#### Figure 4-19 Right-click Contents of Clock List

Clock Name	Туре	Period	Frequency	Rise	Fall	Divide by	Multiply by	Duty cycle
clk1	Base	10ns	100MHz	0	5	N/A	N/A	N/A
clk2	Base	20ns	50MHz	0	10	N/A	N/A	N/A
	Dase	Create	Clock		10	17/2	IN/A	1976
		Create	Generated Clock	c				

Add new constraints in the table edit area.

The related operations are as follows:

- 1. To edit a generated clock, double-click on the corresponding constraint in "Clocks", open the clock and edit the generated clock.
- 2. To remove a generated clock, select the clock you want to delete from the available list, right-click and select "Remove".

#### **Clock Latency**

- Constraints
  - a) Set delay before clock signal reaching the access point. You can set the max./min. delay respectively from the rising along/falling edge to the access point by refining the parameters.
  - b) Clock latency is divided into two types: Network latency and source latency.
    - Network latency is the internal clock path delay;
    - Source latency is the external clock path delay;
    - The STA tool will automatically calculate the clock network latency; as such, users only need to specify the source latency.
- Interface Operation

Users can create clock latency in the following two ways:

a) To create the clock latency constraint in constraints:

From the "Constraints" menu, select "Set Clock Latency" and the "Set Clock Latency" dialog box will open, as shown in Figure 4-20.

Figure 4-20 Create Clock Latency

🐝 Set Clock La	atency		? <mark>x</mark>
Latency typ	e		
🔘 Early	🔘 Late	🖲 Both	
🔘 Rise	🔘 Fall	Soth	
Delay value:		ns	
Objects:			
Clocks:			
			OK Cancel

Input the latency information and click "OK" to save the constraint.

b) To create clock latency in Clocks

Select a clock in Clocks, right-click and select "Set the Latency" to set the latency for the clock.

#### Note!

Create Objects in Latency in this way, Objects in Figure 4-20 will be automatically specified as the clock.

#### **Clock Uncertainty**

- Constraints
  - a) Set clock uncertainty or offset to analyze clock transmission.
  - b) Set uncertainty for setup and hold separately, or set uncertainty for the transmission of the clock rising edge and falling edge.
  - c) It is possible to inform the STA of clock jitter, pessimistic, etc. via the constraint to influence timing calculation.
- Interface Operation

Create clock uncertainty by following the steps outlined below:

a) In "Constraints", select "Set Clock Uncertainty" and the Set Clock Uncertainty window will open, as shown in Figure 4-21.

Figure 4-21 Create Clock Uncertainty

🐳 Set Clock Uncertainty	? ×	
From clock: •		
To clock: 🔻	•	
Uncertainty:	Analysis type ns 💿 Setup 🔘 Hold	
	OK Cancel	

- b) Select the "From clock" type (From clock, Rise From, Fall from) and "To clock" type (To clock, Rise to, Fall to) on the left. Select the clock from all the created clock options listed on the right.
- c) After completing the required information, click "OK" to save the constraint and add the clock uncertainty.

#### Clock Group

- Specify the relationship between different clocks.
- STA provides the relationship between the group members by default. There is no correlation between groups.

Create a clock group by following the steps outlined below:

 In the "Constraints" menu, select "Set Clock Groups". The "Set Clock Groups" dialog box will open, as shown in Figure 4-22.

🐝 Set Clo	ock Groups			? ×
Group:	clk_gp			
Group:				
				Add
Second Excl	lusive	🔘 Asynchronous		
			ОК	Cancel

2. Click on the "...." icon to select the Clock for Group. As shown in Figure 4-23:

Figure 4-23 Clock Group Member List

🐝 Set Clock Groups	8 ×
Group: clk_gp	
Group:	
Group:	🐹
	Add
Exclusive	
	OK Cancel

- 3. Click " $\mathbf{X}$ " to remove the added group.
- 4. Click "OK" to save the constraint.

#### Note!

Click "Add" to add a row if you would like to set multiple clock groups.

## 4.4.2 I/O Delay

#### set\_input\_delay

• Specify the time that the data arrives at the input PORT (PORT),

specify the clock associated with input (PORT) by "-clock", which must be the clock in design.

 Input delay can be related to the rising edge (default) or falling edge (specified by "-clock\_fall").

#### Note!

- Input delay includes external clock delay, add external clock delay when calculating arrival delay in default, when "- source\_latency\_included" specified parameters, exclude external clock delay when calculating arrival delay if specify "-source\_latency\_included".
- Add constraints for PORT based on same clock, if "- clock\_fall" is specified in PORT 1, "- clock\_fall" parameter unspecified in PORT 2, PORT 2 will cover PORT 1, unless specify - add\_delay parameters.
- Input delay type "tIn" in timing report generated by STA.

#### set\_output\_delay

Specify delay output for data by PORT and the reference clock of output delay. Output delay is associated with clock rising edge in default. Specify output delay is associated with falling edge by using "- clock fall".

#### Note!

- By default, the external clock delay is not included in output delay. When using the "-source\_latency\_included" parameter, the external clock delay is included in the output delay.
- By default, the constraint covers the constraints that are added to the same PORT and has the same clock, different clock reference edges, avoid coverage via using the "-add\_delay" parameter.
- In STA timing report, the output delay type is "tOut".

Create I/O delay constraints by following the steps outlined below:

1. In "Constraints", select "Set I/O Delay" and "Set I/O Delay" will open, as shown in Figure 4-24.

Figure 4-24 Create I/O Delay Constraints

🐝 Set I/O Delay			? ×
Clock name: cl Options	lk		<b>•</b>
Input dei	lay 🔘 Output dela	y	
🔘 Minimum	🔘 Maximum 🧿 Bot)	h	
🔘 Rise 🔘	Fall 🖲 Both		
Add delay Source lat	ency included	🔲 Use falling clock d	edge
Delay value:			ns
Objects:			
		OK	Cancel

- 2. I/O delay has input delay and output delay types. Select the delay type first.
- 3. After completing the delay information, click "OK" to save the constraint.

## 4.4.3 Path Constraint

#### False Path

In timing analysis, the timing analysis path is not required by the constraint.

#### Note!

All default clock paths are related STA provided and support cross-clock domain processing by default.

The steps required to create a false oath constraint are outlined below.

1. Select "Constraints > Set False Path", then "Set False Path", as shown in Figure 4-25.

🐝 Set False Path		? <mark>x</mark>
From:		
Through:		
To:		
Analysis type:	💿 Setup 💿 Hold 💿 Both	
	ОК	Cancel

2. Select the "From" type (From, Rise From, Fall From) and "To" type (To, Rise To, Fall To) on the left, and click "...." on the right to select the object. Click "OK" to save the constraint.

#### Max/Min Delay

The steps required to create a Max/Min Delay constraint are outlined below.

1. Select "Constraints > Set Max/Min Delay", then "Set Max/Min Delay", as shown in Figure 4-26.

#### Figure 4-26 Create Max/Min Delay Constraint

🐳 Set Max/Min D	elay	8 ×
Delay type Max	🔘 Min	
From:		
Through:		
To:		
Delay value:		ns
		OK Cancel

 Delay type selects delay type (max or min), From and To type are bound to set false path. After completing the delay information, click "OK".

#### MultiCycle Path

By default, the STA performs a single-cycle clock analysis. The setup time check is the effective clock edge of the next clock cycle on the edge of source clock. However, this approach does not apply to certain timing paths. Logic design circuit analysis is the most typical example. More than one set of clock cycle time data will be required to stabilize the xxx if a logic circuit calculates a long or complex path. STA provides multicycle paths to set command set\_multicycle\_path, which allows users to set the N clock cycle of the path check. The path parameter defines the total clock cycles (total time period of the transmission path from signal original point to signal transmission). The signal propagation path is more than one clock cycle.

More cycle paths set command set\_multicycle\_path, which can be used for regulating setup time analysis and hold time analysis, rising edge/falling edge analysis, launch clock/latch clock analysis, etc. For multi-cycle paths, see the command parameters for details.

An exception is to loosen the relative relation between launch clock and latch clock, including the non-critical path and the associated clock path.

#### Note!

Setting multiple cycle path commands will have an impact on the setup time and hold time. If the -setup or –hold options are not specified, STA is -setup by default. If the -setup value is set, the hold value will not be affected. STA provides the ability to automatically restore hold by default. If the user specifies a hold value, STA will prioritize user settings.

The steps required to create a multicycle path constraint are outlined below.

1. Select "Constraints > Set Multicycle Path", and the "Set Multicycle Path" window will be displayed, as shown in Figure 4-27.

Figure 4-27	Create	Multicy	ycle I	Path	Constrain	t
0			/			

🐳 Set Multicycle Path	8 ×
From:	
Through:	
To: •	
Analysis type	Reference clock
🖲 Setup 💿 Hold	🔘 Start(launch clock) 🔘 End(latch clock)
Value:	
	OK Cancel

2. Complete the relevant information in the dialog box, and click "OK" to save the constraint.

## 4.4.4 Operation Conditions Constraints

Set the constraints for the temperature level, speed level, and process angle of the FPGA chip.

The steps required to create an operating conditions constraint are outlined below.

Select "Constraints > Set Operating Conditions". The "Set Operating Conditions" window will be displayed, as shown in Figure 4-28.

Figure 4-28 Create Operating Conditions Constraints

🐝 Set Oper	ating Conditions			S X
Grade: Model:	<ul><li>Commercial</li><li>Slow</li></ul>	Industrial Fast		
🕅 Hold	🔲 Setup	Max 🗌	🥅 Min	Max-Min
Speed: 5	×			DK Cancel

## 4.4.5 Timing Report

The operations that are incorporated in the constraint editor interface are as follows:

#### **Report Timing**

Report Content

Timing report, the file output report content according to timing report parameters.

• Interface Operation

The interface operation steps are as follows:

- 1. In the main interface, select "Timing Constraints Report Timing".
- 2. Right-click on the blank area of the right column and the "Create Report" window will appear, as shown in Figure 4-29.
- 3. Select "Create Report" and the view will open, as shown in Figure 4-30.
- 4. Fill in the relevant information in the dialog box, and click "OK" to save the timing report settings.

#### **Figure 4-29 Report Timing Creation Interface**

Timing Constraints	Analysis type	From Clock	To Clock	From	Through
Clocks					
Clock Latency					
Clock Uncertainty					
Clock Group					
I/O Delay					
▲ Path					
False Path					
Max/Min Delay					
Multicycle Path					
<ul> <li>Report</li> </ul>					
Report Timing					
Report High Fanout Nets					
Report Route Congestion					
Report Min Pulse Width					
Report Max Frequency					
Report Exception					
Set Operating Conditions					
	•	111			

Report Timing					? ×
Clocks From clock: V					•
Objects From:  Through: To:  Analysis Type					
Setup	🔘 Hold	🔘 Rec	overy	🔘 Removal	
Path Max Paths: Max Common Paths:		Min Logic Level: Max Logic Level:			
Module Instance:					
				OK	Cancel

#### **Figure 4-30 Report Timing Interface**

#### **Report High Fanout Nets**

Report Content

Report the Net numbers.

- Interface Operation
  - a) In the main interface, select "Timing Constraints > Report High Fanout Nets".
  - b) Right-click on the blank area in the right column and the "Create Report" window will open, as shown in Figure 4-31.
  - c) Select "Create Report" and the view will open, as shown in Figure 4-32.
  - d) Complete the relevant information in the dialog box, and click "OK" to save the timing report settings.

Timing Constraints	Max Net Number Max Fanout Number	Min Fanout Number	Report Clock Net	Rep
Clocks				
Clock Latency				
Clock Uncertainty				
Clock Group				
I/O Delay				
▲ Path				
False Path				
Max/Min Delay				
Multicycle Path				
Report				
Report Timing				
Report High Fanout Nets		Create Report		
Report Route Congestion				
Report Min Pulse Width				
Report Max Frequency				
Report Exception				
Set Operating Conditions				
				P.

Figure 4-31 Report High Fanout Nets Create Interface

Figure 4-32 Report High Fanout Nets Interface

🐝 Report Fano	ut Nets				_	? <mark>X</mark>
Max Net:	10					
Min Fanout:						
Max Fanout:						
🔲 Report Clo	rk Net	🔲 Report	Set/Res	set Net	Asce	ending
			l	OK		ancel

#### **Report Route Congestion**

Report Content

Report the congestion degree.

Interface Operation

The operation steps are as follows:

- 1. In the main interface, select "Timing Constraints > Report Route Congestion".
- 2. Right-click on the blank area in the right column. The "Create Report" window will open, as shown in Figure 4-33.
- 3. Select "Create Report" and the view will open, as shown in Figure 4-34.
- 4. Complete the relevant information in the dialog box, and click "OK" to save the timing report settings.

#### Figure 4-33 Report Route Congestion Create

Timing Constraints	Max Grid Number	Min Route Congestion	Max Route Congestion	Location	
Clocks			_		
Clock Latency					
Clock Uncertainty					
Clock Group					
I/O Delay					
▲ Path					
False Path					
Max/Min Delay					
Multicycle Path		Create	e Report		
▲ Report					
Report Timing					
Report High Fanout Nets					
Report Route Congestion					
Report Min Pulse Width					
Report Max Frequency					
Report Exception					
Set Operating Conditions					

Figure 4-34 Report Route Congestion Interface

🐝 Report Route Congesti	on 🦉	X
Max Grid Number:	10	
Min Route Congestion:		(0-1)
Max Route Congestion:		(0-1)
Grid Location:		
	OK Ca	ncel

#### **Report Min Pulse Width**

Report Content

Report the minimum pulse width.

- Interface Operation
  - a) In the main interface, select "Timing Constraints > Report Min Pulse Width".
  - b) Right-click on the blank area in the column on the right and the "Create Report" window will open, as shown in Figure 4-36.
  - c) Select "Create Report" and the view will open, as shown in Figure 4-35.
  - d) Complete the relevant information in the dialog box, and click "OK" to save the timing report settings.

#### Figure 4-35 Report Min Pulse Width Interface

Timing Constraints	Max Path Number Min Pulse Number De	etail
Clocks		
Clock Latency		
Clock Uncertainty		
Clock Group		
I/O Delay		
<ul> <li>Path</li> </ul>		
False Path		
Max/Min Delay		
Multicycle Path		
Report	Create Report	
Report Timing		
Report High Fanout Nets		
Report Route Congestion		
Report Min Pulse Width		
Report Max Frequency		
Report Exception		
Set Operating Conditions		
	< III	4

#### Figure 4-36 Report Min Pulse Width Interface

🐳 Report Min Pulse Wid	th		? ×
Max Clock Path:	10		
Minimum Pulse Width:			
Maximum Pulse Width:			
🔲 Detail			
Objects:			
		OK	Cancel

#### **Report Max Frequency**

Report Content

Report the maximum frequency.

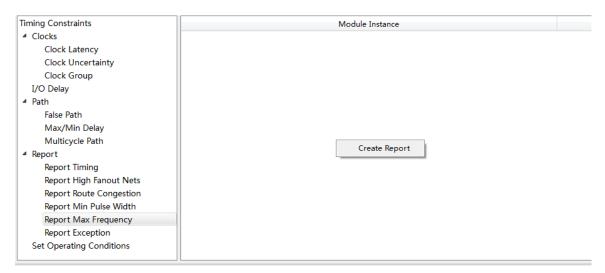
Interface Operation

The operation steps are as follows:

- a) In the main interface, select "Timing Constraints > Report > Report Max Frequency".
- b) Right-click on the blank area of the right column and the "Create Report" window will open, as shown in Figure 4-37.
- c) Select "Create Report", and the view will open, as shown in Figure 4-38.
- d) Complete the relevant information in the dialog box, and click "OK"

to save the timing report settings.

**Figure 4-37 Report Exception Interface** 



#### Figure 4-38 Report Max Frequency Interface

🐝 Report Max Freq	uency	? <mark>x</mark>
Module Instance:		
		Cancel

#### **Report Exception**

Report Content

Report Exception.

• Interface Operation

The operation steps are as follows:

- 1. In the main interface, select "Timing Constraints > Report > Report Exception".
- 2. Right-click on the blank area of the right column and the "Create Report" window will appear, as shown in Figure 4-39.
- 3. Select "Create Report", and the view will open, as shown in Figure 4-40.
- 4. Complete the relevant information in the dialog box, and click "OK" to save the timing report settings.

#### **Figure 4-39 Report Exception Interface**

Timing Constraints	Analysis type	From Clock	To Clock	From	Through
Clocks					
Clock Latency					
Clock Uncertainty					
Clock Group					
I/O Delay					
Path					
False Path					
Max/Min Delay					
Multicycle Path					
▲ Report					
Report Timing			Create Report	t	
Report High Fanout Nets					
Report Route Congestion					
Report Min Pulse Width					
Report Max Frequency					
Report Exception					
Set Operating Conditions					
	•				

#### **Figure 4-40 Report Exception Interface**

Report Exception			२ <mark>×</mark>
Clocks			
From clock: 💌			<b>•</b>
To clock: 🔻			•
Objects			
From: 🔹			
Through:			
To: •			
Analysis Type			
Setup	© Hold	Recovery	🔘 Removal
Path			
Max Paths:		Min Logic Level:	
Max Common Paths:		Max Logic Level:	
			OK Cancel

After performing timing analysis for the whole project, the software will automatically generate the timing report. To view this report, switch to the process interface and double-click "Timing Analysis Report", as shown in Figure 4-41. Click the required command in the middle of the view, and the report will appear on the right.

File Edit Project Tools Winde	ow Help		
🗋 🗁 🗄 🖷 🖶 🖛 🔺 🗅	k 🗅 🗈 👪 🗲 🚷		
rocess &×	•		
📗 Design Summary	Timing Messa		
🛛 厚 User Constraints	Timing Summ		Timing Mossages
📕 FloorPlanner	STA Tool Ru		Timing Messages
🔀 Timing Constraints Editor	E	Report Title	Gowin Timing Analysis Report
Synthesize (Synplify Pro)	Clock Summ		
Synthesis Report	Max Freque	Tool Version	v1.8.1Beta
Netlist File	Total Negati	Series, Device, Package, Speed, Operating Conditions	GW1N, GW1N-4, PBGA256, 6, COMMERCIAL
🥝 Place & Route	Timing Details	Design Name	test
Place & Route Report	Path Slacks	Design File	E:\workspace\test\testCase\impl\synthesize\rev 1\testCase.vm
📃 Timing Analysis Report		-	
📄 Ports & Pins Report	Setup Pat	Timing Constraint File	E:\workspace\test\testCase\src\test.sdc
Power Analysis Report	Hold Path	Timing Report File	E:\workspace\test\testCase\impl\pnr\testCase.tr.html
🕌 Program Device	Recovery	Created Time	Wed Jul 04 15:30:13 2018
	Removal	Command Line	D:\1.8\Pnr\bin\gowin.exe -do E:\workspace\test\testCase\impl\pnr\cmd.do
	Minimum Pu 🗸	Legal Announcement	Copyright (C)2014-2018 Gowin Semiconductor Corporation. All rights reserved.
	• III •		
Design Process	💡 Start Page 🖂 📘	Design Summary 🖂 🛛 🍃 test.	v 🙁 📝 test. sdc 🗵 🃝 testCase. tr. html 🛛
put			
nfo (TA0001) : Timing and	alysis completed.		
	generation in progress.		
nfo (FS0002) : Bitstream nfo (PW0001) : Power ana	generation completed.		
	'testCase.power.html' fil	le completed	
	'testCase.tr.html' file (		
	'testCase.rpt.html' file		
nfo (CM0008) : Generate	'testCase.rpt.txt' file (	completed.	

Figure 4-41 View Timing Reporting Interface

## 4.4.6 Save and Export

After editing all constraints, click "File > Save" or "File > Save As". Save the constraint information in the current editor to a temporal constraint File (.sdc). see <u>Appendix B</u> for the timing constraints file format.

## 4.5 Priority of Timing Constraints

The STA covers multiple types of timing constraints. The following list is ordered from highest priority to lowest.

- 1. create\_clock and create\_generated\_clock
- 2. set\_multicycle\_path
- 3. set\_max\_delay and set\_min\_delay
- 4. set\_false\_path
- 5. set\_clock\_groups

#### Note!

STA only sorts the timing constraints that generate competition on the same timing path. Constraints that are not listed do not create competition between different types of constraints.

# **5**Timing Analysis View

The FloorPlanner tool provides timing optimization, helping users to realize timing closure by modifying physical location constraints and the key path, etc.

Please refer to the following steps to optimize timing using the FloorPlanner:

- 1. Create a new project.
- 2. Double click on the "Synthesize" option to generate a netlist file with an .vm suffix after the synthesis.
- 3. Add the physical constraints file and the timing constraints file. The physical constraints and timing constraints are not imperative; however, they are recommended for better project implementation.
- Run "Place & Route" for placement and routing, and generate the data stream file concurrently. Note!

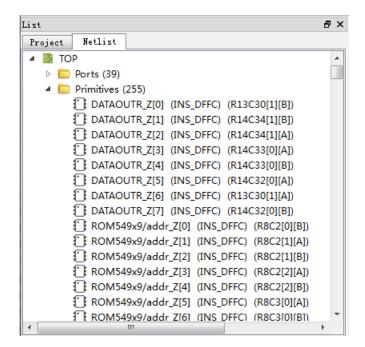
Before placing and routing, the "Place & Route" dialog box needs to be set and the "Generate Post-Place File" needs to be set to "True". The .posp file is used for the FloorPlanner to read the layout location, as shown in Figure 5-1.

Synthesis Place & Route Dual	-Purpose Pin   BitStream
Category: All	Reset all to default
Label	Value
Generate SDF File	False
Generate Constraint File of Ports	False 📃
Generate IBIS File	False
Generate Post-Place File	True
Generate Post-PNR Simulation Mo	del File False True
Initialize Primitives	False
Print BSRAM Initial Value	True
Generate Post-Place File. Default	: *. posp

**Figure 5-1 Posp File Settings** 

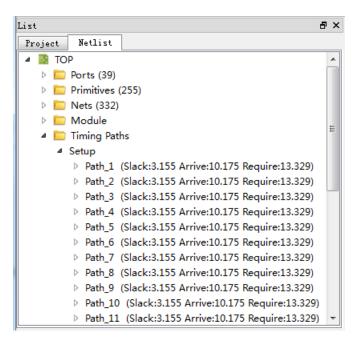
- 5. View the Timing Report to verify whether the Max. frequency meets the needs or not. If not, use FloorPlanner to generate multi-constraints and multi-iterations to achieve timing closure.
- 6. Read the physical constraints and timing constraints files.
- 7. Run the FloorPlanner tool. Open the .posp file after placement and routing has been performed. See the "Netlist" window for the .posp files, ports, primitives, and location constraints information, as shown in Figure 5-2.

Figure 5-2 Read .posp File



8. Open the .timing\_paths file. The critical path information is listed in the "Netlist > Timing Paths" window. This includes the slack, arrival time, require time, etc. of each path, as shown in Figure 5-3.

Figure 5-3 Read Timing Constraint File



#### Note!

In the process of repeated debugging, there is no requirement to open the .posp file and .timing\_paths file each time. You can reload them by using the "Reload" option. 9. Analyze and adjust constraints location.

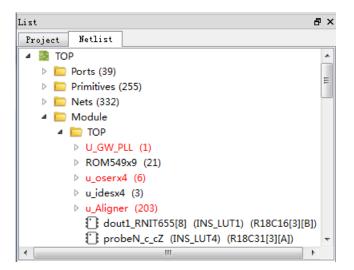
Finding the key path and modifying the code or device location may help to achieve timing closure. In the FloorPlanner tool, you can also adjust the pin location for timing closure. The steps required to do this are as follows:

- Find the module info. in the key path:
  - Check the instance info. along the key path and find the module from the instance name.
  - Right-click on the key path and select "Highlight Corresponding Module", as shown in Figure 5-4. The modules will be highlighted in red, as shown in Figure 5-5.

Figure 5-4 Module Operation of Highlighting Key Path

List	6	×
Project Netlist		
🚺 dout1_Z[0]	(INS_DFFC) (R17C7[1][A])	*
🚺 dout1_Z[8]	(INS_DFFC) (R16C30[0][A])	
🔂 dout1_Z[7]	(INS_DFFC) (R18C7[1][B])	
🚺 dout1_Z[6]	(INS_DFFC) (R17C7[0][B])	
🚺 dout1_Z[5]	(INS_DFFC) (R18C7[1][A])	
🚺 dout1_Z[4]	(INS_DFFC) (R18C7[0][A])	
🔺 🚞 Timing Paths		
▲ Setup		
▷ Path_1 (S	Highlight	7
▷ Path_2 (S	Highlight Corresponding Module	
⊳ Path_3 (S	Highlight Corresponding Module	=
▷ Path_4 (S	Edit Constraint	
▷ Path_5 (S <del>rack,</del>	3.133 AHIVE.10.173 Nequile.13.325)	-
	3.155 Arrive:10.175 Require:13.329)	
	3.155 Arrive:10.175 Require:13.329)	
Path_8 (Slack:	3.155 Arrive:10.175 Require:13.329)	
	3.155 Arrive:10.175 Require:13.329)	
Path_10 (Slack	::3.155 Arrive:10.175 Require:13.329)	Ŧ

Figure 5-5 Highlight Module of Key Path in Red



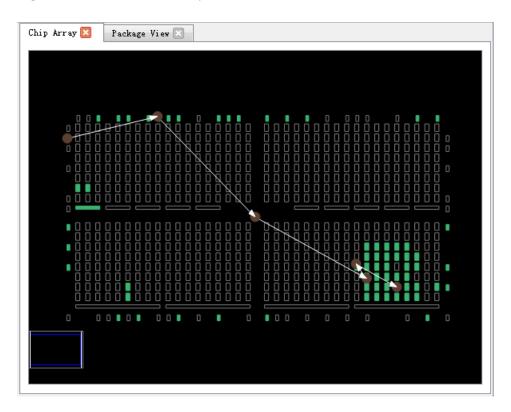
- Check the module location and key path signal flow.
  - Select one Module, right-click and select "Highlight Group Constraints". The group module location will be highlighted in white in the chip array view.
  - During timing closure, the key path signal flow is important. Select one key path in the FloorPlanner tool, right-click and select "Highlight". Check the path signal flow in the "Chip Array" view, as shown in Figure 5-6.

#### Figure 5-6 Key Path Signal Flow

Chip Array 🔀	ackage View 🔀	
		ו

• Adjust improper locations.

As shown in Figure 5-6, the locations are relatively centralized; only one is located far apart. Check the key path signal flow. A winding path with a large span influences timing. You can drag the location that is out of line and shorten the winding path, as shown in Figure 5-7.



**Figure 5-7 Location After Adjustment** 

10. Rerun "Place & Route" to view the timing result. If the frequency does not meet the requirements, repeat Steps 5-9.

# Appendix **A** Physical Constraints

## **Syntax Definition**

## A.1 I/O Constraints

The registers (port, buffer, I/O) and the I/O logic devices can be constrained to the specified IOB location using the IO constraint.

#### Syntax

"IO\_LOC" """obj\_name""" obj\_location ["exclusive"] ";"

#### **Constraint Elements**

• obj\_name

obj\_name: the name of the port  $\$  the I/O buffer, the I/O register, and the I/O logic device can be used for obj\_name.

obj\_location

obj\_location is the IOB location. For example "A11", "B12" etc. If multiple locations are specified, the locations need to be separated by commas, such as "A11, B2".

exclusive

Exclusive is optional. After the constraint position, the device that specified by the obj\_name can only be placed in the obj\_location of the constraint statement.

#### Note!

When the obj\_name is escaped name format (starting with a backslash and ending with a space), the obj\_name must be quoted on both sides.

#### **Examples and Explanation**

Example 1:

IO\_LOC "io\_1" A1;

//io\_1 should be located to the pin A1.

Example 2:

IO\_LOC "io\_1" A1, B14, A15;

//io\_1 should be located in the pin A1, pin B14, or pin A15, one of the three positions will be taken for the layout.

Example 3:

IO\_LOC "io\_2" A1 exclusive;

// io\_2 should be located in pin A1, and pin A1 can only be used by io\_2.

Example 4:

IO\_LOC "io\_2" A1, B14, A15 exclusive;;

//io\_2 should be located in pin A1, B14, or A15, and all these locations can only be used by io\_2.

## **A.2 PORT Attributes Constraints**

Port attribute constraint is used for setting various attribute values of the ports. For example, the level standard IO\_TYPE of the port, PULL\_MODE (pull-up/pull-down mode), drive capability DRIVE, etc.. Please refer to the corresponding data sheet for the detailed attribute setting standards.Syntax

"IO\_PORT" """port\_name """ attribute "=" attribute\_value ";"

Multiple attributes can be set in a constraint statement, Each attribute can be separated by spaces.

#### **Constraint Element**

- The port name that requires the attribute constraints
- attribute and attribute value

#### **Examples and Explanation**

Example 1:

```
IO_PORT "port_1" IO_TYPE = LVTTL33;
```

// Set the IO\_TYPE at port\_1 as "LVTTL33".

Example 2:

IO\_PORT "port\_2" IO\_TYPE = LVTTL33 SLEW\_RATE = FAST PULL\_MODE =KEEPER;

// Set the IO\_TYPE at port\_2 as the "LVTTL33', SLEW\_RATE value is "FAST", PULL\_MODE value is "KEEPER".

Example 3:

IO\_PORT "port\_3" IO\_TYPE=LVDS25;

// BUF at port\_3 is general IBUF, transform the IBUF as TLVDS\_IBUF through the constraint  $_{\circ}$ 

Example 4:

IO\_PORT "port\_4" I3C\_MODE=ON OPEN\_DRAIN=OFF;

// I3C\_MODE/MIPI\_INPUT/MIPI\_OUTPUT attribute is available only in the GW1N6K/GW1N9K/GW1NR9K/ GW1NS2K/ GW1NSR2K; The I3C\_MODE attribute and the OPEN\_DRAIN attribute can't be set as "ON".

## **A.3 Primitive Constraints**

Primitive Constraints are used to lay out the instances to the specified GRIDs. The constraints (LUT/BSRAM/SSRAM/DSP/PLL/DLL) and other instances can be restrained using the Primitive Constrains.

#### Syntax

"INS\_LOC" """ obj\_name""" obj\_location ["exclusive"]";"

#### **Constraint Elements**

obj\_name

The instance name.

obj\_location

obj\_location includes:

1) A location information, specifies to the LUT, such as: RxCy[0-3][A-B]

2) A range of the location information, specifies to the Multiple rows or columns, such as:

Includes multiple PLS or LUT: "RxCy", "RxCy[0-3]"

To specify multiple rows: "R[x:y]Cm", "R[x:y]Cm[0-3]", "R[x:y]Cm[0-3][A-B]"

To specify multiple columns: "RxC[m:n]", "RxC[m:n][0-3]", "RxC[m:n][0-3][A-B]"

To specify multiple rows and columns: "R[x:y]C[m:n]", "R[x:y]C[m:n][0-3]", "R[x:y]C[m:n][0-3][A-B]"

Multiple ins\_locations can be included in a constraint statement, which are separated by ",".

3) PLL Constraint Location

For the "PLL\_L" or "PLL\_R" of the PLL constraint position information, if more than one PLL are placed on the left side, it can be set to "PLL\_L[0]", "PLL\_L[1]" ...; if more than one PLL are placed on the right side, it can be set to "PLL\_R[0]", "PLL\_R[1]" ...

4) DLL Constraint Location

The DLL constraint locations are "DLL\_TL", "DLL\_BL", "DLL\_TR", "DLL\_BR".

5) BSRAM Constraint Location

The BSRAM constraint locations are "BSRAM\_R10[0]" (the first BSRAM at Line 10), "BSRAM\_R10[1]"....

6) DSP Constraint Location

The DSP constraint locations are "DSP\_R19[0]" (the first DSP Block at Line 19), "DSP\_R19[1]"... To specify a specific macro, it can be marked as: DSP\_R19[0][A] or DSP\_R19[0][B].

• exclusive

The keyword "exclusive" is optional. After the location is restrained, the obj\_location in the constraint statement can only place the instance specified by obj\_name.

#### **Examples and Explanation**

Example 1

INS\_LOC "lut\_1" R2C3, R5C10[0][A];

// lut\_1 is constrained at the R2C3 and the first LUT of the 0th PLS

of the R5C10.

#### Example 2

INS\_LOC "ins\_2 " R5C6[2] exclusive;

// ins\_2 is constrained at the 2nd PLS of the R5C6, and only the instance can be placed at this location.

#### Example 3

INS\_LOC "ins\_3" R[2:6]C1;

// ins\_1 is constrained in the rows between the row 2 and the row 6, and in the column 1.

Example 4

INS\_LOC "ins\_4" R[1:4]C[2:6] exclusive;

// ins\_3 is constrained in the row between row 1 and row 4, and in the column between column 2 and column 6. The location of the region can only be occupied by the instance.

Example 5.

INS\_LOC "ins\_5" R[1:4]C[2:6][1];

// ins\_4 is constrained in the row between row 1 and row 4, and the first PLS of a GRID in the column between column 2 and column 6.

Example 6

INS\_LOC "reg\_name" B14;

// It is constrained to the IOB by restraining the INS\_LOC of the REGISTER/IOLOGIC.

Example 7

INS\_LOC "dll\_name" DLL\_TL;

// It is constrained to the DLL top left corne by restraining the INS\_LOC

of the DLL.

Example 8

INS\_LOC "pll\_name" PLL\_L;

// It is constrained to the PLL left by restraining the INS\_LOC of the

PLL.

#### Example 9

INS\_LOC "bsram\_name" BSRAM\_R10[2];

// It is constrained to the third BSRAM in line 10 by restraining the INS\_LOC of the BSRAM.

Example 10

INS\_LOC "dsp\_name" DSP\_R19[2];

// It is constrained to the third DSP in line 19 by restraining the INS\_LOC of the DSP.

Note!

A LUT4 can be placed in a lut1/lut2/lut3/lut4, a lut5 needs to occupy two LUT4s (one PLS), a lut6 needs to occupy four LUT4s (two PLS), a lut7 needs to occupy four PLSs (one GRID), and a lut8 needs to occupy eight PLSs (two GRIDs). Therefore, for constraints of the different Instance type, the minimum unit of the constraint location is also different. For the BSRAM/SSRAM/DSP (one DSP unit includes two MICROs and one MICRO includes two UNITs). The examples are as follows:

#### Example 11 LUT4 Unit Constraint

INS\_LOC "lut4\_name" R5C15[1][A];

// lut4\_name is constrained to the first LUT in the first PLS of the

R5C15.

#### Example 12 PLS Unit Constraint

INS\_LOC "lut5\_name" R5C15[3];

// lut5\_name is constrained to the third PLS of the R5C15.

#### Example 13 PLS Unit Constraint

INS\_LOC "lut6\_name" R5C15[0];

// lut6\_name is constrained to the zeroth PLS of the R5C15 (the PLS[0] and PLS[1] will be occupied).

#### Example 14 GRID Unit Constraint

INS\_LOC "lut7\_name" R5C15;

BSRAM type: INS\_LOC "bsram\_name" R10C5; // for GW2A55K

// lut7\_name is constrained to the R5C15, the LUT7 will occuppy one

**GRID**。

#### Example 15 GRID Unit Constraint

INS\_LOC "lut8\_name" R5C15;

// lut8\_nam is constrained to the R5C15, *lut8\_name* will occuppy two GRID of the R5C15 and the R5C16。

#### Example 16 DSP MICRO Unit Constraint

INS\_LOC "mult\_name" DSP\_R19[1][A]; // for GW2A55K

// mult\_name is constrained to the the first macro of the second DSP

in line 19.

## **A.4 Primitive Group Constraints**

Primitive GroupConstraint is used to define a group constraint, the group

is a integration containing various of Instance objects. The general Instances such as LUT, DFF, etc., or BUF, IOLOGIC, etc. can be added to a group using the Primitive Group constraints. And the location constraints of all objects in the group can be achieved by constraining the location of the group.

#### Syntax

GROUPdefinition:

"GROUP" group\_name "=" "{" ""obj\_names """ "}" ["exclusive"]";"

Add the Instance to the group:

• "GROUP" group\_name "+=" "{" """ obj\_names """ "}" ["exclusive"]";"

The location of the group is constrained:

• "GRP\_LOC" group\_name group\_location["exclusive"]";"

#### Note!

If group\_name's format is the format of escaped name (begin with "\" and end with a space), the quotes at two sides of group\_name are necessary.

#### **Constraint Element**

group\_name

Define a name as the name of the group

obj\_name

Obj\_name is used to add the specified Instance object to the group

• group\_location

Specify the constraint location of the group, the group\_location can at the IOB and the  $\ensuremath{\mathsf{GRID}}$ 

exclusive

The keyword "exclusive" is optional, which is at the end of the group definition statement or the location constraint statement;

An object can be included in multiple groups, but the object can only be included in the group that the "exclusive" keyword is added at the end of the group definition statement;

The "exclusive" at the end of the location constraint statement indicates that the constraint location can only be occupied by the objects within the group.

#### **Examples and Explanation**

#### Example 1

GROUP group\_1 = { "ins\_1" "ins\_2" "ins\_3" "ins\_4" };

// Create a group named group\_1 and add the objects ins\_1, ins\_2, ins\_3, ins\_4 into the group.

#### Example 2

GROUP group\_2 = { "ins\_5" "ins\_6" "ins\_7" } exclusive;

// Create a group named group\_2 and the objects ins\_5, ins\_6, ins\_7, ins\_4 can only be added into the group.

#### Example 3

GROUP group\_1 += { "io\_1" "io\_2"};

// Add io\_1, io\_2 into group\_1.

#### Example 4

GRP\_LOC group\_1 R3C4, A14, B4;

// The objects in the group\_1 can be placed at R3C4, A14, B4.

#### Example 5

GRP\_LOC group\_2 R[1:3]C[1:4] exclusive;

// The Instance object in group\_2 can be placed in the range of region R[1:3]C[1:4], and the Instance object in group\_2 can only be placed in the scope.

## A.5 Resource Reservation

The specified location or region can be reserved using the Resource Reservation constraint to avoid it is occupied in the layout.

#### Syntax

• "LOC\_RESERVE" location [ res\_obj ] ";"

#### **Examples and Explanation**

#### Example 1

LOC\_RESERVE R2C3[0][A] -LUT;

LOC\_RESERVE R2C3[0][A] -REG;

#### Example 2

LOC\_RESERVE IOR3, IOR6, R2C3, R3C4;

#### Example 3

LOC\_RESERVE R[2:5]C[3:6], R3C[8:9];

// The location information for the constraints in the above examples will be preserved during the layout phase.

## A.6 Relative Group Constraints

The relative position constraints on the instance object can be achieved using the Relative Group Constraints.

#### Syntax

A group that defines a Relative constraint:

"REL\_GROUP" group\_name "=" "{" ""obj\_names """ "}";"

Add the Instance to the defined group:

• "REL\_GROUP" group\_name "+=" "{" "" obj\_names """ "}";"

The instance is restrained on the relative location in the group:

"INS\_RLOC" "" obj\_name"" relative\_location ";"

#### **Constraint Element**

• obj\_name

The name of the constraint object.

• relative\_location

The information description on the relative location in row and column.

#### **Examples and Explanation**

#### Example 1

REL\_GROUP grp\_1 = { "ins\_1" "ins\_2" "ins\_3" "ins\_4" }; INS\_RLOC "ins\_1" R0C0; INS\_RLOC "ins\_2" R2C3; INS\_RLOC "ins\_3" R3C5;

// Define a group constraint named grp\_1 and add the ins\_1, ins\_2, ins\_3, ins\_4 into grp\_1. The ins\_1 is the relative location origin point R0C0, the ins\_2 is constrained to the R2C3 relative to the ins\_1, and the ins\_3 is constrained to the R3C5 relative to ins\_1.

## A.7 Vref Constraints

The chip supports the external reference voltage. Each PAD of the chip (include IOLOGIC) can be used as an input PAD for the external reference voltage, which is valid for the entire BANK. The Vref Constraints can be used to constrain the name and location of the input pin of the external reference voltage.

#### Syntax

• "USE\_VREF\_DRIVER" vref\_name [location]";"

#### **Constraint Element**

vref\_name

Customized VREF pin name

location

Any PAD (include IOLOGIC) location in the chip can be used as a location for the VREF pin constraints.

#### **Examples and Explanation**

#### Example 1

USE\_VREF\_DRIVER vref\_pin;

IO\_PORT "port\_1" IO\_TYPE = SSTL25 VREF=vref\_pin;

IO\_PORT "port\_2" IO\_TYPE = SSTL25 VREF=vref\_pin;

// Define a VREF pin named "vref\_pin" and set the VREF attribute of
port\_1 and port\_2 to vref\_pin.

#### Example 2

USE\_VREF\_DRIVER vref\_pin C7;

IO\_PORT "port\_1" IO\_TYPE = SSTL25 VREF=vref\_pin;

IO\_PORT "port\_2" IO\_TYPE = SSTL25 VREF=vref\_pin;

// Define a VREF pin named "vref\_pin", constrain it to PAD C7 (bank 3, GW1N-4, WLCSP72), set the VREF value of port\_1 and port\_2 to vref\_pin, and port\_1 and port\_2 will be placed in the bank that C7 locates.

## A.8 Quadrant Constraints

The Quadrant is used to constrain objects such as DCS/DQCE that

require quadrant layout to a specified quadrant (the GW1N family has LEFT and RIGHT quadrants, and the GW1N6K/GW1N9K/GW1NR9K and GW2A families have four Quadrants: TOPLEFT, TOPRIGHT, BOTTOMLEFT, BOTTOMRIGHT, see the relevant data sheet for the specific information).

#### Syntax

• "INS\_LOC" """obj\_name""" quadrant ";"

#### **Constraint Element**

obj\_name

The name of the constraint object.

• quadrant

GW1N series: "LEFT" ("L"), "RIGHT" ("R")

The GW1N6K/GW1N9K/GW1NR9K and GW2Aseries: "TOPLEFT"("TL"), "TOPRIGHT"("TR"), "BOTTOMLEFT"("BL"), "BOTTOMRIGHT"("BR")

(Note: Abbreviations are in parentheses.)

#### **Examples and Explanation**

#### Example 1

INS\_LOC "dcs\_name" LEFT;

// Constrain the DCS object dcs\_name to the LEFT quadrant (GW1N family).

## A.9 Clock Assignment

The Clock Assignment constraint is a constraint on a specific wire to global clock line in the design. There are eight main clocks and eight long-line resources in each quadrant of the chip resource. This constraint can be used to implement the routing constraints for the global clock line on the wire of the specific fanout (CLK/CE/SR/LOGIC) of the net.

BUFG[0-7] represents the resources of eight master clocks.

BUFS representseightLong-term resources.

The CLK signal is the wire signal connected to the CLK pin, the CE signal is the wire signal connected to the CE pin, the SR signal is the wire signal connected to the SET/RESET/CLEAR/PRESET pin, and the LOGIC is the wire signal connecting the output pins of other logic devices.

#### Syntax

• "NET\_LOC" """net\_name """ global\_clocks "=" fanout [quadrant]";"

#### **Constraint Element**

net\_name

net name

global\_clocks

BUFG[0-7] represents the resources of eight master clocks.

BUFS[0-7] represents the resources of eight Long terms.

• fanout

CLK: fanout is the wire of the CLK

CE: fanout is the wire of the CE

SR: fanout is the wire of SET/RESET (synchronous reset signal), CLEAR/PRESET (asynchronous reset signal)

LOGIC: fanout is the wire excluding the fanout

ALL: All fanout wire

Specify multiplefanouts, The symbol "|" can be used to separated.

quadrant

GW1N series: "LEFT" ("L"), "RIGHT" ("R")

The GW1N6K/GW1N9K/GW1NR9K and GW2Aseries: "TOPLEFT"("TL"), "TOPRIGHT"("TR"), "BOTTOMLEFT"("BL"), "BOTTOMRIGHT"("BR")

(Note: The word in the parentheses is an abbreviation, the quadrant constraint keyword is valid only when the main clock BUFG[0-7] resource is specified.)

#### **Examples and Explanation**

#### Example 1

NET\_LOC "net" BUFG[0] = CLK LEFT;

// Constrain the wire of the CLK fanout of the NET object net to the 0th main clock resource (GW1N family) of the LEFT quadrant.

#### Example 2

NET\_LOC "net" BUFG = CLK|CE;

// Constrain the wire of the CLK fanout of the NET object net and the wire of the CE fanout to the main clock resource.

#### Example 3

NET\_LOC "net" BUFS = CE;

// Constrain the wire of the CE fanout of the NET object net to the long term resources.

## A.10 Hclk Constraints

The CLKDIV/DLLDLY can be constrained to the relevant location through the CLKDIV/DLLDLY constraint. The constraint location at CLKDIV/DLLDLY is different from the normal instance object constraint position. The "TOPSIDE", "BOTTOMSIDE", "LEFTSIDE", and "RIGHTSIDE" indicates the four sides of the constraint location.

#### Syntax

"INS\_LOC" """ location";"

#### **Constraint Element**

obj\_name

The instance name of the CLKDIV/DLLDLY is the obj\_name.

location

"TOPSIDE[0-1]" ("TS[0-1]")

"BOTTOMSIDE[0-1]" ("BS[0-1]")

"LEFTSIDE[0-1]" ("LS[0-1]")

"RIGHTSIDE[0-1]" ("RS [0-1]")

(Note: Abbreviations are in parentheses.)

#### **Examples and Explanation**

Example 1

INS\_LOC "clkdiv\_name" TS[0];

// Place the clkdiv\_name to TOPSIDE[0].

## Appendix **B** Timing Constraints Syntax Definition

## **B.1 Clock Constraints**

## B.1.1 create\_clock

#### Syntax

Command: create\_clock Parameter: -period <period\_value> [-name <clock\_name>] [-waveform <edge\_list>] <source\_objects> [-add]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-period**: Specify the clock period. The parameter value should be greater than 0, and the period unit is ns.

**-name**: Specify the clock name. The clock name must be unique. If the new clock has the same name as an existing clock, the existing clock will be overwritten with the new clock. If the name is not specified, the name of the first source object will be regarded as the clock name.

**-waveform**: Specify the time of the rising edge and falling edge of the clock. The difference time between the rising edge and falling edge should be less than one period. In general, if the rising edge arrives first, both the rising edge and the falling edge time should be less than one period. For example, "{0 5}" means the clock rising edge arrives at 0 ns, and the clock falling edge arrives at 5 ns; if the clock falling edge arrives, set the clock rising edge time to less than one period, and the falling edge time to equal to or greater than one period. If the period is set to 10 ns, "-waveform {5 10}" means the clock falling edge arrives at 5 ns.

-add: Use the -add option to add multiple clocks to the same source object, or a new clock with a different clock name on the same source object will be ignored when the source object already has a clock.

-source\_objects: Specify the actual source objects that the clock arrives at, such as PORT, PIN, NET, etc. When the source object already has a clock, the new clock with a different clock name on the same source object will be not ignored if the –add option is specified. Users can add multiple clocks to the same source object with the –add option. If the source object is not specified when the clock is created, the new clock will be ignored.

#### **Examples:**

# Create a clock that has a period of 10 ns and the falling edge arrives first:

create\_clock -name clk -period 10.000 -waveform {5 10} [get\_ports {clk}]

# Create a clock with the duty cycle of 40%:

create\_clock -name clk -period 10.000 -waveform {6 10} [get\_ports {clk}]

# Create two clocks to one input port:

- 1. create\_clock -period 10 -name clk # command is ignored and no clk can be created
- create\_clock -period 10 -name clk [get\_ports {clk}] # Create clk successfully
- 3. create\_clock -period 10 -name clk1 [get\_ports {clk}] # commands is ignored and no clk can be created because -add is not used.

create\_clock -period 20 -name clk1 -add [get\_ports {clk}]

# Create clk1 successfully.

## B.1.2 create\_generated\_clock

#### Syntax

Command: create\_generated\_clock

Parameter: [-name <clock name>]

-source <master pin>

[-edges <edge list>]

[-edge\_shift <shift list>]

[-divide\_by <factor>]

[-multiply\_by<factor>]

[-duty\_cycle <percent>]

[-add]

[-invert]

[-master\_clock <clock>]

[-phase <phase>]

[-offset <offset>]

<source objects>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-name**: Specify the name of the generated clock. If –name is not specified, the name of the first source object will be regarded as the clock name. The clock name must be unique. If the new clock has the same name as an existing clock, the existing clock will be overwritten with the new clock.

**-source**: Specify the source from which the generated clock is from. If there is more than one clock on this object, the -master\_clock option must be used to specify the original clock.

-master\_clock: Specify the master clock of the generated clock.

-edges: Specify the edge list of the generated clock. This option specifies a three positive ascending order integer parameter list, which

indicates the relationship between the first rising edge of the generated clock, the first falling edge of the generated clock, the second rising edge of the generated clock, and the master clock edge. For instance, we mark the first rising edge of the master clock as 1, the next falling edge is 2, the next rising edge is 3 ..., the marker numbers of the master clock edges are elements of the edge list; we can create a two divider generated clock with "-edges {1 3 5}".

**-edge\_shift**: Specify the edge shift of each edge. This option should be used together with the "-edges" option. It can be specified as any number, but the edge cannot go beyond the adjacent border.

#### Note!

"-edge" and "-edge\_shift" cannot be used together with the other parameters used for waveform adjustment, except "-invert".

-divide\_by: Specify the divider value; the value should be a positive integer.

**-multiply\_by**: Specify the multiplier value; the value should be a positive integer.

**-duty\_cycle**: Specify the duty cycle of the generated clock; the value should be a positive integer lower than 100.

-add: When specifying this option, this clock can coexist with an existing clock.

-invert: Specify if the waveform of the generated clock is inverted.

-phase: Specify the phase shift of clock edges in degrees.

-offset: Specify the offset of clock edges in time values.

**-source object**: Specify the actual source objects that the generated clock arrives at, such as PORT, PIN, NET, etc.

Examples and Explanation

# Create a 2\*divider generated clock to port A by using "-divide\_by":

create\_clock -period 10 [get\_ports clk]

create\_generated\_clock -name genClk -source [get\_ports {clk}] -divide\_by 2 [get\_ports {a}]

# Create a 2\*divider generated clock to port a by using "-edges":

create\_generated\_clock -name genClk -source [get\_ports {clk}] -edges {1 3 5} [get\_ports {a}] # Create a 2\*multiplier clock with a 40% duty cycle:

create\_generated\_clock -name genClk0 -source [get\_ports {clk}] -multiply\_by 2 -duty\_cycle 40 [get\_pins {pll\_out}]

# Create an inverted clock 2\*divider relative to the output of the source clock:

create\_generated\_clock -name genClk1 -source [get\_ports {clk}]
-divide\_by 2 -invert [get\_pins {pll\_out}]

# Create a clock 2\*multiplier with a 90-degree phase shift:

create\_generated\_clock -name genClk2 -source [get\_ports{clk}] -multiply\_by 2 -phase 90[get\_pins {pll\_out}]

# Create a 2\*divider generated clock:

create\_generated\_clock -name genClk3 -source [get\_ports {clk}] -edges {2 4 6}[get\_pins {pll\_out}]

#Create two clocks to an input port that are switched externally:

create\_clock -period 10 -name clk [get\_ports {clk}]

create\_clock -period 20 -name clk1 -add [get\_ports {clk}]

create\_generated\_clock -name genClk -source [get\_ports {clk}]
-divide\_by 2 -master\_clock clk -add [get\_pins {pll\_out}]

create\_generated\_clock -name genClk1 -source [get\_ports {clk}]
-master\_clock clk1 -divide\_by 2 -add [get\_pins {pll\_out}]

B.1.3 set\_clock\_latency

Syntax

Command: Set\_clock\_latency

Parameter: -source [-rise | -fall]

[-late | -early]

<delay>

[-clock <clock list>]

<object list>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-source: Specify the clock latency type of source.

**-rise | -fall**: Specify the rising | falling clock latency. -rise| -fall cannot be specified in one statement. If both are not specified, the clock latency is applied to all conditions.

#### Note!

The user needs to specify the source latency value. The default value for Gowin YunYuan software is 0 ns.

-late | -early: Specify the late | early clock latency; -late is used for regular setup analysis, and -early is used for regular hold analysis.

<delay>: Specify the clock latency value.

#### Note!

The default setting provided by STA is 0 ns.

Users can specify the clock that the source latency affects when more than one clocks are on one source object. If this option is used, all clock have the same delay.

<source objects>: Specify the source objects that the clocks arrive at.

#### **Examples and Explanation**

create\_clock -period 10 -name clk [get\_ports {clk}]
create\_clock -period 10 -name clk0 [get\_ports {clk}] -add
# Specify 2 ns clock latency for clk
set\_clock\_latency -source 2 [get\_clocks {clk}]
# Specify 2 ns clock latency for clk0
set\_clock\_latency -source 2 -clock [get\_clocks {clk0}] [get\_ports {clk}]

#### **B.1.4 set\_clock\_uncertainty**

#### Syntax

Command: Set\_clock\_uncertainty

Parameter: [-from <from clock>]

[-rise\_from <rise from clock>]

[-fall\_from <-fall from clock>]

[-to <to clock>]

[-rise\_to <rise to clock>]

[-fall\_to <fall to clock>] [-setup | -hold] <uncertainty value>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-from/-rise\_from/-fall\_from**: Specify from the clock list. Specify the clock edge with "-rise\_from" and "-fall\_from".

**-from/-rise\_from/-fall\_from**: Specify to the clock list. Specify the clock edge with "-rise\_to" and "-fall\_to".

-setup/-hold: Specify that clock uncertainty affects setup or hold analysis; If both "-setup" and "-hold" are not specified, this uncertainty value is applied to both analysis types.

<uncertainty value>: Specify clock uncertainty value.

#### Note!

The default setting value provided by STA is 0.02 ns.

#### **Examples and Explanation**

# Set the clock uncertainty setup time from clk to clk to 0.5:

set\_clock\_uncertainty -setup -from clk -to clk 0.5

# Set the clock uncertainty hold time from clk0 to clk to 0.0:

set\_clock\_uncertainty -hold -from clk0 -to clk 0.0

#### B.1.5 set\_clock\_groups

#### Syntax

Command: Set\_clock\_groups

Parameter: [-asynchronous | -Exclusive]

-group <clock name>

-group <clock name>

[-group <clock name>] ...

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-asynchronous | -Exclusive: Specify the created clock groups relationship as exclusive;

-group: Create clock group;

#### **Examples and Explanation**

# Set the relationship between clk and clk0 as exclusive

set\_clock\_groups -Exclusive -group [get\_clocks {clk}] -group
[get\_clocks {clk0}]

## **B.2 I/O Constraints**

## B.2.1 set\_input\_delay

#### Syntax

Command: Set\_input\_delay Parameter: -clock clock\_name [-clock\_fall] [-rise] [-fall] [-max] [-min] [-add\_delay] [-source\_latency\_included] <delay\_value> <port\_list>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects;
- Supports the SDC constraint type.

-clock: Specify the clock name;

-clock\_fall: Specify that the input delay is relative to falling clock edge;

#### Note!

If the option is not specified, the input delay is relative to rising clock edge by default.

**-rise/-fall**: Specify rise | fall input delay. If only one of them is specified, the specified input delay is applied to the other.

-max/-min: Specify max | min input delay. If only one of them is specified, the specified input delay is applied to the other.

-add\_delay: When this option is specified, constraints that have already been input will not be overwritten.

-source\_latency\_included: Specify that the input delay has already included source latency.

#### Note!

If this option is specified, the source latency is added to input delay.

<delay\_value>: Specify input delay value;

#### Note!

The default STA input delay value is 0ns.

ort\_list>: Specify port list for this constraint;

#### **Examples and Explanation**

# Set the input delay based on clk rising edge for port a as 0.8 ns:

set\_input\_delay -clock clk 0.8 [get\_ports {a}]

# Set the input delay based on clk rising edge for all input ports as 0.8 ns:

set\_input\_delay -clock clk 0.8 [all\_inputs]
# Set the input delay based on clk falling edge for port a as 0.8 ns:
set\_input\_delay -clock clk -clock\_fall 0.8 [get\_ports {a}]
# Create input delays for different min/max and rise/fall combinations:
set\_input\_delay -clock clk -max -rise 1.4 [get\_ports {a}]
set\_input\_delay -clock clk -max -fall 1.5 [get\_ports {a}]
set\_input\_delay -clock clk -min -rise 0.7 [get\_ports {a}]

# Create several related input delays with more than one clock: set\_input\_delay -clock clk0 -min 1.2 [get\_ports {a}] set\_input\_delay -clock clk0 -max 1.8 [get\_ports {a}] set\_input\_delay -clock clk0 -clock\_fall 1.6 -add\_delay [get\_ports a] set\_input\_delay -clock clk1 -min 2.1 -add\_delay [get\_ports {a}] set\_input\_delay -clock clk1 -max 2.5 -add\_delay [get\_ports {a}]

#### B.2.2 set\_output\_delay

#### Syntax

Command: Set\_output\_delay Parameter: -clock clock\_name [-clock\_fall] [-rise] [-fall] [-max] [-max] [-min] [-add\_delay] [-source\_latency\_included] <delay\_value> <port\_list>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-clock: Specify clock name related with output delay.

-clock\_fall: Specify the clock reference edge of output delay.

#### Note!

Rising edge is the default reference edge.

**-rise/-fall**: Specify rise | fall input delay. If only one of them is specified, the specified input delay is applied to the other.

**-max/-min**: Specify max | min input delay. If only one of them is specified, the specified input delay is applied to the other.

**-add\_delay**: When this option is specified, constraints that have already been input will not be overwritten.

-source\_latency\_included: Specify that the input delay has already included source latency.

<delay\_value>: Specify output delay value.

#### Note!

The default STA output delay value is 0 ns.

ort\_list>: Specify port list for this constraint;

#### **Examples and Explanation**

# Set the output delay of port b as 0.5 ns: set output delay -clock clk 0.5 [get ports {b}] # Set the output delay of all ports as 0.5 ns: set output delay -clock clk 0.5 [all outputs] # Set the output delay based on falling edge for all ports as 0.5 ns: set output delay -clock clk -clock fall 0.5 [get ports {b}] # Set the output delay based on the rising edge for all port b: set output delay -clock clk -max -rise 0.3 [get ports {b}] set output delay -clock clk -max -fall 0.5 [get ports {b}] set output delay -clock clk -min -rise 0.8 [get ports {b}] set output delay -clock clk -min -fall 0.7 [get ports {b}] # Create several input delays related to more than one clock: set output delay -clock clk0 -min 0.5 [get ports {b}] set\_output\_delay -clock clk0 -max 0.6 [get\_ports {b}] set output delay -clock clk0 -clock fall 0.7 -add delay [get ports {b}] set output delay -clock clk1 -min 0.8 -add delay [get ports {b}] set output delay -clock clk1 -max 0.9 -add delay [get ports {b}]

## **B.3 Path Constraints**

## B.3.1 set\_max\_delay | set\_min\_delay

Syntax

Command: Set\_max\_delay Parameter: [-from <from list>] [-to <to list>] [-through <through\_list>] <delay value> Command: Set\_min\_delay Parameter: [-from <from list>] [-to <to list>] [-through <through\_list>] <delay value>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-from**: Specify for the begining point; "-from" objects can be PORTS, NETS, REGS, CLOCKS, PINS, etc.

-to: Specify for the ending point. "-to" objects can be PORTS, NETS, REGS, CLOCKS, PINS, etc.

**-through**: Specify through objects list; through objects can be nets, pins, etc. Only one "-through" can be used in one statement at one time.

#### Note!

The three parameters described above can be used in combination or alone. If the specified objects of the three parameters are not on the same path, STA will ignore this constraint, and timing analysis will not be affected.

#### **Examples and Explanation**

# Set 5 ns max delay between two clocks:

set\_max\_delay -from [get\_clocks {clk}] -to [get\_clocks {clk}] 5

# Set 2 ns max delay from port a to register 0: set\_max\_delay -from [get\_ports {a}] -to [get\_registers {reg0}] 2 # Set 2 ns max delay from reg0 to port b: set max delay -from [get registers {reg0}] -to [get ports {b}] 2 # Set 5 ns max delay for all timing objects driven by clocks: set max delay -from [all clocks] 5 -to [get ports {out\*}] # Set 2 ns max delay from port a to port b: set max delay -from [get ports {a}] -to [get ports {b}] 2 #set 2 ns max delay rise from regs to fall clock: set\_max\_delay\_from [get\_regs {reg0}] - to [get\_clocks {clk}] 2 # Set 5 ns min delay between two clocks: set min delay -from [get clocks {clk}] -to [get clocks {clk}] 0.5 # Set 0.5 ns min delay from port a to register 0: set min delay -from [get ports {a}] -to [get registers {reg0}] 0.5 # Set 0.5 ns min delay from reg0 to port b: set min delay -from [get registers {reg0}] -to [get ports {b}] 0.5 # Set 0.5 ns min delay from port a to port b: set min delay -from [get ports {a}] -to [get ports {b}] 0.5 # Set 0.5 ns min delay from input port only to falling clock: set max delay -from [get ports {a}] -to [get clocks {clk}] 0.5

#### B.3.2 set\_false\_path

#### Syntax

Command: Set\_false\_path Parameter: [-from <from list>] [-to <to list>] [-through <through list>] [-setup] [-hold]

Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-setup/-hold: Specify constraints for setup time or hold time. The two parameters are mutually exclusive and have an effect on setup time.

**-from**: Specify for the beginning port; users can specify them through "get\_ports", "get\_regs", or "get\_clocks".

**-to**: Specify for the ending port; users can specify them through "get\_ports", "get\_regs", or "get\_clocks".

#### Note!

"-rise\_to/-fall\_to" can only collect from objects through "get\_clocks", and only one of the three parameters outlined can be used in one statement.

**-through:** Specify through objects list; through objects can be nets, pins; multiple pins and nets can be specified. Only one "-through" parameter can be used in the same statement.

#### Note!

"-from/-rise\_from/-fall\_from", "-to/-rise\_to/-fall\_to", and "-through" can be used in combination or alone. If the specified objects of the three parameters are not on the same path, STA will ignore this constraint, and timing analysis will not be affected.

#### **Examples and Explanation**

# Set false path between two unrelated clocks:

set\_false\_path -from [get\_clocks {clk0}] -to [get\_clocks {clk1}]

# Set false-path between two regs:

set\_false\_path -from [get\_regs {reg0}] -to [get\_regs {reg1}]

# Set false path rising from regs to falling clock:

set\_false\_path -rise\_from [get\_clocks {clk}] -fall\_to [get\_clocks {clk1}]

# Set false path from port a to port b:

set\_false\_path - from [get\_ports {a}] to [get\_ports {b}]

#### B.3.3 set\_multicycle\_path

#### Syntax

Command: Set\_multicycle\_path

Parameter: [-setup|-hold]

[-start|-end]
[-from <from\_list>]
[-rise\_from <rise\_from\_list>]
[-fall\_from <fall\_from\_list>]
[-to <to list>]
[-rise\_to <rise\_to\_list>]
[-fall\_to <fall\_to\_list>]
[-through <through\_list>]
<path multiplier>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-start/-end: Specify that the constraint clock is launch clock or latch clock.

#### Note!

The reference clock of STA is latch clock by default.

-setup/-hold: Specify constraints for setup time or hold time. The two parameters are mutually exclusive.

#### Note!

STA has an effect on setup time by default.

-from/-rise\_from/-fall\_from: Specify from objects list; users can specify them through "get\_ports", "get\_regs", or "get\_clocks". Only one of the three parameters outlined above can be used in one constraint at one time.

**-to/-rise\_to/-fall\_to**: Specify to objects list; users can specify them through "get\_ports", "get\_regs", or "get\_clocks". Only one of the three parameters outlined above can be used in one constraint at one time.

**-through**: Specify through objects list; through objects can be nets, pins; multiple pins and nets can be specified. Only one "-through" parameter can be used in the same statement.

#### Note!

"-from/-rise\_from/-fall\_from", "-to/-rise\_to/-fall\_to", and "-through" can be used in combination or alone. If the specified objects of the three parameters are not on the same path, STA will ignore this constraint, and timing analysis will not be affected.

#### **Examples and Explanation**

create\_clock -name clk -period 10 [get\_ports {clk}]

create\_generated\_clock -name genClk -multiply\_by 2 -source [get\_ports {clk}] [get\_pins {pll\_out}]

# Set end setup multicycle path of 2 with the reference clock genClk:

set\_multicycle\_path -end -setup -from [get\_clocks {clk}] -to [get\_clocks
{genClk}] 2

# Set end setup multicycle path of 2 with the reference clock reg0:

set\_multicycle\_path -start -setup -from [get\_regs {reg0}] -to [get\_regs
{reg1}] 3

set\_multicycle\_path -start -hold -from [get\_regs {reg0}] -to [get\_regs
{reg1}] 1

# Set multicycle constraint of 3 rising from a clock and falling to an object:

set\_multicycle\_path -end -setup -rise\_from [get\_clocks {clk}] -fall\_to
[get\_clocks {clk0}] 3

## **B.4 Operation Conditions Constraints**

#### Syntax

Command: Set\_operation\_conditions

Parameter: [-grade <c|i>]

[-model <slow|fast>]

[-speed <speed>]

[-setup]

[-hold]

[-max]

[-min]

[-max\_min]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- -grade: Specify the device temperature grade, support commercial and industrial; it defaults to black.
- -model: Specify the process corner of timing analysis; it defaults to slow.
- -speed: Specify the device speed grade.
- -setup: Setup time check under current process corner; same function with -max.
- -hold: Setup time check under current process corner; same function with -min.
- -max: Setup time check under current process corner; same function with -setup.
- -min: Setup time check under current process corner; same function with -hold.
- -max\_min: Setup time and hold time check under current process corner; same function with -setup and -hold.

## **B.5 Timing Report**

#### B.5.1 report\_timing

#### Syntax

Command: report\_timing

Parameter:[-setup|-hold|-recovery|-removal]

[-max\_paths <value>]

[-max\_common\_paths < value >]

[-rise\_from <rise\_from\_list>]

[-fall\_from <fall\_from\_list>]

[-to <to list>]

[-rise\_to <rise\_to\_list>]

[-fall\_to <fall\_to\_list>]

[-through <through list>]

[-from clock<from clok>]

[-fall\_from\_clock <from clok>]

[-rise\_from\_clock <from clok>]

[-to\_clock <to clok>]

[-rise\_to\_clock <to clok>]
[-fall\_to\_clock <to clok>]
[-min\_logic\_level]
[-max\_logic\_level]
[-mod\_ins {mod\_ins1 mod\_ins2 ...}]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.
- -setup|-hold|-recovery|-removal: Specify report setup or hold timing path, recovery or removal timing path.
- -max\_paths: Specify the max path number to report.
- -max\_common\_paths: Specify the max path number of each end node to report.
- -rise\_from/-fall\_from: Specify from the objects list.
- -to /-rise\_to /-fall\_to: Specify to the objects list.
- -through: Specify through the objects list.
- -from\_clock /-fall\_from\_clock /-rise\_from\_clock /-to\_ clock /-rise\_to\_ clock
   /-fall\_to\_clock: Specify from clock and to clock.
- -min\_logic\_level/-max\_logic\_levell: Specify the minimum/maximum logic level of the reported timing paths.
- -mod\_ins {mod\_ins1 mod\_ins2 ...}: Specify multiple module instances, separated by a space; the whole design timing will be reported by default if this parameter is not specified.

#### **Examples and Explanation**

# report setup timing with the maximum timing path of 100:

report\_timing -setup -max\_paths 100 -max\_common\_paths 5

#### **B.5.2 report\_high\_fanout\_nets**

#### Syntax

Command: report\_high\_fanout\_nets Parameter: [-clock regions]

[-slr]

[-ascending]

[-max\_nets <max\_net\_value>] [-min\_fanout <min\_fanout\_value>] [-max\_fanout <max\_fanout\_value>]

#### Note!

- -clock\_regions: Optional, report the clock net only.
- -slr: Optional, report the set/reset (synchronous or asynchronous) net only.
- -ascending: Optional, specify the report nets fanout arranging in descending order; if this parameter is not specified, the report nets fanout arranges in ascending order by default.
- -max\_net: Optional, specify the max net number to report. The value should be an integer greater than zero. Its default value is 10.
- -min\_fanout: Optional, report a net that has a fanout greater than or equal to the specified number. Its value must be an integer greater than zero.
- -max\_fanout: Optional, report a net that has a fanout less than or equal to the specified number. Its value must be an integer greater than zero.

#### **Examples and Explanation**

#Report nets that have fanout between 1 and 15, report 10 nets at most:

report\_high\_fanout\_Nets -slr -max\_nets 10 -min\_fanout 1 -max\_fanout 15

#Report the top 10 fanout nets: report\_high\_fanout\_Nets -max\_nets 10.

#### **B.5.3 report\_route\_congestion**

#### Syntax

Command: report\_route\_congestion

Parameter: [-max\_grids <max grids value>]

[-min\_route\_congestion <min route congestion value>]

[-max\_route\_congestion <max route congestion>]

[-LOC <position>]

- -max\_grids: Optional, specify the maximum number of grids to report, its default value is 10. Its value must be an integer greater than zero, or the parameter will be ignored.
- -min\_route\_congestion: Optional, specify the minimum route congestion of grid to report, its default value is 0. Its value must be a float number between 0 and 1.

- -max\_route\_congestion: Optional, specify the maximum route congestion of grid to report, its default value is 1. Its value must be a float number between min\_route\_congestion value and 1, or the parameter will be ignored. The default value 1 will be used. Its value should not be less than the parameter value min\_route\_congestion, or the report warning information will be ignored.
- -LOC: Optional, specify the physical location of grids to report. Its value could be a single location, such as R1C3, which means the first row and the third column of the grid. Its value could also be a location range, such as R1C[1:3], which means column 1~3, row 3; R[1:3]C1, which means column 1, row 1~3; or R[1:3]C[1:3], which means column 1~3, row 1~3.

#### **Examples and Explanation**

#report route congestion of grids locating on row 1 to 5, column 1 to 5 whose route congestion is between 0 and 0.5:

report\_route\_congestion -max\_grids 5 -min\_route\_congestion 0 -max\_route\_congestion 0.5 -LOC R[1:5]C[1:5]

#### B.5.4 report\_min\_pulse\_width

#### Syntax

Command: report\_min\_pulse\_width

Parameter: [-nworst <nworst value>]

[-min\_pulse\_width <min pulse width value>]

[-max\_pulse\_width <max pulse width value>]

[-detail]

[get\_regs {regIns name}]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- -nworst: Specify the maximum worst path number of the clock path to report.
- -min\_pulse\_width: Specify the minimum pulse width of the clock path to report. Its value must be a float greater than zero.
- -max\_pulse\_width: Specify the maximum pulse width of the clock path to report. Its
  value must be a float greater than zero.
- -detail: Specify the report format. The report will be detailed if this parameter is specified. Otherwise, the report will be brief.

• get\_regs {regIns name}: Specify reg. One or more regs can be specified. All regs pulse width timing analysis will be reported by default.

#### **Examples and Explanation**

#report the worst 3 clock paths that have pulse width between 0.1 and 4 in detail:

report\_min\_pulse\_width -nworst 3 -min\_pulse\_width 0.1 -max\_pulse\_width 4 -detail

#report the worst 20 clock paths that have pulse width between 0.001 and 4 in brief:

report\_min\_pulse\_width -nworst 20 -min\_pulse\_width 0.001 -max\_pulse\_width 4

## **B.5.5 report\_max\_frequency**

#### Syntax

Command: report\_max\_frequency

Parameter: -mod\_ins {mod\_ins1 mod\_ins2 ...}

#### Note!

-mod\_ins {mod\_ins1 mod\_ins2 ...}: Specify multiple module instances, separated by a space; The whole design maximum frequency will be reported by default regardless of whether this parameter is specified or not.

#### **B.5.6 report\_exceptions**

#### Syntax

Command: report\_exceptions

Parameter: -setup|-hold | -recovery | removal

[-max\_paths<number>]

[-max\_common\_paths< number >]

[-max\_logic\_level <number>]

[-min\_logic\_level <number>]

[-rise\_from <rise\_from\_list>]

[-fall\_from <fall\_from\_list>]

[-to <to list>]

[-rise\_to <rise\_to\_list>] [-fall\_to <fall\_to\_list>] [-through <through list>] [-rise\_through <rise\_through\_list>] [-fall\_through <fall\_through\_list>] [-from\_clock<from clock>] [-fall\_from\_clock<from clock>] [-rise\_from\_clock<from clock>] [-to\_clock<to clock>] [-rise\_to\_clock<to clock>] [-fall\_to\_clock<to clock>]

### Note!

The key parameters' names and meanings are the same as those of report\_timing.

## **Examples and Explanation**

set\_input\_delay -clock sysclk 1 all\_inputs

set\_output\_delay -clock virtual\_clock 1 all\_outputs

set\_max\_delay -from [get\_clocks {sysclk}] 5 -to [get\_ports{out\*}]

set\_min\_delay -from [get\_clocks{sysclk}] 3 -to [get\_ports {out\*}]

set\_multicycle\_path -end -setup -from [get\_clocks {sysclk}] -to
[get\_clocks {sysclk}] 2

set\_multicycle\_path -end -hold -from [get\_clocks {sysclk}] -to
[get\_clocks {sysclk}] 2

report\_exceptions -setup

Command: create\_clock Parameter: -period <period\_value> [-name <clock\_name>] [-waveform <edge\_list>] <source\_objects> [-add]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-period**: Specify the period of the clock. The parameter value should be greater than 0, and the period unit is ns.

**-name**: Specify the clock name. The clock name must be unique. If the new clock has the same name as an existing clock, the existing clock will be overwritten with the new clock. If the name is not specified, the name of the first source object will be regarded as the clock name.

**-waveform**: Specify the time of the rising edge and falling edge of the clock. The difference time between the rising edge and falling edge should be less than one period. In general, if the rising edge arrives first, both the rising edge time and the falling edge time should be less than one period. For example, "{0 5}" means the clock rising edge arrives at 0 ns, and the clock falling edge arrives at 5 ns; if the clock falling edge arrives, set the clock rising edge time to less than one period, and the falling edge time to equal to or greater than one period. If the period is set to 10 ns, "-waveform {5 10}" means the clock falling edge arrives at 5 ns.

**-add**: Use -add option to add multiple clocks to the same source object, or the new clock with different clock name on the same source object will be ignored when the source object already has one created clock.

-source\_objects: Specify the actual source objects which the clock arrives at, such as PORT, PIN, NET, etc. When the source object already has one created clock, the new clock with different clock name on the same source object will be not ignored if –add option is specified. Users can add multiple clocks to the same source object using the –add option. If the source object is not specified when you create the clock, the new clock will be ignored. Examples

# Create a clock that has a 10 ns, and the falling edge arrives first:

create\_clock -name clk -period 10.000 -waveform {5 10} [get\_ports {clk}]

# Create a clock with the duty cycle of 40%:

create\_clock -name clk -period 10.000 -waveform {6 10} [get\_ports {clk}]

# Create two clocks to one input port:

- 1. create\_clock -period 10 -name clk # command is ignored and no clk can be created
- create\_clock -period 10 -name clk [get\_ports {clk}] # Create clk successfully
- 3. create\_clock -period 10 -name clk1 [get\_ports {clk}] # commands is ignored and no clk can be created because -add is not used.

create\_clock -period 20 -name clk1 -add [get\_ports {clk}]

# Create clk1 successfully

# B.1.2 create\_generated\_clock

# Syntax

Command: create\_generated\_clock

Parameter: [-name <clock name>]

-source <master pin>

[-edges <edge list>]

[-edge\_shift <shift list>]

[-divide\_by <factor>]

[-multiply\_by<factor>]

[-duty\_cycle <percent>]

[-add]

[-invert]

[-master\_clock <clock>]

[-phase <phase>]

[-offset <offset>]

### <source objects>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-name**: Specify the name of the generated clock. If –name is not specified, the name of the first source object will be regarded as the clock name. The clock name must be unique. If the new clock has the same name as the existing clock, the existing clock will be overwritten with the new clock.

**-source**: Specify the source in which the generated clock is located; if there is more than one clock on this object, the -master\_clock option must be used to specify the original clock.

-master\_clock: Specify the master clock of the generated clock.

**-edges**: Specify the edge list of the generated clock; this option specifies a three positive ascending order integer parameter list, which indicates the relationship between the first rising edge of the generated clock, the first falling edge of the generated clock, the second rising edge of the generated clock, and the master clock edge. For instance, we mark the first rising edge of master clock as 1; the next falling edge is 2, the next rising edge is 3 ..., the marker numbers of the master clock edges are elements of edge list. We can create a two divider generated clock with "-edges {1 3 5}".

**-edge\_shift**: Specify the edge shift of each edge. This should be used together with the "-edges" option. It can be specified as any number, but the edge cannot go beyond the adjacent border.

### Note!

"-edge" and "-edge\_shift" cannot be used together with the other parameters used for waveform adjustment, except "-invert".

-divide\_by: Specify the divider value; the value should be a positive integer.

**-multiply\_by**: Specify the multiplier value; the value should be a positive integer.

**-duty\_cycle**: Specify the duty cycle of generated clock; the value should be a positive integer lower than 100.

-add: When this option is specified, this clock can coexist with an

existing clock.

-invert: Specify to invert the waveform of the generated clock.

-phase: Specify the phase shift of the clock edges in degrees.

-offset: Specify the offset of the clock edges in time values.

**-source object**: Specify the actual source objects that the generated clock arrives at, such as PORT, PIN, NET, etc.

# **Examples and Explanation**

# Create a 2\*divider generated clock to port a by using "-divide\_by":

create\_clock -period 10 [get\_ports clk]

create\_generated\_clock -name genClk -source [get\_ports {clk}] -divide\_by 2 [get\_ports {a}]

# Create a 2\*divider generated clock to port a by using "-edges":

create\_generated\_clock -name genClk -source [get\_ports {clk}] -edges {1 3 5} [get\_ports {a}]

# Create a 2\*multiplier clock with a 40% duty cycle:

create\_generated\_clock -name genClk0 -source [get\_ports {clk}] -multiply\_by 2 -duty\_cycle 40 [get\_pins {pll\_out}]

# Create an inverted clock 2\*divider relative to the output of the source clock:

create\_generated\_clock -name genClk1 -source [get\_ports {clk}]
-divide\_by 2 -invert [get\_pins {pll\_out}]

# Create a clock 2\*multiplier with a 90-degree phase shift:

create\_generated\_clock -name genClk2 -source [get\_ports{clk}] -multiply\_by 2 -phase 90[get\_pins {pll\_out}]

# Create a 2\*divider generated clock:

create\_generated\_clock -name genClk3 -source [get\_ports {clk}] -edges {2 4 6}[get\_pins {pll\_out}]

# Create two clocks to an input port that are switched externally:

create\_clock -period 10 -name clk [get\_ports {clk}]

create\_clock -period 20 -name clk1 -add [get\_ports {clk}]

create\_generated\_clock -name genClk -source [get\_ports {clk}] -divide\_by 2 -master\_clock clk -add [get\_pins {pll\_out}]

create\_generated\_clock -name genClk1 -source [get\_ports {clk}]

-master\_clock clk1 -divide\_by 2 -add [get\_pins {pll\_out}]

B.1.3 set\_clock\_latency

### Syntax

Command: Set\_clock\_latency

Parameter: -source [-rise | -fall]

[-late | -early]

<delay>

[-clock <clock list>]

<object list>

### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-source: Specify the clock latency type of source.

**-rise** | **-fall**: Specify the rising | falling clock latency. -rise| -fall cannot be specified in one statement. If both are not specified, the clock latency is applied to all conditions.

## Note!

User needs to specify the source latency value. The default value for Gowin YunYuan software is 0 ns.

-late | -early: Specify the late | early clock latency; -late is used for regular setup analysis, and -early is used for regular hold analysis.

<delay>: Specify the clock latency value.

## Note!

The default setting provided by STA is 0 ns.

Users can specify the clock that the source latency affects when more than one clock is on one source object. If this option is used, all clocks have the same delay.

<source objects>: Specify source objects that the clocks arrive at.

## **Examples and Explanation**

create\_clock -period 10 -name clk [get\_ports {clk}]

create\_clock -period 10 -name clk0 [get\_ports {clk}] -add

# Specify 2 ns clock latency for clk:

set\_clock\_latency -source 2 [get\_clocks {clk}]

# Specify 2 ns clock latency for clk0:

set\_clock\_latency -source 2 -clock [get\_clocks {clk0}] [get\_ports {clk}]

# **B.1.4 set\_clock\_uncertainty**

Syntax

Command: Set\_clock\_uncertainty

Parameter: [-from <from clock>]

[-rise\_from <rise from clock>]

[-fall\_from <-fall from clock>]

[-to <to clock>]

[-rise\_to <rise to clock>]

[-fall\_to <fall to clock>]

[-setup | -hold]

<uncertainty value>

### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-from/-rise\_from/-fall\_from**: Specify from the clock list. Specify the clock edge with "-rise\_from" and "-fall\_from".

**-from/-rise\_from/-fall\_from**: Specify to the clock list. Specify the clock edge with "-rise\_to" and "-fall\_to".

-setup/-hold: Specify that clock uncertainty affects setup or hold analysis; If both "-setup" and "-hold" are not specified, this uncertainty value is applied to both analysis type.

<uncertainty value>: Specify clock uncertainty value.

### Note!

The default setting value provided by STA is 0.02 ns.

# **Examples and Explanation**

# Set the clock uncertainty setup time from clk to clk to 0.5:

set\_clock\_uncertainty -setup -from clk -to clk 0.5

# Set the clock uncertainty hold time from clk0 to clk to 0.0:

set\_clock\_uncertainty -hold -from clk0 -to clk 0.0

# B.1.5 set\_clock\_groups

### Syntax

Command: Set\_clock\_groups

Parameter: [-asynchronous | -Exclusive]

-group <clock name>

-group <clock name>

[-group <clock name>] ...

### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-asynchronous | -Exclusive: Specify the created clock groups relationship as exclusive;

-group: Create clock group;

## **Examples and Explanation**

# Set the relationship between clk and clk0 as exclusive:

set\_clock\_groups -Exclusive -group [get\_clocks {clk}] -group
[get\_clocks {clk0}]

# **B.2 I/O Constraints**

# B.2.1 set\_input\_delay

## Syntax

Command: Set\_input\_delay

Parameter: -clock clock\_name

[-clock\_fall] [-rise] [-fall] [-max] [-min] [-add\_delay] [-source\_latency\_included] <delay\_value> <port\_list>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-clock: Specify the clock name.

-clock\_fall: Specify that the input delay is relative to falling clock edge.

#### Note!

If the option is not specified, the input delay is relative to the rising clock edge by default.

-rise/-fall: Specify rise | fall input delay. If only one of them is specified, the specified input delay is applied to the other.

-max/-min: Specify max | min input delay. If only one of them is specified, the specified input delay is applied to the other.

-add\_delay: When this option is specified, constraints that have already been input will not be overwritten.

-source\_latency\_included: Specify that the input delay has already included the source latency.

#### Note!

If this option is specified, the source latency is added to the input delay.

<delay\_value>: Specify input delay value.

#### Note!

The default STA input delay value is 0 ns.

ort\_list>: Specify port list for this constraint.

# **Examples and Explanation**

# Set the input delay based on clk rising edge for port a as 0.8 ns:

set\_input\_delay -clock clk 0.8 [get\_ports {a}]

# Set the input delay based on clk rising edge for all input ports as 0.8 ns:

set\_input\_delay -clock clk 0.8 [all\_inputs]
# Set the input delay based on clk falling edge for port a as 0.8 ns:
set\_input\_delay -clock clk -clock\_fall 0.8 [get\_ports {a}]
# Create Input delays for different min/max and rise/fall combinations:
set\_input\_delay -clock clk -max -rise 1.4 [get\_ports {a}]
set\_input\_delay -clock clk -max -fall 1.5 [get\_ports {a}]
set\_input\_delay -clock clk -min -rise 0.7 [get\_ports {a}]
# Create several input delays related with more than one clock:
set\_input\_delay -clock clk0 -min 1.2 [get\_ports {a}]
set\_input\_delay -clock clk0 -min 1.2 [get\_ports {a}]
set\_input\_delay -clock clk0 -min 2.1 -add\_delay [get\_ports {a}]
set\_input\_delay -clock clk1 -min 2.1 -add\_delay [get\_ports {a}]

# B.2.2 set\_output\_delay

# Syntax

Command: Set\_output\_delay Parameter: -clock clock\_name [-clock\_fall] [-rise] [-fall] [-max] [-min] [-add\_delay] [-source\_latency\_included] <delay\_value>

<port\_list>

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

-clock: Specify clock name related with output delay.

-clock\_fall: Specify the clock reference edge of output delay.

### Note!

Rising edge is the default reference edge.

**-rise/-fall**: Specify rise | fall input delay. If only one of them is specified, the specified input delay is applied to the other.

**-max/-min**: Specify max | min input delay. If only one of them is specified, the specified input delay is applied to the other.

**-add\_delay**: When this option is specified, constraints that have already been input will not be overwritten.

-source\_latency\_included: Specify that the input delay has already included source latency.

<delay\_value>: Specify the output delay value.

## Note!

The default STA output delay value is 0ns.

ort\_list>: Specify port list for this constraint;

## **Examples and Explanation**

# Set the output delay of port b as 0.5 ns:

set\_output\_delay -clock clk 0.5 [get\_ports {b}]

# Set the output delay of all ports as 0.5 ns:

set\_output\_delay -clock clk 0.5 [all\_outputs]

# Set the output delay based on falling edge for all ports as 0.5 ns:

set\_output\_delay -clock clk -clock\_fall 0.5 [get\_ports {b}]

# Set the output delay based on rising edge for all port b:

set\_output\_delay -clock clk -max -rise 0.3 [get\_ports {b}]

set\_output\_delay -clock clk -max -fall 0.5 [get\_ports {b}]
set\_output\_delay -clock clk -min -rise 0.8 [get\_ports {b}]
set\_output\_delay -clock clk -min -fall 0.7 [get\_ports {b}]
# Create several input delays related with more than one clocks:
set\_output\_delay -clock clk0 -min 0.5 [get\_ports {b}]
set\_output\_delay -clock clk0 -max 0.6 [get\_ports {b}]
set\_output\_delay -clock clk0 -clock\_fall 0.7 -add\_delay [get\_ports {b}]
set\_output\_delay -clock clk1 -min 0.8 -add\_delay [get\_ports {b}]

# **B.3 Path Constraints**

# B.3.1 set\_max\_delay | set\_min\_delay

Syntax

Command: Set max delay Parameter: [-from <from list>] [-rise from <rise from list>] [-fall\_from <fall\_from\_list>] [-to <to list>] [-rise to <rise to list>] [-fall\_to <fall\_to\_list>] [-through <through list>] <delay value> Command: Set min delay Parameter: [-from <from list>] [-rise from <rise from list>] [-fall\_from <fall\_from\_list>] [-to <to list>] [-rise to <rise to list>] [-fall to <fall to list>] [-through <through list>]

### <delay value>

### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-from**: Specify for the beginning point; "-from" objects can be PORTS, NETS, REGS, CLOCKS, PINS, etc.

-to: Specify for the end point. "-to" objects can be PORTS, NETS, REGS, CLOCKS, PINS, etc.

**-through**: Specify through objects list; through objects can be nets, pins, etc. Only one "-through" can be used in one statement at one time.

### Note!

The three parameters outlined can be used in combination or alone. If the specified objects of the three parameters are not on the same path, STA will ignore this constraint, and timing analysis will not be affected.

# **Examples and Explanation**

# Set 5 ns max delay between two clocks set max delay -from [get clocks {clk}] -to [get clocks {clk}] 5 # Set 2 ns max delay from port a to register 0 set\_max\_delay -from [get\_ports {a}] -to [get\_registers {reg0}] 2 # Set 2 ns max delay from reg0 to port b set max delay -from [get registers {reg0}] -to [get ports {b}] 2 # Set 5 ns max delay for all timing objects driven by clocks set\_max\_delay -from [all\_clocks] 5 -to [get\_ports {out\*}] # Set 2 ns max delay from port a to port b set max delay -from [get ports {a}] -to [get ports {b}] 2 #set 2 ns max delay rise from regs to fall clock set\_max\_delay -rise\_from [get\_regs {reg0}] -fall\_to [get\_clocks {clk}] 2 # Set 5 ns min delay between two clocks set min delay -from [get clocks {clk}] -to [get clocks {clk}] 0.5 # Set 0.5 ns min delay from port a to register 0 set min delay -from [get ports {a}] -to [get registers {reg0}] 0.5

# Set 0.5 ns min delay from reg0 to port b
set\_min\_delay -from [get\_registers {reg0}] -to [get\_ports {b}] 0.5
# Set 0.5 ns min delay from port a to port b
set\_min\_delay -from [get\_ports {a}] -to [get\_ports {b}] 0.5
# Set 0.5 ns min delay from input port only to falling clock
set\_max\_delay -from [get\_ports {a}] -fall\_to [get\_clocks {clk}] 0.5

# B.3.2 set\_false\_path

### Syntax

Command: Set\_false\_path Parameter: [-from <from list>] [-to <to list>] [-through <through list>] [-setup] [-hold]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-setup/-hold**: Specify constraints for setup time or hold time. The two parameters are mutually exclusive and have an effect on setup time.

**-from**: Specify for the beginning point; users can specify them through "get\_ports", "get\_regs", or "get\_clocks".

**-to**: Specify for the end point; users can specify them through "get\_ports", "get\_regs", or "get\_clocks".

**-through:** Specify through objects list; through objects can be nets, pins; multiple pins and nets can be specified. Only one "-through" parameter can be used in the same statement.

### Note!

"-from ", "-to ", and "-through" can be used in combination or alone. If the specified objects of the three parameters are not on the same path, STA will ignore this constraint, and timing analysis will not be affected.

# **Examples and Explanation**

# Set false path between two unrelated clocks: set\_false\_path -from [get\_clocks {clk0}] -to [get\_clocks {clk1}] # Set false-path between two regs: set\_false\_path -from [get\_regs {reg0}] -to [get\_regs {reg1}] # Set false path rising from regs to falling clock: set\_false\_path –from [get\_clocks {clk}] -to [get\_clocks {clk1}] # Set false path from port a to port b: set\_false\_path –from [get\_ports {a}] to [get\_ports {b}]

# B.3.3 set\_multicycle\_path

### Syntax

Command: Set\_multicycle\_path Parameter: [-setup|-hold] [-start|-end] [-from <from\_list>] [-to <to list>] [-through <through\_list>] <path multiplier>

### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.

**-start/-end**: Specify that the constraint clock is launch clock or latch clock.

### Note!

The reference clock of STA is latch clock by default.

**-setup/-hold**: Specify constraints for setup time or hold time. The two parameters are mutually exclusive.

Note!

STA has an effect on setup time by default.

**-from**: Specify for the beginning point; users can specify them through "get\_ports", "get\_regs", or "get\_clocks".

**-to**: Specify for the end point; users can specify them through "get\_ports", "get\_regs", or "get\_clocks".

**-through**: Specify through objects list; through objects can be nets, pins; multiple pins and nets can be specified. Only one "-through" parameter can be used in the same statement.

### Note!

"-from ", "-to ", and "-through" can be used in combination or alone. If the specified objects of the three parameters are not on the same path, STA will ignore this constraint, and timing analysis will not be affected.

## **Examples and Explanation**

create\_clock -name clk -period 10 [get\_ports {clk}]

create\_generated\_clock -name genClk -multiply\_by 2 -source
[get\_ports {clk}] [get\_pins {pll\_out}]

# Set end setup multicycle path of 2 with the reference clock genClk;

set\_multicycle\_path -end -setup -from [get\_clocks {clk}] -to [get\_clocks
{genClk}] 2

# Set end setup multicycle path of 2 with the reference clock reg0;

set\_multicycle\_path -start -setup -from [get\_regs {reg0}] -to [get\_regs
{reg1}] 3

set\_multicycle\_path -start -hold -from [get\_regs {reg0}] -to [get\_regs
{reg1}] 1

# Set multicycle constraint of 3 rising from a clock and falling to an object

set\_multicycle\_path -end -setup -from [get\_clocks {clk}] -to [get\_clocks
{clk0}] 3

# **B.4 Operation Conditions Constraints**

## Syntax

Command: Set\_operation\_conditions

Parameter: [-grade <c|i>]

[-model <slow|fast>]

[-speed <speed>] [-setup] [-hold] [-max] [-min] [-max min]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects;
- -grade: Specify the device temperature grade, supports commercial and industrial temperatures; it defaults to black;
- -model: Specify the process corner of timing analysis; it defaults to slow;
- -speed: Specify the device speed grade;
- -setup: Setup time check under current process corner; same function with -max;
- -hold: Setup time check under current process corner; same function with -min;
- -max: Setup time check under current process corner; same function with -setup;
- -min: Setup time check under current process corner; same function with -hold;
- -max\_min: Setup time and hold time check under current process corner; same function with -setup and -hold.

# **B.5 Timing Report**

# **B.5.1 report\_timing**

### Syntax

Command: report\_timing

Parameter:[-setup|-hold|-recovery|-removal]

[-max\_paths <value>]

[-max\_common\_paths < value >]

[-rise\_from <rise\_from\_list>]

[-fall\_from <fall\_from\_list>]

[-to <to list>]

[-rise\_to <rise\_to\_list>]

[-fall\_to <fall\_to\_list>] [-through <through list>] [-from\_clock<from clok>] [-fall\_from\_clock <from clok>] [-rise\_from\_clock <from clok>] [-to\_clock <to clok>] [-rise\_to\_clock <to clok>] [-fall\_to\_clock <to clok>] [-fall\_to\_clock <to clok>] [-min\_logic\_level] [-max\_logic\_level] [-mod\_ins {mod\_ins1 mod\_ins2 ...}]

#### Note!

- Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.
- Supports the SDC constraint type.
- -setup|-hold|-recovery|-removal: Specify report setup or hold timing path, recovery or removal timing path.
- -max\_paths: Specify the max path number to report.
- -max\_common\_paths: Specify the max path number of each end node to report.
- -rise\_from/-fall\_from: Specify from objects list.
- -to /-rise\_to /-fall\_to: Specify to objects list.
- -through: Specify through objects list;
- -from\_clock /-fall\_from\_clock /-rise\_from\_clock /-to\_ clock /-rise\_to\_ clock
   /-fall\_to\_clock: Specify from clock and to clock;
- -min\_logic\_level/-max\_logic\_levell: Specify the minimum / maximum logic level of the reported timing paths.
- -mod\_ins {mod\_ins1 mod\_ins2 ...}: Specify multiple module instance, separated by a space; The whole design timing will be reported by default if this parameter is not specified.

### **Examples and Explanation**

# report setup timing with the maximum timing path of 100: report\_timing -setup -max\_paths 100 -max\_common\_paths 5

# B.5.2 report\_high\_fanout\_nets

# Syntax

Command: report\_high\_fanout\_nets Parameter: [-clock\_regions] [-slr] [-ascending] [-max\_nets <max\_net\_value>] [-min\_fanout <min\_fanout\_value>] [-max\_fanout <max\_fanout\_value>]

### Note!

- -clock\_regions: Optional, report the clock net only.
- -slr: Optional, report the set/reset (synchronous or asynchronous) net only.
- -ascending: Optional, specify the report nets fanout arranging in descending order. If this parameter is not specified, the report nets fanout arranges in ascending order by default.
- -max\_net: Optional, specify the max net number to report. The value should be an integer greater than zero. Its default value is 10.
- -min\_fanout: Optional, report the net that has fanout greater than or equal to the specified number. Its value must be an integer greater than zero.
- -max\_fanout: Optional, report the net that has fanout less than or equal to the specified number. Its value must be an integer greater than zero.

## **Examples and Explanation**

#Report nets that have fanout between 1 and 15, report 10 nets at most:

report\_high\_fanout\_Nets -slr -max\_nets 10 -min\_fanout 1 -max\_fanout 15

#Report the top 10 fanout nets:

report\_high\_fanout\_Nets -max\_nets 10

# **B.5.3 report\_route\_congestion**

## Syntax

Command: report\_route\_congestion

Parameter: [-max\_grids <max grids value>]

[-min\_route\_congestion <min route congestion value>]

[-max\_route\_congestion <max route congestion>]

[-LOC <position>]

- -max\_grids: Optional, specify the maximum number of grids to report, the default value is 10. The value must be an integer greater than zero, or the parameter will be ignored.
- -min\_route\_congestion: Optional, specify the minimum route congestion of grid to report. The default value is 0. The value must be a float number between 0 and 1.
- -max\_route\_congestion: Optional, specify the maximum route congestion of grid to report. The default value is 1. The value must be a float number between min\_route\_congestion value and 1, or the parameter will be ignored. The default value 1 will be used. The value should not be less than parameter value min\_route\_congestion, or the report warning information will be ignored.
- -LOC: Optional, specify the physical location of grids to report. The value could be a single location, such as R1C3, which means the first row and the third column of the grid. Its value could also be a location range, such as R1C[1:3], which means column 1~3, row 3; R[1:3]C1, which means column 1, row 1~3; or R[1:3]C[1:3], which means column 1~3, row 1~3.

# **Examples and Explanation**

#report route congestion of grids locating on row 1 to 5, column 1 to 5 for which the route congestion is between 0 and 0.5:

report\_route\_congestion -max\_grids 5 -min\_route\_congestion 0 -max\_route\_congestion 0.5 -LOC R[1:5]C[1:5]

# B.5.4 report\_min\_pulse\_width

## Syntax

Command: report\_min\_pulse\_width

Parameter: [-nworst <nworst value>]

[-min\_pulse\_width <min pulse width value>]

[-max\_pulse\_width <max pulse width value>]

[-detail]

[get\_regs {regIns name}]

### Note!

• Parameters within "[]" are optional. The more options you use, the more detailed the constraints are. If no options are used, the generic constraint applies to more objects.

- -nworst: Specify the maximum worst path number of the clock path to report.
- -min\_pulse\_width: Specify the minimum pulse width of the clock path to report. Its
  value must be a float greater than zero;
- -max\_pulse\_width: Specify the maximum pulse width of the clock path to report. Its
  value must be a float greater than zero;
- -detail: Specify the report format. The report will be detailed if this parameter is specified. Otherwise, the report will be brief.
- get\_regs {regIns name}: Specify reg, one or more regs can be specified. All regs
  pulse width timing analysis will be reported by default.

## **Examples and Explanation**

#report the worst 3 clock paths that have pulse width between 0.1 and 4 in detail:

report\_min\_pulse\_width -nworst 3 -min\_pulse\_width 0.1 -max\_pulse\_width 4 -detail

#report the worst 20 clock paths that have pulse width between 0.001 and 4 in brief:

report\_min\_pulse\_width -nworst 20 -min\_pulse\_width 0.001 -max\_pulse\_width 4

# B.5.5 report\_max\_frequency

## Syntax

Command: report\_max\_frequency

Parameter: -mod\_ins {mod\_ins1 mod\_ins2 ...}

### Note!

-mod\_ins {mod\_ins1 mod\_ins2 ...}: Specify multiple module instances, separated by a space; The whole design maximum frequency will be reported by default no matter this parameter is specified or not.

# **B.5.6 report\_exceptions**

Syntax

Command: report\_exceptions

Parameter: -setup|-hold | -recovery | removal

[-max\_paths<number>]

[-max\_common\_paths< number >] [-max\_logic\_level <number>] [-min\_logic\_level <number>] [-rise\_from <rise\_from\_list>] [-fall\_from <fall\_from\_list>] [-to <to list>] [-rise\_to <rise\_to\_list>] [-fall\_to <fall\_to\_list>] [-through <through list>] [-rise\_through <rise\_through\_list>] [-fall\_through <fall\_through\_list>] [-from\_clock<from clock>] [-fall\_from\_clock<from clock>] [-rise\_from\_clock<from clock>] [-to\_clock<to clock>] [-rise\_to\_clock<to clock>] [-fall\_to\_clock<to clock>]

## Note!

The names and meanings of the key parameters are the same as those of report\_timing.

## **Examples and Explanation**

set\_input\_delay -clock sysclk 1 all\_inputs

set\_output\_delay -clock virtual\_clock 1 all\_outputs

set\_max\_delay -from [get\_clocks {sysclk}] 5 -to [get\_ports{out\*}]

set\_min\_delay -from [get\_clocks{sysclk}] 3 -to [get\_ports {out\*}]

set\_multicycle\_path -end -setup -from [get\_clocks {sysclk}] -to
[get\_clocks {sysclk}] 2

set\_multicycle\_path -end -hold -from [get\_clocks {sysclk}] -to
[get\_clocks {sysclk}] 2

report\_exceptions -setup

