



深圳市紫光同创电子有限公司  
SHENZHEN PANGO MICROSYSTEMS CO.,LTD

# Pango Design Suite 用户手册

(Version 1.5)

深圳市紫光同创电子有限公司

版权所有 侵权必究

## 文档版本修订记录

版本号	发布日期	修订记录
V1.0	2022.7.12	初始版本
V1.1	2022.8.19	更新图片
V1.2	2022.10.25	更新PPC参数配置相关描述
V1.3	2022.11.16	补充时钟降级相关描述
V1.4	2022.11.18	更新General相关信息：增加特征控制数据选项
V1.5	2022.11.18	更新Place&Route配置选项相关说明信息

## 目录

1	关于手册 .....	21
1.1	软件简介 .....	21
1.2	更多帮助 .....	21
2	基本操作 .....	23
2.1	启动软件 .....	23
2.2	新建工程 .....	23
2.3	主体界面 .....	29
2.4	菜单介绍 .....	31
2.5	工具介绍 .....	41
2.6	打开工程 .....	42
2.7	工程管理 .....	47
2.8	设置器件 .....	52
2.9	添加文件 .....	53
2.9.1	添加源文件 .....	53
2.9.2	新建源文件 .....	60
2.9.3	刷新 design hierarchy .....	65
2.9.4	Global include 设置 .....	66
2.9.5	添加约束文件 .....	67
2.10	移除文件 .....	78
2.11	参数设置 .....	80
2.12	报告系统 .....	81
2.13	文本编辑工具 .....	83
2.14	MESSAGES 管理 .....	87
3	开发流程 .....	88
3.1	COMPILE .....	88
3.2	SYNTHESIZE .....	90

3.2.1	ADS .....	90
3.2.2	Synplify_Pro.....	94
3.3	DEVICE MAP.....	104
3.4	PLACE & ROUTE.....	107
3.4.1	Place & Route 设置.....	107
3.4.2	Place & Route 报告.....	119
3.5	REPORT TIMING .....	126
3.5.1	Report Timing 设置.....	126
3.5.2	Report Timing 报告.....	129
3.6	REPORT POWER .....	142
3.6.1	Report power 设置 .....	142
3.6.2	Report Power 报告 .....	146
3.7	GENERATE NETLIST.....	147
3.8	DESIGN EDITOR .....	148
3.9	GENERATE BITSTREAM.....	148
3.10	命令行操作 .....	160
4	相关插件 .....	161
4.1	ADS .....	161
4.2	SYNPLIFY_PRO.....	161
4.3	USER CONSTRAINT EDITOR .....	161
4.4	IP COMPILER .....	161
4.5	PHYSICAL CONSTRAINT EDITOR .....	161
4.6	DESIGN EDITOR .....	161
4.7	FABRIC INSERTER .....	161
4.8	FABRIC DEBUGGER .....	162
4.9	FABRIC CONFIGURATION .....	162
4.10	FABRIC JTAGSERVER.....	162
4.11	PANGO POWER CALCULATOR.....	162

4.12	PANGO POWER PLANNER .....	162
4.13	TIMING ANALYZER .....	162
4.14	ROUTE CONSTRAINT EDITOR .....	162
4.15	PANGO SSN ESTIMATOR .....	162
4.16	PANGO SSN ANALYZER .....	162
4.17	SIMULATION .....	162
4.18	VIEW RTL SCHEMATIC .....	162
4.19	VIEW TECHNOLOGY SCHEMATIC .....	163
5	命令行运行 .....	164
5.1	工程运行方式 .....	164
5.2	TCL 脚本运行方式 .....	164
5.3	SHELL 运行方式 .....	165
5.4	PDS 支持的命令 .....	166
6	时序约束命令 .....	169
6.1	对象获取命令 .....	169
6.1.1	all_clocks .....	169
6.1.2	all_inputs .....	169
6.1.3	all_outputs .....	169
6.1.4	all_registers .....	170
6.1.5	get_cells .....	171
6.1.6	get_clocks .....	171
6.1.7	get_nets .....	171
6.1.8	get_pins .....	171
6.1.9	get_ports .....	172
6.2	时钟约束命令 .....	172
6.2.1	create_clock .....	172
6.2.2	create_generated_clock .....	174

6.2.3	set_clock_latency .....	179
6.2.4	set_clock_uncertainty .....	180
6.2.5	set_clock_group.....	182
6.2.6	set_external_delay .....	186
6.3	IO 时序约束命令 .....	187
6.3.1	set_input_delay.....	187
6.3.2	set_output_delay.....	193
6.4	ASSERTION 约束命令 .....	198
6.4.1	set_max_skew .....	198
6.5	EXCEPTION 约束命令 .....	200
6.5.1	set_max_delay .....	200
6.5.2	set_min_delay.....	202
6.5.3	set_multicycle_path.....	204
6.5.4	set_false_path .....	207
6.6	OTHER 约束命令 .....	209
6.6.1	set_disable_timing.....	209
6.7	报告命令 .....	210
6.7.1	report_timing .....	210
6.8	命令的语法格式 .....	212
6.8.1	tcl 的命令格式.....	212
6.8.2	List 的问题.....	212
7	局部动态重配 .....	214
7.1	启用动态重配 .....	214
7.1.1	操作演示 .....	214
7.1.2	界面说明 .....	214
7.2	创建 PARTITION DEFINITION .....	215
7.2.1	操作演示 .....	215
7.2.2	界面说明 .....	216

7.2.3	Partition Definition Instance 图标.....	216
7.2.4	Partition Definition.....	217
7.2.5	PD 的默认 RM.....	217
7.2.6	Multiple Flows .....	218
7.2.7	创建 PD 的限制.....	218
7.2.8	查看和编辑 PD 属性 .....	223
7.3	删除 PARTITION DEFINITION .....	224
7.4	添加 RECONFIGURE MODULE .....	226
7.4.1	操作演示 .....	226
7.4.2	查看和更改 RM 属性 .....	227
7.5	删除 RECONFIGURE MODULE .....	228
7.6	RM 增删 DESIGN 文件 .....	230
7.6.1	RM 添加 Design 文件.....	230
7.6.2	RM 的 Top Module .....	230
7.6.3	RM 删除 Design 文件.....	231
7.7	添加/删除 PD_INSTANCE .....	232
7.7.1	添加 PD_Instance.....	232
7.7.2	删除 PD_Instance.....	233
7.8	约束可重配区域 .....	234
7.9	RUN FLOWS .....	236
7.9.1	流程与 PD、PD Instance 及 RM 的对应关系.....	236
7.9.2	切换流程树 .....	237
7.9.3	流程依赖 .....	237
7.9.4	流程关系 .....	238
7.9.5	流程并行运行 .....	238
7.10	组件相关 .....	240
7.11	PROJECT SETTING .....	241
7.11.1	操作演示 .....	241

7.11.2 相关说明 .....	241
7.12 REPORT SUMMARY .....	242
7.12.1 操作演示 .....	242
7.13 MESSAGES .....	243
7.13.1 操作演示 .....	243
7.14 LOG .....	244
7.14.1 操作演示 .....	244
7.15 重配流程的工程文件 .....	245
7.16 位流生成与编程 .....	245
7.16.1 位流文件说明 .....	245
8 软件原则 .....	246
8.1 文件和文件夹命名规则 .....	246
8.2 PDS 套件中打开工具软件规则 .....	246
8.3 流程与工具软件的关系 .....	246
9 常用命令列表 .....	247
9.1 ADD_CONSTRAINT 指定约束文件 .....	247
9.2 ADD DESIGN 指定设计文件 .....	247
9.3 ADD_FIC 指定约束文件 .....	247
9.4 ADD_SIMULATION 指定仿真文件 .....	247
9.5 ADD_SOURCES 指定文件夹 .....	248
9.6 CLEAN 清除 ACTION 状态 .....	248
9.7 SET_ARCH 指定器件信息 .....	248
9.8 COMPILE 编译用户设计 .....	249
9.9 DEV_MAP 映射设计到架构 .....	249
9.10 GEN_BIT_STREAM 生成配置位流 .....	250
9.11 GEN_NETLIST 生成网表 .....	251
9.12 PNR 布局和布线设计 .....	251

9.13 REPORT_POWER 报告功耗信息.....	252
9.14 REPORT_TIMING 报告时序信息 .....	254
9.15 COMPILE_SIMLIB 编译仿真库 .....	254
10 常见错误 .....	255
10.1 CONSTRAINT ERROR .....	255
10.2 PHYSICAL CONSTRAINTEDITOR.....	255
10.3 USERCONSTRAINTEDITOR .....	257
10.4 FLOW ERROR .....	257
10.4.1 FabFlow.....	257
10.4.2 DB .....	259
10.4.3 UiWgt.....	260
10.5 DRC ERROR.....	261
10.6 MAP.....	263
10.7 PLACE.....	263
10.8 ROUTE ERROR.....	266
10.9 DESIGN EDITOR ERROR.....	267
10.10 TIMING ERROR.....	267
10.11 SIMULATION ERROR.....	272
免责声明 .....	273

## 图目录

图 2-1 初始界面.....	23
图 2-2 新建工程-菜单 .....	24
图 2-3 新建工程-快捷方式 .....	24
图 2-4 新建工程简介页面.....	25
图 2-5 新建工程向导.....	25
图 2-6 新建工程类型界面.....	26
图 2-7 新建工程添加 designs.....	27
图 2-8 新建工程添加 IP .....	27
图 2-9 新建工程添加约束.....	28
图 2-10 新建工程选择器件.....	28
图 2-11 新建工程 summary 界面 .....	29
图 2-12 建立工程后的 Pango Design Suite 主界面 .....	29
图 2-13 软件主体界面.....	30
图 2-14 File 菜单 .....	32
图 2-15 Import Project Settings.....	33
图 2-16 export project to tcl .....	34
图 2-17 Edit 菜单 .....	35
图 2-18 prefecences 设置 .....	36
图 2-19 UCE 网表解析配置 .....	36
图 2-20 Integrated Tools 配置 .....	36
图 2-21 license 设置 .....	37
图 2-22 人性化误操作配置 .....	37
图 2-23 color 配置表 .....	38
图 2-24 View 菜单 .....	38
图 2-25 Project 菜单 .....	39
图 2-26 Process 菜单 .....	39
图 2-27 Tools 菜单.....	40

图 2-28 Window 菜单 .....	41
图 2-29 Help 菜单 .....	41
图 2-30 Language Templates 界面 .....	42
图 2-31 Report Summary 界面 .....	42
图 2-32 Run 工具 .....	42
图 2-33 打开工程-菜单 .....	43
图 2-34 打开工程-快捷方式 .....	43
图 2-35 打开工程对话框 .....	44
图 2-36 打开工程后的 Pango Design Suite 主界面 .....	45
图 2-37 打开工程-最近工程-菜单 .....	45
图 2-38 打开工程-最近工程-向导栏 .....	46
图 2-39 device 信息不存在提示窗口 .....	46
图 2-40 选择 device 信息窗口 .....	47
图 2-41 Navigator 工程管理区 .....	48
图 2-42 source 窗口右键菜单 .....	48
图 2-43 Set file type 界面 .....	49
图 2-44 File 窗口工具栏 .....	50
图 2-45 Navigator 工程管理区 .....	50
图 2-46 File 右键菜单 .....	51
图 2-47 Navigator 右键菜单 .....	52
图 2-48 out of date 按钮变亮 .....	52
图 2-49 器件选择窗口 .....	53
图 2-50 Sources 窗口中单击工具栏 Add Source .....	53
图 2-51 Sources 窗口中右键菜单中 Add Source .....	54
图 2-52 Add Source 界面 .....	54
图 2-53 添加 design 界面 .....	55
图 2-54 添加设计文件后的软件界面 .....	55
图 2-55 添加文件失败对话框 .....	56

图 2-56 “Find In Files” 工具栏位置.....	56
图 2-57 “Find In Files” 工具界面 .....	57
图 2-58 “Find In Files” 工具可搜索范围.....	57
图 2-59 “Find In Files” 工具搜索结果.....	58
图 2-60 添加源文件右键菜单.....	58
图 2-61 添加 ip 文件对话框.....	58
图 2-62 添加 IP 后主界面 .....	59
图 2-63 更新 IP 界面 .....	59
图 2-64 Copy IP 界面 .....	60
图 2-65 添加源文件主界面.....	60
图 2-66 点击 Create File 后弹出界面 .....	61
图 2-67 输入文件名界面.....	61
图 2-68 输入文件名重名提示界面.....	62
图 2-69 Create File 结束后界面 .....	62
图 2-70 .v 文件语法框架设置界面.....	63
图 2-71 module 设置界面.....	63
图 2-72 .v 文件设置好语法框架后生成界面 .....	64
图 2-73 .vhdl 文件的语法框架设置界面 .....	64
图 2-74 .vhdl 文件设置完成后界面 .....	65
图 2-75 刷新 hierarchy .....	65
图 2-76 set Global include.....	67
图 2-77 添加约束文件.....	68
图 2-78 右键菜单添加源文件.....	68
图 2-79 添加源文件界面.....	69
图 2-80 选择添加约束界面.....	69
图 2-81 添加用户约束文件.....	70
图 2-82 Tools 菜单下 Physical Constraint Editor (Post-Map) .....	71
图 2-83 添加物理约束.....	72

图 2-84 添加物理约束后的 Physical Constraint Editor 界面.....	72
图 2-85 管脚约束视图.....	75
图 2-86 布局约束视图.....	77
图 2-87 布局约束视图.....	78
图 2-88 文件移除菜单.....	79
图 2-89 文件移除对话框.....	79
图 2-90 pcf 文件移除 .....	80
图 2-91 Configure 菜单 .....	80
图 2-92 工程管理区 configure 菜单 .....	81
图 2-93 选项配置窗口.....	81
图 2-94 报告 Report Summary .....	82
图 2-95“Project Location” 链接 .....	83
图 2-96 “Project Directory” 界面 .....	83
图 2-97 软件提供的默认文本编辑工具.....	84
图 2-98 指定文本编辑工具.....	86
图 2-99 Messages 窗口 .....	87
图 3-1 Compile 右键菜单 .....	89
图 3-2Compile 的选项设置 .....	89
图 3-3 Compile 结果界面 .....	90
图 3-4 Synthesize 右键菜单 .....	90
图 3-5 Synthesize 选项设置 .....	91
图 3-6 运行 Synthesize 结果界面 .....	91
图 3-7Synthesize 文本报告 .....	92
图 3-8 报告区右键菜单.....	92
图 3-9 Synthesize Report .....	93
图 3-10 表格右键菜单.....	93
图 3-11 设置表格行高界面 .....	94
图 3-12 Synthesize 右键菜单 .....	94

图 3-13 synthesize-device 选项卡配置选项界面 .....	95
图 3-14 synthesize-Timing Report 选项卡配置选项界面.....	96
图 3-15 synthesize-Options 选项卡配置选项界面 .....	97
图 3-16 synthesize-Constraints 选项卡配置选项界面 .....	98
图 3-17 synthesize-Implementation Results 选项卡配置选项界面 .....	99
图 3-18 synthesize-Verilog 选项卡配置选项界面 .....	100
图 3-19 synthesize-VHDL 选项卡配置选项界面.....	101
图 3-20 synthesize-GCC 选项卡配置选项界面.....	103
图 3-21 运行 Synthesize .....	103
图 3-22 运行 Synthesize 后软件界面 .....	104
图 3-23 Device Map 的选项设置 .....	105
图 3-24 Device Map 后软件界面 .....	106
图 3-25 Device Map Resource Usage Summary 界面.....	107
图 3-26 Place & Route 的选项设置 .....	108
图 3-27 Place & Route Multi-Run State 界面 .....	115
图 3-28 Place & Route 运行后软件界面 .....	115
图 3-29 Place & Route 右键界面 .....	116
图 3-30 Place & Route Select a Seed To Apply 界面 .....	116
图 3-31 Place & Route Resource Usage Summary 界面 .....	<b>错误!未定义书签。</b>
图 3-32 Place & Route IO Report 界面 .....	120
图 3-33 IO_REGISTER 状态 OREGIS .....	120
图 3-34 IO_REGISTER 状态 I/O/TREGIS .....	120
图 3-35 Clock Summary 界面显示 .....	121
图 3-36 Global Clock 界面显示 .....	121
图 3-37 PLL Clock 界面显示 .....	122
图 3-38 Report Timing 的选项设置 .....	127
图 3-39 Report Timing 的选项设置 .....	129
图 3-40 Timing Report 目录 .....	130

图 3-41 Report Timing 设置 .....	130
图 3-42 Check Timing Report Summary.....	131
图 3-43 Clock Summary .....	133
图 3-44 Fmax Summary .....	133
图 3-45 Clock Interaction Report.....	134
图 3-46 Clock Network Report .....	135
图 3-47 Setup Summary .....	135
图 3-48 Hold Summary .....	136
图 3-49 Recovery Summary.....	136
图 3-50 Removal Summary.....	136
图 3-51 Minimum Pulse Width Summary .....	137
图 3-52 Setup Path Summary Report .....	137
图 3-53 Detail timing Path Report.....	138
图 3-54 Minimum Pulse Width Path Report .....	138
图 3-55 时序是否违例显示.....	139
图 3-56 Input Ports Setup/Hold.....	139
图 3-57 Output ports Clock-to-out .....	139
图 3-58 Combination Delays.....	140
图 3-59 Setup/Hold times for input bus .....	140
图 3-60 Max/Min delays for output bus .....	141
图 3-61 Max Skew Report.....	141
图 3-62 Exception Report.....	142
图 3-63 Report power 的选项设置.....	142
图 3-64 Report power 的报告.....	146
图 3-65 Generate Netlist 的选项设置.....	147
图 3-66 Generate Bitstream 的选项设置.....	148
图 3-67 特征控制位的选项界面.....	154
图 3-68 运行 Generate Bitstream 后软件界面 .....	159

图 3-69 生成的位流文件.....	160
图 5-1 pds.Exe 打开 pds .....	164
图 5-2 脚本运行 tcl 文件.....	164
图 6-1 时钟创建实例.....	173
图 6-2 时钟创建波形图.....	174
图 6-3 generated clock 周期计算 .....	175
图 6-4 DCLKDIV2 波形.....	175
图 6-5 相移波形示例.....	176
图 6-6 -invert 实例.....	177
图 6-7 随路时钟示例.....	178
图 6-8 Source latency 示意图.....	179
图 6-9 set_clock uncertainty 示意图.....	182
图 6-10 Logically exclusive with no interaction 示例 .....	183
图 6-11 Logically exclusive with interaction 示例 .....	183
图 6-12 Physically exclusive 示例 .....	184
图 6-13 Asynchronous 示例.....	185
图 6-14 System Edge Rise 波形图.....	189
图 6-15 System Edge Fall 波形图 .....	189
图 6-16 System Edge Dual 波形图 .....	189
图 6-17 Source Center Rise 波形图 .....	190
图 6-18 Source Center Fall 波形图 .....	190
图 6-19 Source Center Dual 波形图 .....	190
图 6-20 Source Edge Direct Rise 波形图 .....	191
图 6-21 Source Edge Direct Fall 波形图 .....	191
图 6-22 Source Edge Direct Dual 波形图.....	191
图 6-23 Source Edge PLL Rise 波形图 .....	192
图 6-24 Source Edge PLL Fall 波形图 .....	192
图 6-25 Source Edge PLL Dual 波形图.....	192

图 6-26 System Setup/Hold Rise 波形图.....	195
图 6-27 System Setup/Hold Fall 波形图.....	195
图 6-28 System Setup/Hold Dual 波形图 .....	195
图 6-29 Source Setup/Hold Rise 波形图 .....	196
图 6-30 Source Setup/Hold Fall 波形图 .....	196
图 6-31 Source Setup/Hold Dual 波形图.....	196
图 6-32 Source Skew Rise 波形图.....	197
图 6-33 Source Skew Fall 波形图 .....	197
图 6-34 Source Skew Dual 波形图 .....	197
图 6-35 Set Multicycle Path 示例 (start) .....	205
图 6-36 Set Multicycle Path 示例 (end) .....	206
图 7-1 启动局部动态重配.....	214
图 7-2 界面显示.....	215
图 7-3 创建 Partition Definition 操作.....	215
图 7-4 界面说明.....	216
图 7-5PD_led 图标.....	216
图 7-6 Partition Definition 界面 .....	217
图 7-7 PD 的默认 RM .....	217
图 7-8 流程运行时.....	217
图 7-9 Multiple Flow 窗口 .....	218
图 7-10 禁止操作 1.....	219
图 7-11 禁止操作 2.....	219
图 7-12 禁止操作 3.....	220
图 7-13 禁止操作 4.....	221
图 7-14 禁止操作 5.....	221
图 7-15 禁止操作 6.....	222
图 7-16 禁止操作 7.....	222
图 7-17 查看和编辑 PD 属性操作.....	223

图 7-18 remove PD .....	224
图 7-19 删除前界面 .....	224
图 7-20 删除后界面 .....	225
图 7-21 remove inactive flow .....	225
图 7-22 添加 RM .....	226
图 7-23 添加页面 .....	226
图 7-24 添加 RM 后 .....	227
图 7-25 查看和更改 RM .....	227
图 7-26 删除 RM .....	228
图 7-27 删除 RM 前 .....	228
图 7-28 删除 RM 后 .....	229
图 7-29 删除 IF .....	229
图 7-30 RM 添加 Design .....	230
图 7-31 RM 的 module heir 节点添加 Design .....	230
图 7-32 RM 的 top module .....	231
图 7-33 RM 删除 Design .....	231
图 7-34 原 design 和界面 .....	232
图 7-35 添加后 .....	233
图 7-36 原 design 和界面 .....	233
图 7-37 删除后 .....	234
图 7-38 删除 Inactive Flow .....	234
图 7-39 region1 .....	235
图 7-40 流程与 PD、PD Instance 及 RM 的对应关系 .....	236
图 7-41 切换流程树 .....	237
图 7-42 流程依赖 .....	237
图 7-43 流程关系 .....	238
图 7-44 运行多流程 .....	239
图 7-45 同时启动多流程 .....	239

---

图 7-46 主流程支持组件.....	240
图 7-47 打开 project setting .....	241
图 7-48 project setting 界面 .....	241
图 7-49 点击 report.....	242
图 7-50 report summary .....	242
图 7-51 点击 Messages .....	243
图 7-52 显示 RM2 流程 Messages .....	243
图 7-53 点击 Log .....	244
图 7-54 显示 RM2 流程 Log .....	244

## 表目录

表 3-1 PGC 系列支持情况.....	155
表 3-2 其他功能支持情况.....	159
表 9-1 add_constraint .....	247
表 9-2 add_designt .....	247
表 9-3 add_fic .....	247
表 9-4 add_simulation.....	248
表 9-5 add_sources.....	248
表 9-6 clean.....	248
表 9-7 set_arch .....	249
表 9-8 compile.....	249
表 9-9 dev_map.....	249
表 9-10 位流 .....	251
表 9-11 生成网表.....	251
表 9-12 pnr .....	252
表 9-13 report power .....	254
表 9-14 report_timing.....	254
表 9-15 仿真 .....	254
表 10-1 constraint error .....	255
表 10-2 PCE 错误 .....	257
表 10-3 UCE 错误.....	257
表 10-4 FabFlow .....	259
表 10-5 DB .....	260
表 10-6 UiWgt.....	261

---

表 10-7 DRC error.....	263
表 10-8 map.....	263
表 10-9 place .....	266
表 10-10 router .....	267
表 10-11 DE error.....	267
表 10-12 timing error .....	272
表 10-13 Simulation error .....	272

# 1 关于手册

Pango Design Suite 详细手册面向所有 Pango Design Suite 用户，旨在帮助用户通过 Pango Design Suite 快速地完成 FPGA 的设计工作。Pango Design Suite 详细手册提供了使用 Pango Design Suite 进行 FPGA 开发的基本操作、开发流程、常见错误等详细说明。

## 1.1 软件简介

Pango Design Suite 是一款致力于 FPGA 开发的工具软件，其主要功能包括设计输入、综合、仿真、实现和位流。Pango Design Suite 具有界面友好、操作简单等特点，能够借助一些常用第三方 EDA 软件(主要是逻辑综合工具和仿真工具)完成 FPGA 全流程开发。

## 1.2 更多帮助

[Pango\\_Design\\_Suite\\_User\\_Guide](#)

[Pango\\_Design\\_Suite\\_Quick\\_Start\\_Tutorial](#)

[Pango\\_Power\\_Calculator\\_User\\_Guide](#)

[Pango\\_Power\\_Planner\\_User\\_Guide](#)

[Physical\\_Constraint\\_Editor\\_User\\_Guide](#)

[User\\_Constraint\\_Editor\\_User\\_Guide](#)

[Design\\_Editor\\_User\\_Guide](#)

[Fabric\\_Configuration\\_User\\_Guide](#)

[Fabric\\_Debugger\\_User\\_Guide](#)

[Fabric\\_Inserter\\_User\\_Guide](#)

[IP\\_Compiler\\_User\\_Guide](#)

[Simulation\\_User\\_Guide](#)

[ADS\\_Synthesis\\_User\\_Guide](#)

[ADS\\_Language\\_Support\\_Reference\\_Manual](#)

[Route Constraint Editor User Guide](#)

[Timing Analyzer User Guide](#)

[Pango SSN Analyzer User Guide](#)

Pango SSN Estimator User Guide

## 2 基本操作

以下均基于 Logos 系列下的 PGL50H 行介绍，一般采用默认设置。

### 2.1 启动软件

双击桌面的 Pango Design Suite 快捷方式或在程序菜单中单击相应 Pango Design Suite 的快捷方式，即可启动 Pango Design Suite。Pango Design Suite 启动后的初始界面如下图所示：

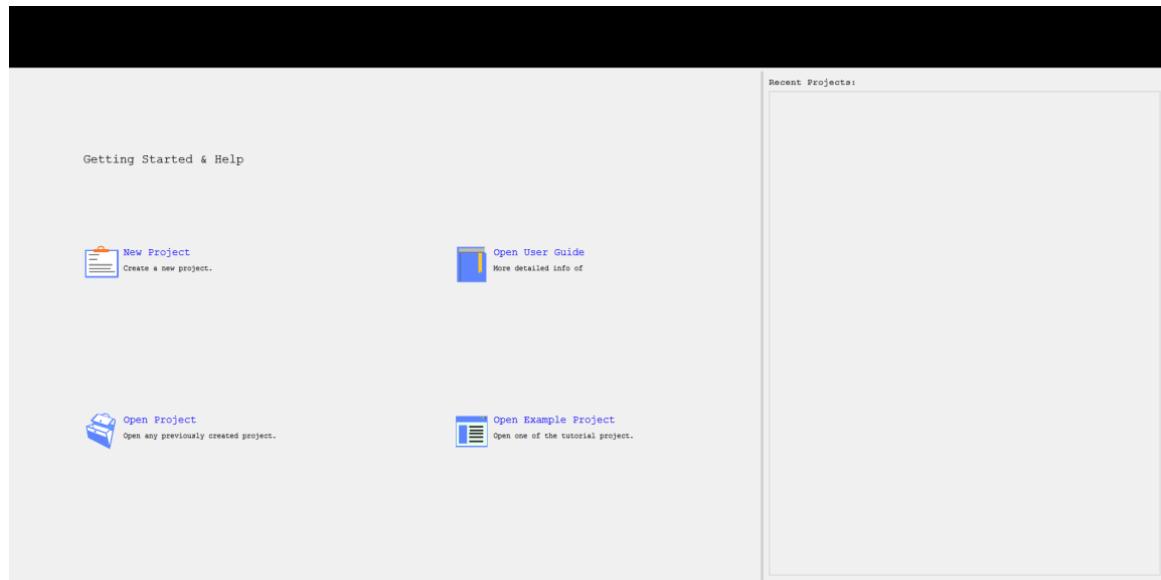


图 2-1 初始界面

每次启动软件时，如果软件的 license 快到期（小于等于 30 天），会提示 license 到期时间。能够让用户有足够的时间提前规划申请 license，以免耽误项目进度。

### 2.2 新建工程

新建工程一般可在两种情形下进行，一是在软件启动的初始界面中新建工程；二是在已有工程上新建工程。软件启动的初始界面有两种方式可进入，第一种是启动软件方式，另一种是在已有工程中选择 Project 菜单的【Close Project】项关闭当前工程后跳转得到。

(1)、新建工程主要是打开新建工程向导【New Project】，然后对其设置。通过在软件启动的初始界面中新建工程，有三种新建工程的方式，下面将介绍这三种方式；在已有工程上新建工程时，可以选择其中的前两种方式创建工程。

a. 可通过选择菜单项【Project】/【New Project...】命令。如下图所示：

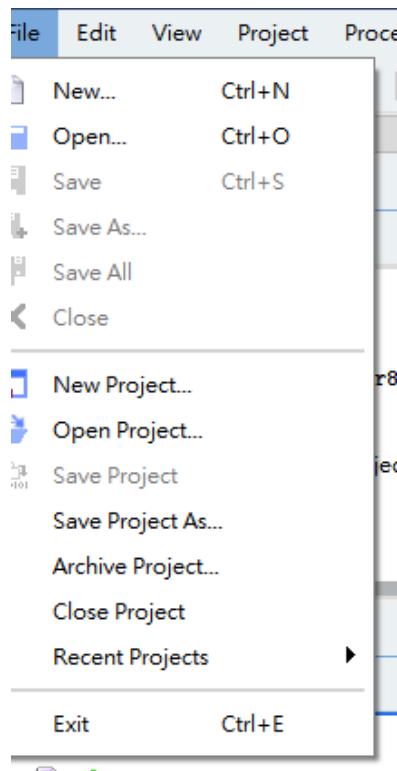


图 2-2 新建工程-菜单

b. 可通过 New Project 快捷方式命令打开。如下图所示：

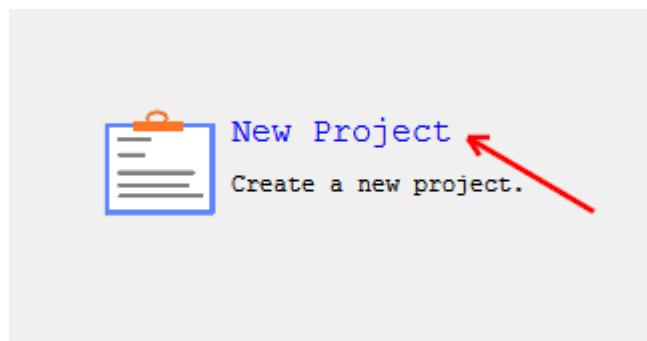


图 2-3 新建工程-快捷方式

(2)、新建工程向导【New Project】，如下图所示：

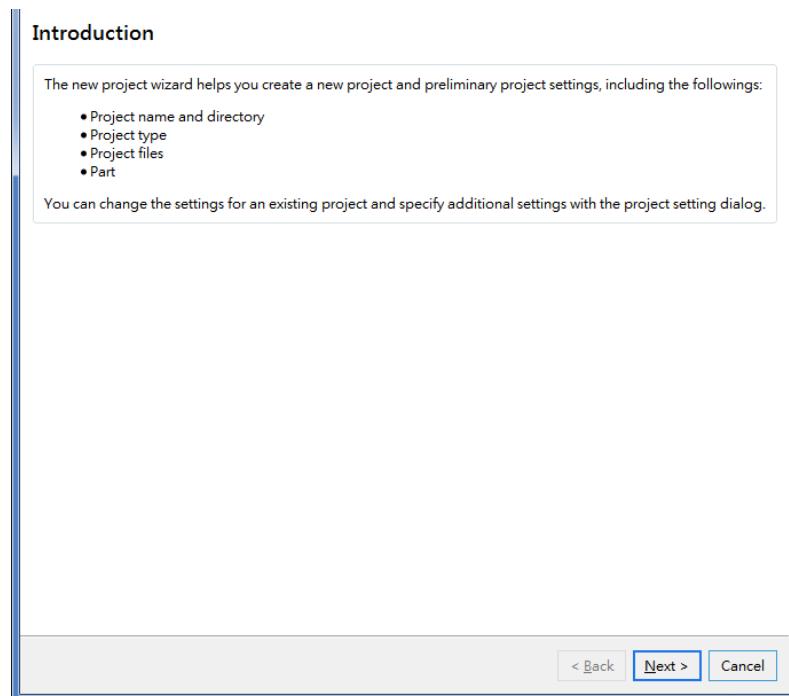


图 2-4 新建工程简介页面

新建工程简单介绍。新建工程大致包括设置工程名和工程路径、工程类型、工程文件及器件信息。单击 Next 出现如下界面：

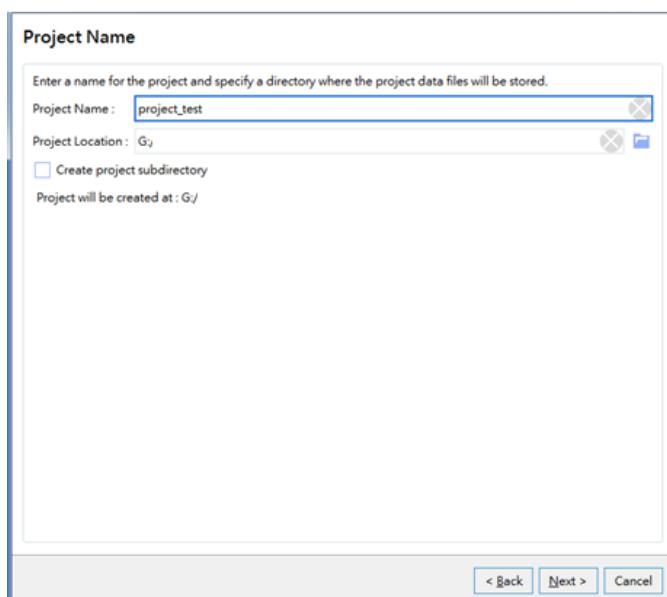


图 2-5 新建工程向导

**【Project Name】**默认为 project，是工程文件名称。（只允许字母、数字、下划线（\_）、杠（-）、点（.））。此示例默认为 project。特殊的，IPC 文件名只允许字母、数字和下划线（\_）。

**【Project Location】**用于选择新工程的工作路径(文件夹名只允许字母、数字、下划线（\_）、杠（-）、点（.）、@、~、,、+、=、#、空格（ ），但空格不能出现在路径名首

尾），即工程文件放置的路径。比如选择添加文件时，软件添加文件的对话框就会以该路径作为默认路径。软件除支持工程文件模式的运行方式以外，也支持脚本模式，其脚本中的相对路径就是相对于此路径。如 tcl 命令 add\_design –verilog ./add.v，就是默认在该工作路径中。一般情况下，源文件等文件会放到此文件夹下。此示例选择 G:/。

**【Create Project Subdirectory】** 将工程文件名作为工作目录的一部分。

设置好工程名和工程路径后，单击 Next 出现选择工程类型界面：

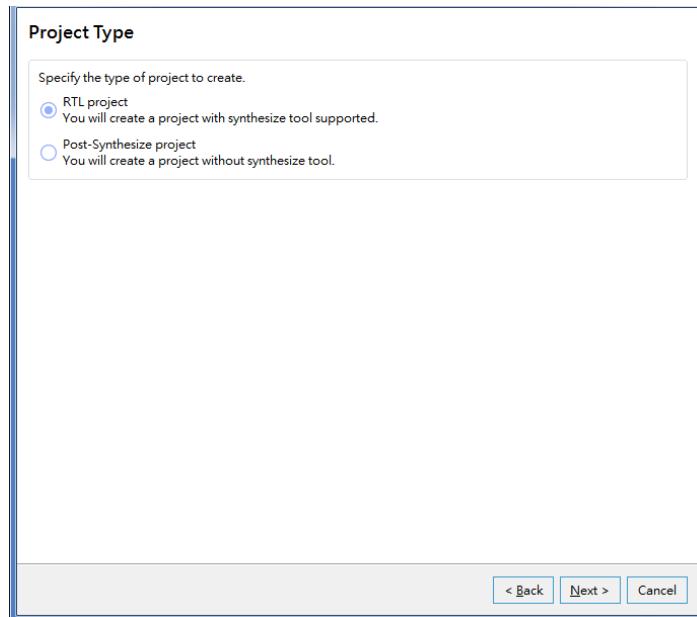


图 2-6 新建工程类型界面

**【RTL Project】** 用于创建 RTL 工程。新建的工程可以执行 synthesize, device map, place&route, report timing, report power, generate netlist 及 generate bitstream 等。

**【Post-Synthesize Project】** 用于创建综合后工程。新建的工程可以执行 device map, place&route, report timing, report power, generate netlist 及 generate bitstream 等。

选择好工程类型后，点击 Next 出现添加 rtl 文件界面：

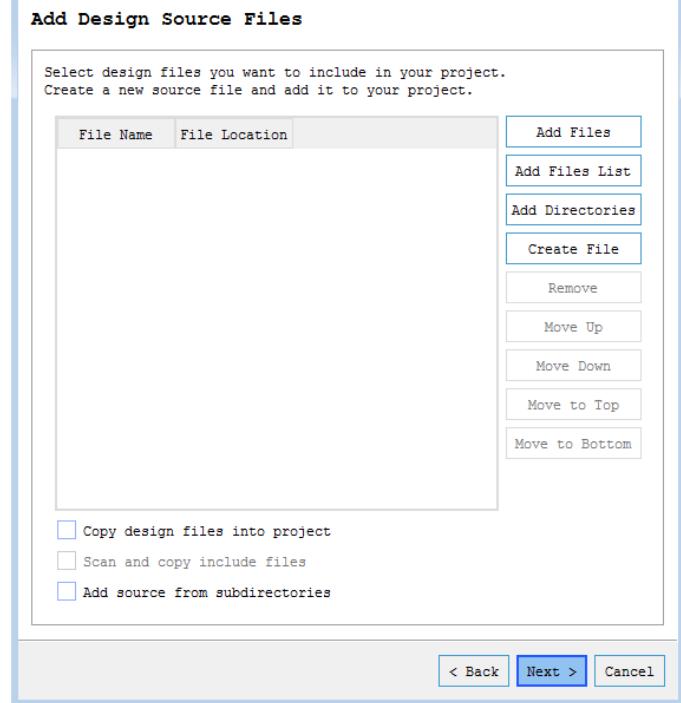


图 2-7 新建工程添加 designs

该界面可以 Add Files 和 Add Directories 来添加 rtl 源文件及新建 rtl 源文件，以及调整 rtl 文件编译顺序，Add Files 添加选中的文件，Add Directories 添加选中的文件夹下所有合适的文件，若勾选了下方的 Add source from subdirecotires 则添加所有的子目录下合适的文件，添加完 rtl 文件后，单击 Next 出现添加 IP 界面，IP 界面的 Add Files 与 Add Directories 与上述相同如下：

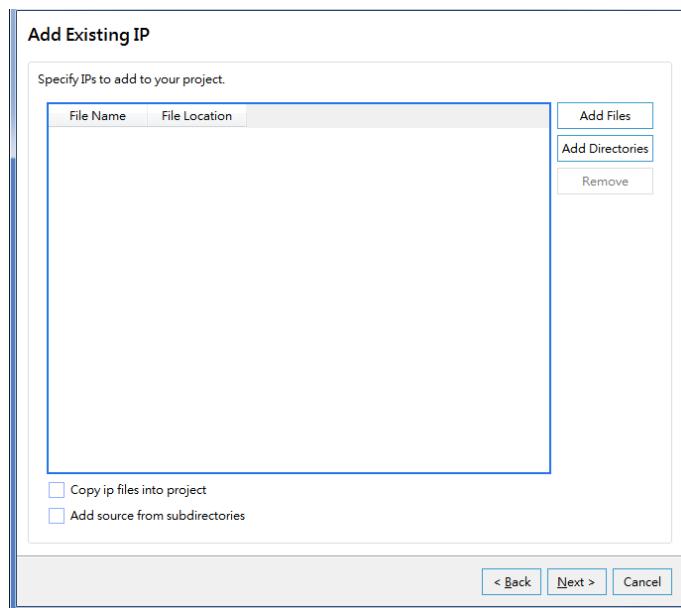


图 2-8 新建工程添加 IP

该界面可以添加 IP 文件。单击 Next 出现添加约束界面：

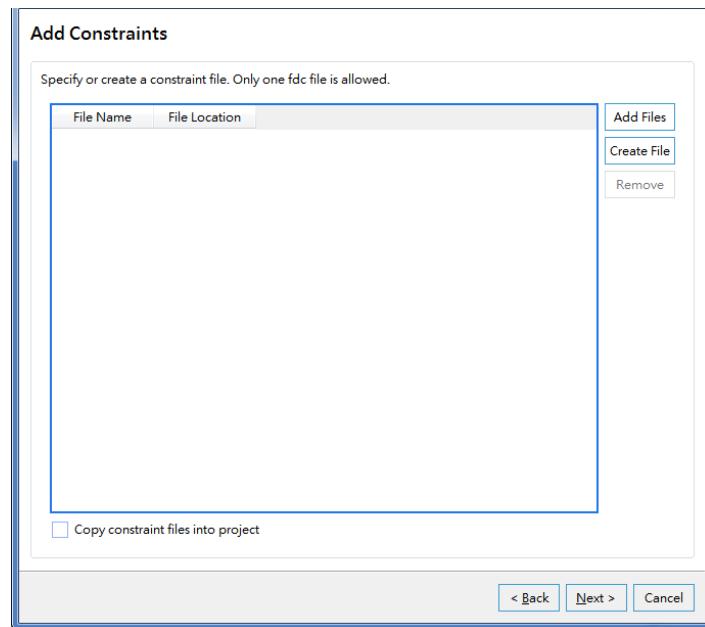


图 2-9 新建工程添加约束

该界面可以添加或创建约束文件。点击 Next 出现器件选择界面：

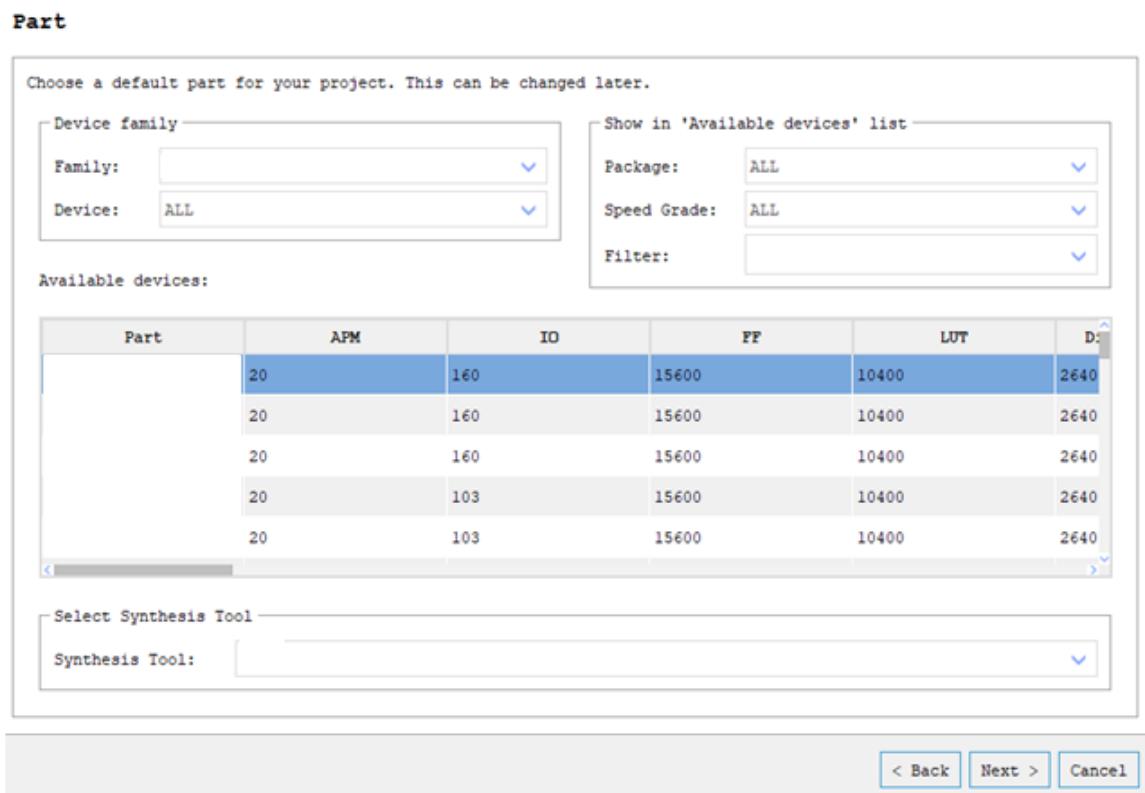


图 2-10 新建工程选择器件

在该界面可以通过 Synthesis Tool 的下拉列表选择需要的综合工具，配置有 Synplify Pro 工具的 PDS 可选项有 Synplify Pro 和 ADS。没有配置 Synplify Pro 工具的 PDS 只有 ADS 一种综合工具。配置好综合工具后单击 Next 出现 summary 界面：

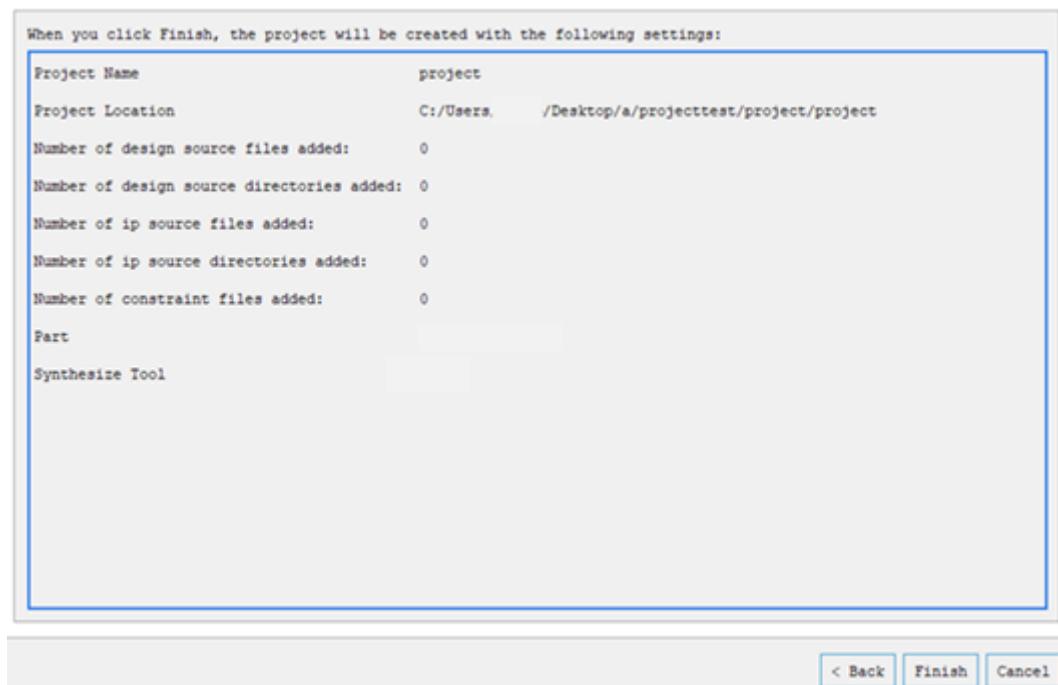
**Summary**

图 2-11 新建工程 summary 界面

(3)、单击 Finish 按钮，工程建立完毕。工程建立完毕后，软件如下图所示。

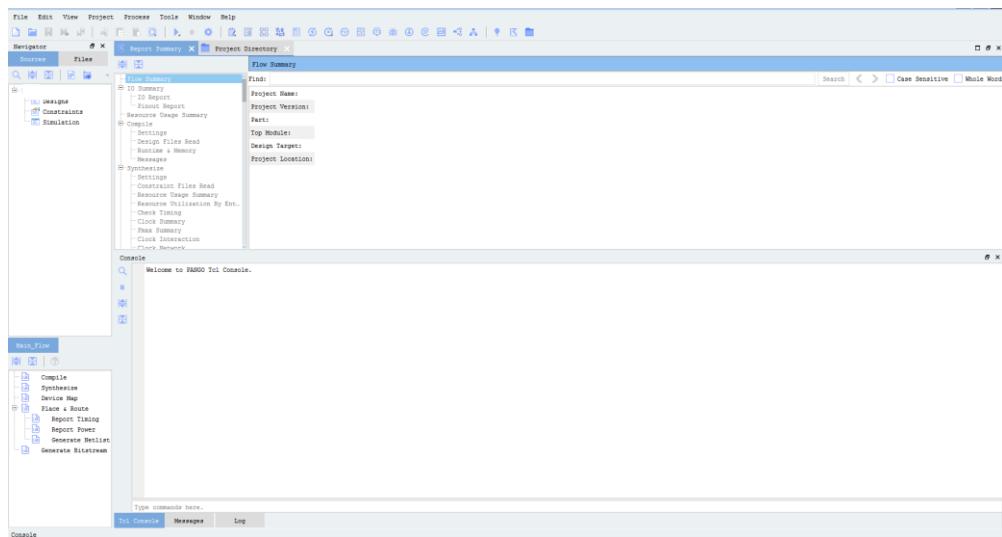


图 2-12 建立工程后的 Pango Design Suite 主界面

## 2.3 主体界面

新建工程后出现 Pango Design Suite 主体界面，如下图所示，由上到下主要分为标题栏、菜单栏、工具栏、工程管理区、报告区、控制台和状态栏 7 个部分。

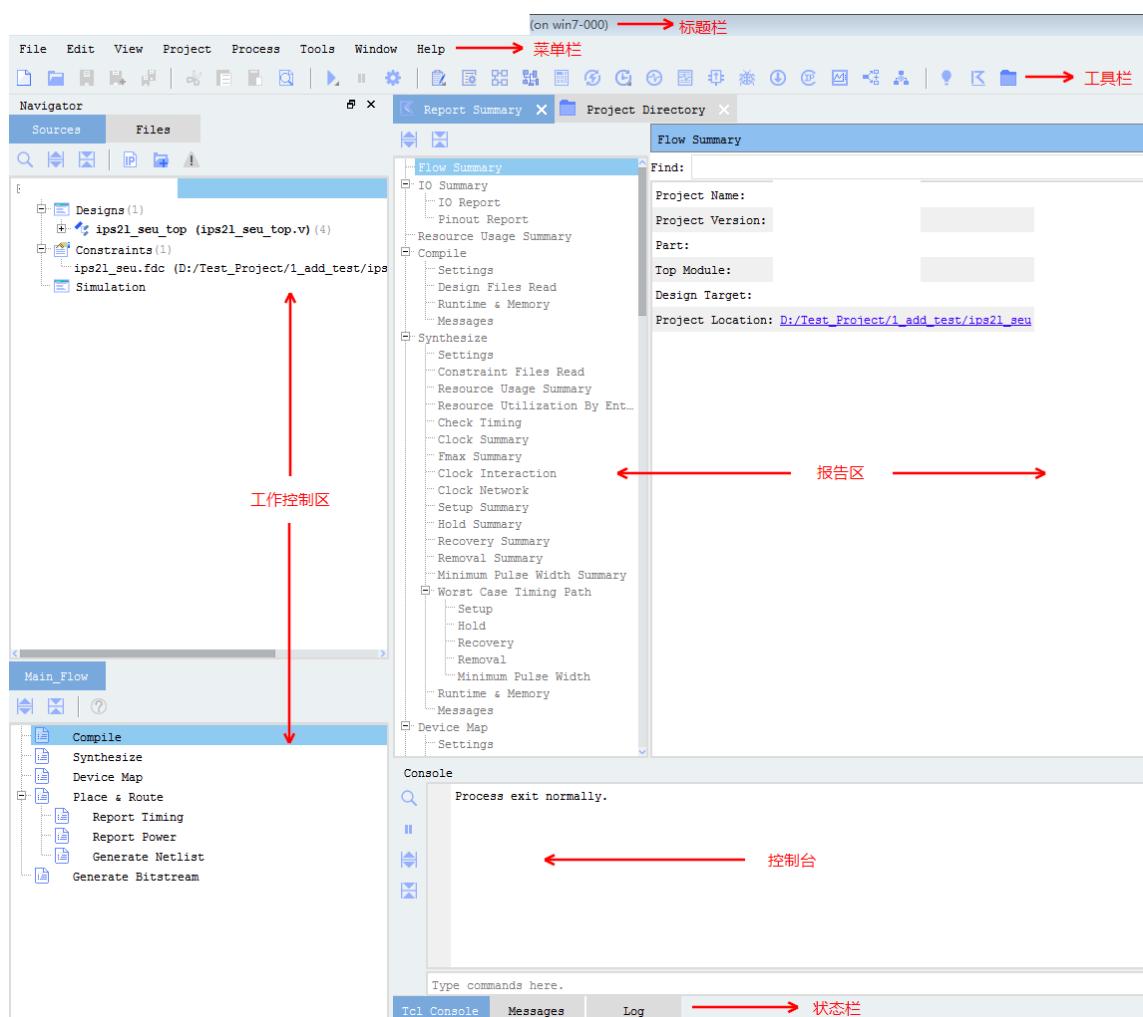


图 2-13 软件主体界面

**标题栏：**主要显示当前工程的名称（包括工程路径）。

**菜单栏：**主要包括“文件【File】”、“编辑【Edit】”、“视图【View】”、“工程【Project】”、“操作【Process】”、“工具【Tools】”、“窗口【Window】”和“帮助【Help】”8个下拉菜单。其使用方法和常用的Windows软件类似。

**工具栏：**主要包含常用命令的快捷按钮。在工具栏中的快捷按钮在菜单栏中都可以找到，灵活运用工具栏可以极大地方便用户在软件中的操作。

**工程管理区【Navigator】：**本窗口是软件操作的主体部分，相关操作和FPGA设计流程紧密相关，包括设计输入、综合、设计实现和生成配置文件等基本功能。进行某个操作后，在处理步骤的前面会出现一个图标来表示该步骤的状态。工程管理区包括【Sources】和【Flow】两部分。【Sources】主要是用来选择器件，添加测试向量以及约束文件等。【Flow】相关操作与Fabric设计流程紧密相关，包括综合、Map、PnR以及生成配置文件等。

**报告区【Report Summary】：**报告区主要是软件运行过程中的一些报告信息的按照action

分类显示。

控制台【Console】：显示软件处理信息，如操作步骤信息、警告信息和错误信息等。控制台的左下脚的 Tcl Console 标签可以在此窗口输入 tcl 命令，实现软件的命令行操作和脚本操作。控制台的左下脚的另一个标签 Messages 窗口中以树形结构显示 error、critical warning、warning、info 消息。在窗口顶部显示各类消息数目，errors、critical\_warnings、warnings、和 infos 的复选框控制消息框中是否需要显示该类信息。Message 窗口支持信息树的折叠和展开和对每一条 message 的搜索功能，Message 窗口支持按消息 ID 分组，即相同 ID 的消息以第一条为父节点，其余的为这个父节点的子节点，父节点需提供其子节点（消息）数目。控制台的左下脚的 Log 标签显示各 action 运行产生的日志

状态栏：显示相关命令和操作信息。

## 2.4 菜单介绍

Pango Design Suite 所有的操作都可以通过菜单完成，下面简要介绍主要的菜单及相应功能。

### (1) 、File 菜单

File 菜单的命令包括 New...、Open...、Save、Save As...、Save All、Close、New Project...、Open Project...、Save Project、Save Project As...、Archive Project...、Close Project、Import Project Settings、Recent Projects 以及 Exit。

File 菜单如下图所示：

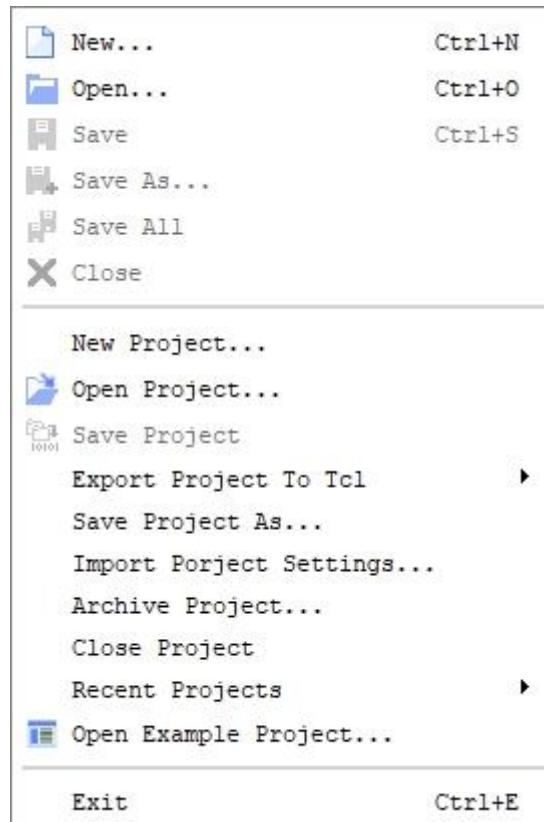


图 2-14 File 菜单

**【New...】**：用于新建源文件，快捷键为“Ctrl+N”。

**【Open...】**：用于打开 Pango Design Suite 软件支持的文件格式的文件。

**【Save】**：用于保存当前源文件。

**【Save As...】**：用于另存当前源文件。

**【Save All】**：用于保存所有当前打开的源文件。

**【Close】**：关闭当前文件。

**【New Project...】**：新建一个工程

**【Open Project...】**：用于打开已有的 Pango Design Suite 工程

**【Save Project】**：保存当前工程

**【Save Project As...】**：将当前工程另存为

**【Export Project To Tcl】**：将当前工程导出为 Tcl 脚本文件

**【Import Project Settings】**：将另一个工程的配置选项导入到当前工程

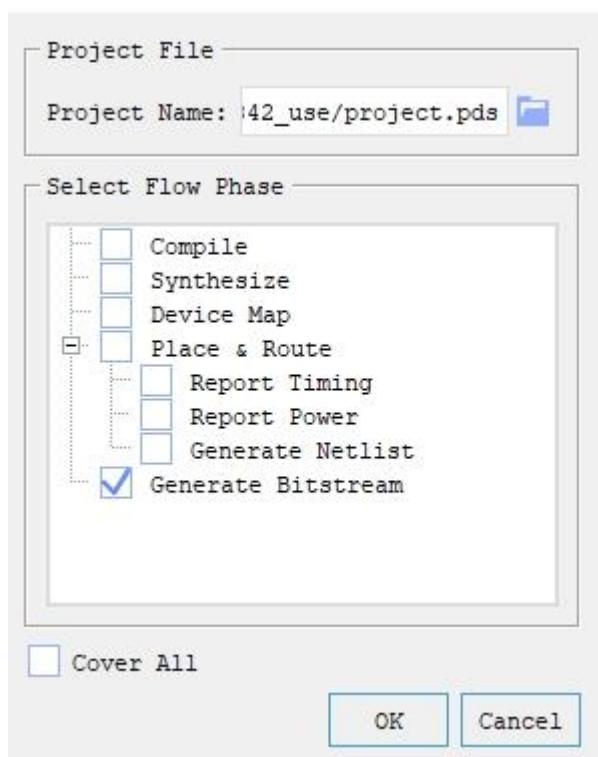


图 2-15 Import Project Settings

选择工程，以及选择将该工程中的哪些步骤的配置选项导入到当前工程，默认不勾选 Cover All 按钮，此时仅导入非默认值的配置选项；若勾选 Cover All 按钮，则会将导入工程的配置选项包括默认值完全复制到当前工程。

**【Archive Project ...】**：将当前工程打包成一个压缩包

**【Close Project】**：关闭当前工程

**【Recent Projects】**：打开最近操作的工程文件列表

**【Exit】**：退出软件，快捷键为“Ctrl+E”。

其中 Export Project To Tcl 打开后界面如下：

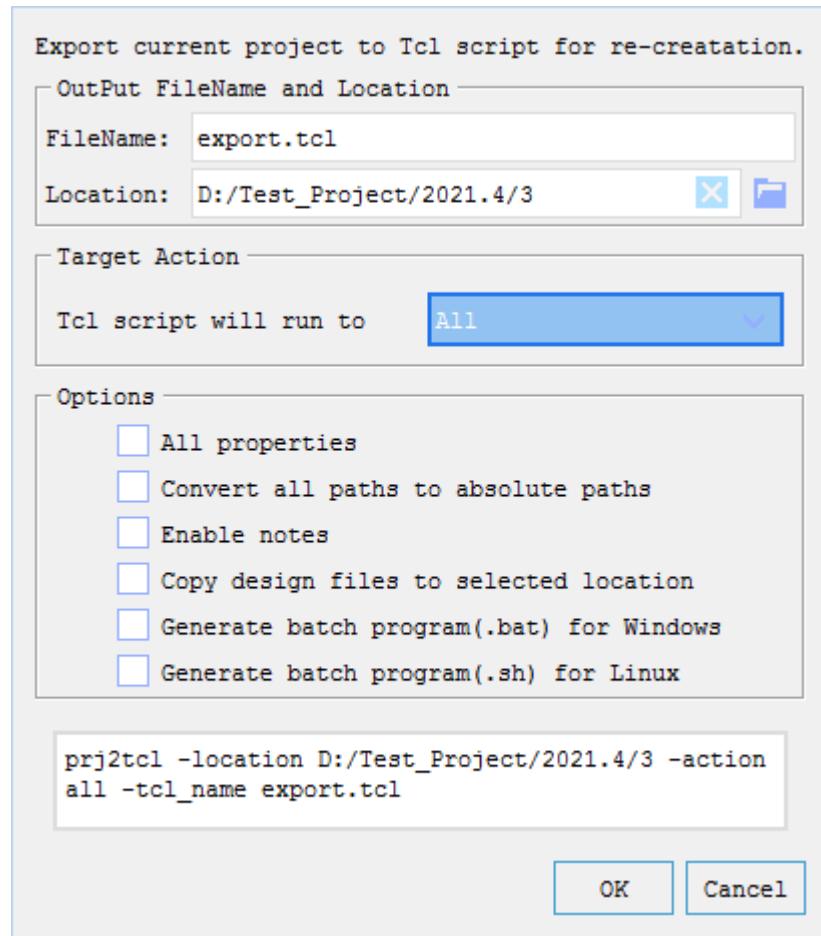


图 2-16 export project to tcl

**【FileName】**：指定导出 tcl 文件的文件名。

**【Location】**：指定导出 tcl 文件的路径。

**【Target Action】**：指定 tcl 文件最终运行的 action

**【All properties】**：勾选后该项后，所有 option 都被勾选。

**【Convert all paths to absolute paths】**：将文件中的相对路径转换为绝对路径。

**【Enable note】**：对 Tcl 命令的用途加以注释。

**【copy design files to selected location】**：拷贝文件内的 Design 文件到选定的路径中。

**【Generate batch program(.bat) for windows】**：生成 Bat 文件，以便在 Windows 平台快速打开 PDS 运行 tcl script。

**【Generate batch program(.sh) for Linux】**：生成 Sh 文件，以便在 Linux 平台快速打开 PDS 运行 tcl script。

操作完成后命令栏中可以生成相应的 prj2tcl 命令，然后点击 OK，在指定路径可以生成 tcl 文件。也可以在 tcl console 中执行生成的 prj2tcl 命令导出 tcl 文件。

## (2) 、Edit 菜单

Edit 菜单的命令包括 Cut、Copy、Paste 和 Preferences...。

Edit 菜单如下图所示：

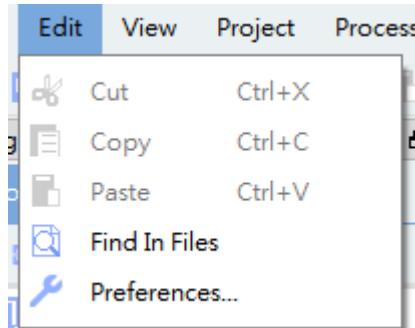


图 2-17 Edit 菜单

**【Cut】**：剪贴选中的代码，快捷键为“Ctrl+X”。

**【Copy】**：复制选中的代码，快捷键为“Ctrl+C”。

**【Paste】**：粘贴剪贴和复制的代码，快捷键为“Ctrl+V”。

**【Find In Files】**：在文件中查询搜索输入的关键字。

**【Preferences...】**：配置选项,如下图。设置 Pango Design Suite 文本样式，一体化工具路径，License 设置。

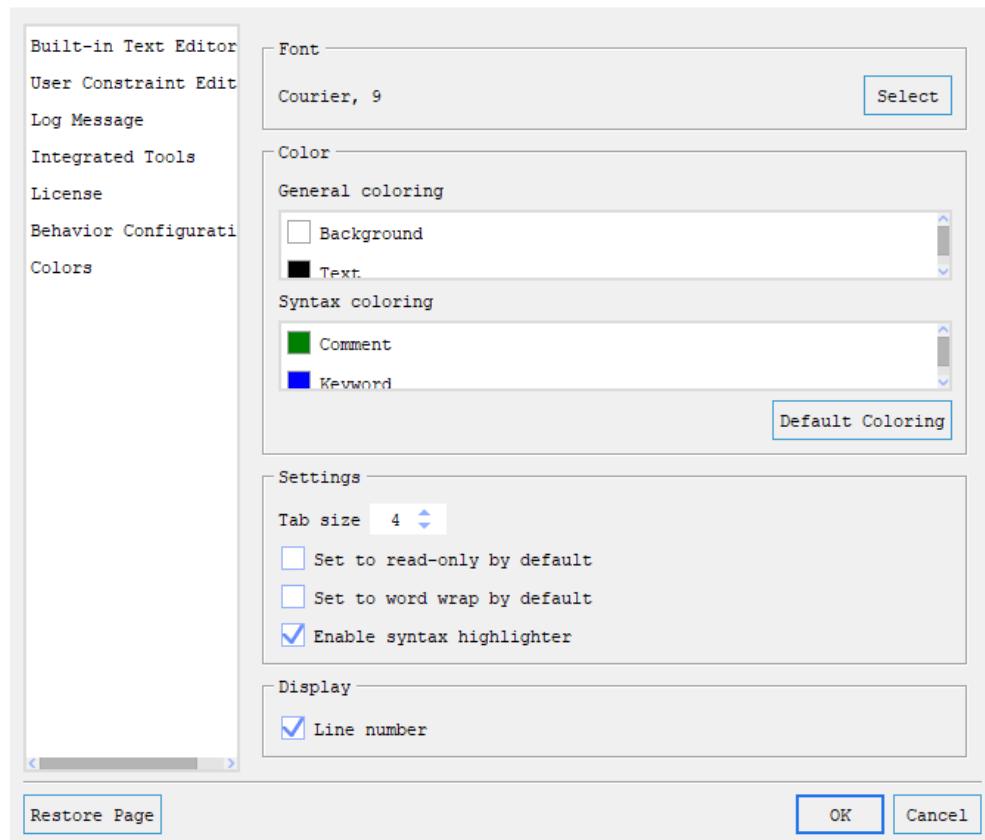


图 2-18 preferences 设置

点击 User Constraint Editor 出现配置 UCE 界面，此界面可以配置进入 UCE 的时候，是全量解析整个网表还是仅解析顶层端口，如下：

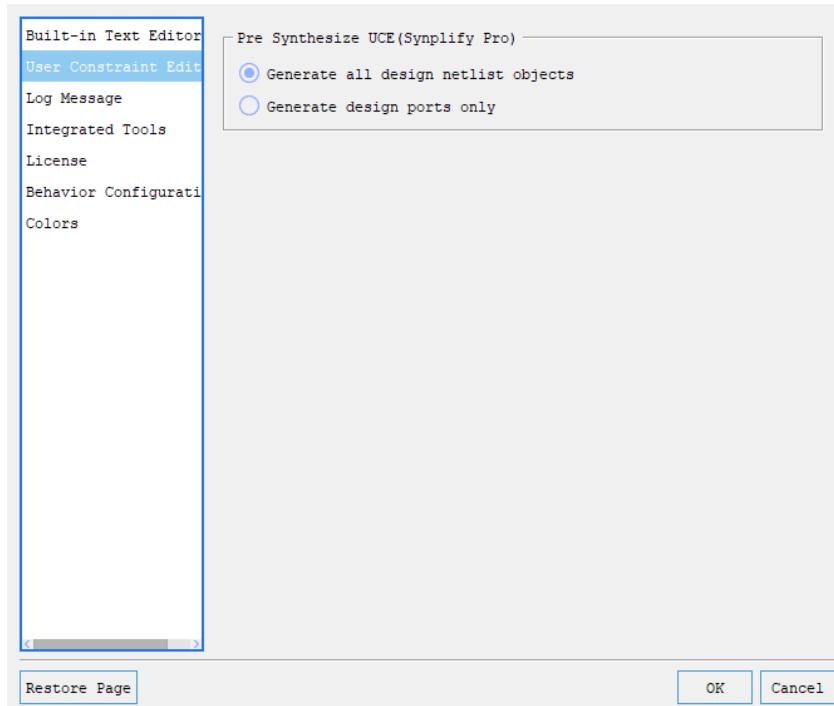


图 2-19 UCE 网表解析配置

点击 Integrated Tools 第三方文本编辑器的路径，如下图所示：

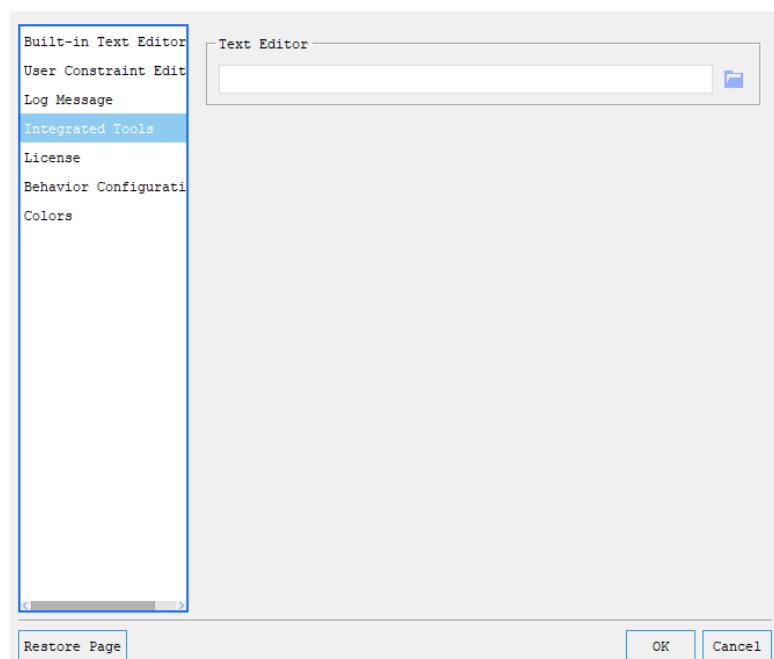


图 2-20 Integrated Tools 配置

在上图中，点击 license，出现如下界面：

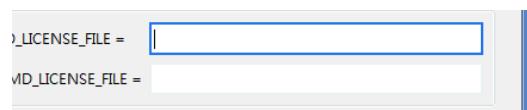


图 2-21 license 设置

其中 PANGO\_LICENSE\_FILE 用于设置 PDS，SNPSLMD\_LICENSE\_FILE 设置 Synplify license（仅配置 Synplify Pro 工具的 PDS 有此设置项）。这两项设置功能和在系统环境变量中设置效果一致。

点击 Behavior configuration 出现人性化防误操作配置界面，有如下配置：

Ask before run out-of-date implementation：出现 run out-of-date 时是否提示

Ask before rerun implementation：点击 rerun 按钮是否配置提示

Ask before rerun all implementation：点击 rerun all 按钮时是否配置提示

Ask before close project：在关闭工程时是否配置提示功能

Ask before close PDS：在关闭软件时是否配置提示功能

Open recent project before starting PDS：启动 PDS 时是否打开最近工程

出现的界面如下：

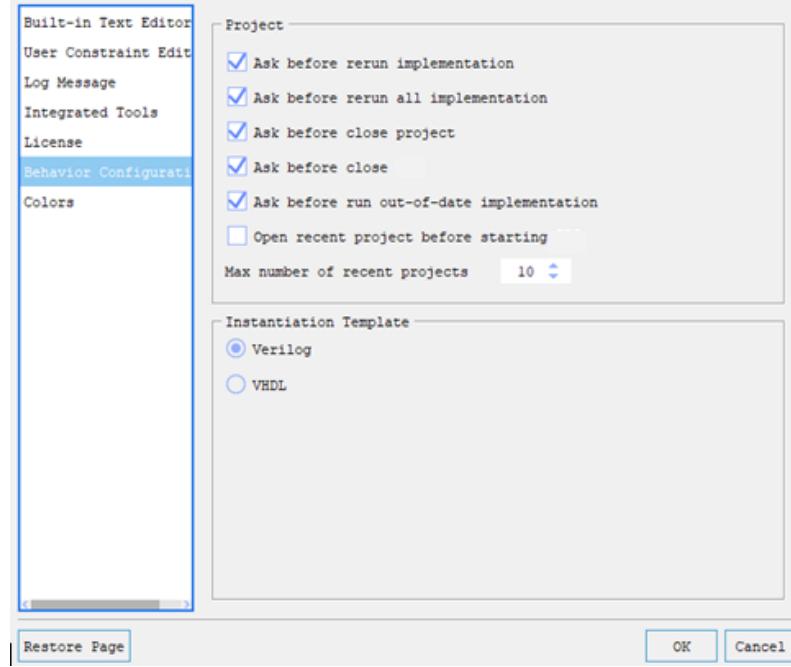


图 2-22 人性化误操作配置

Instantiation Template 是用户配置生成的实例模板的语言格式。

点击 Color 可以进行 mark 功能颜色配置表的配置工作，出现界面如下：

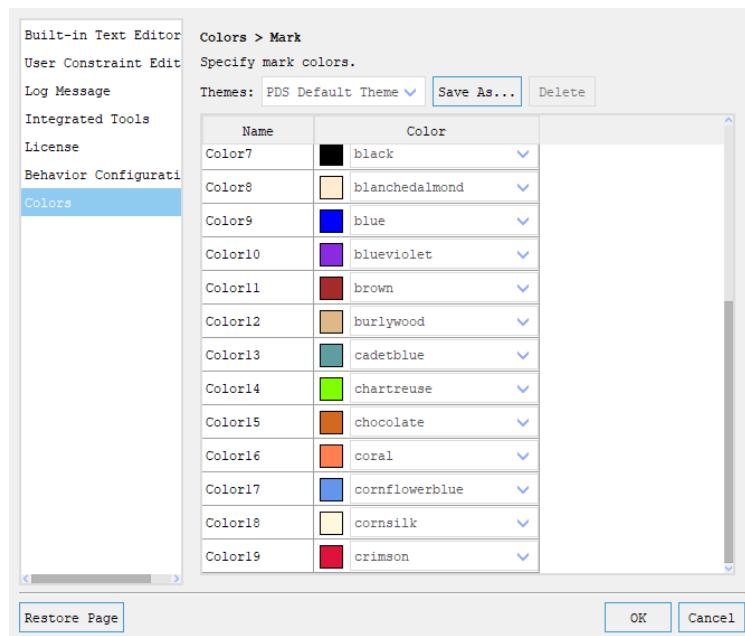


图 2-23 color 配置表

### (3) View 菜单

View 菜单的命令包括 Reset Window Layout、Navigator、Console、File Tool、Edit Tool、Process Tool、Widget Tool、Help Tool。其中的命令从名字上就可以看出其功能，勾选相关选项，则在软件界面会有对应的图形显示。

View 菜单如下图所示：

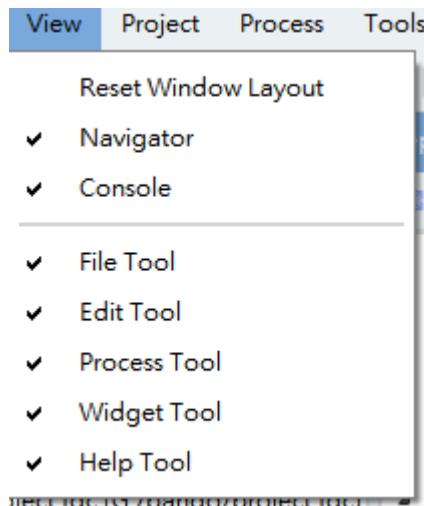


图 2-24 View 菜单

### (4) Project 菜单

Project 菜单的命令包括 Enable Partition Reconfigure 、Select Tcl Script To Run 、New IP、Add Source、Project Setting、Project Cleanup。

Project 菜单如下图所示

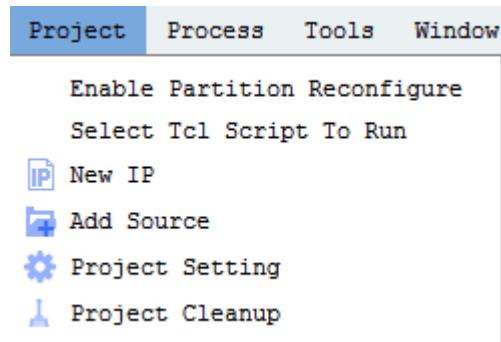


图 2-25 Project 菜单

**【Enable Partition Reconfigure】**：局部动态重配

**【Select Tcl Script To Run】**：指定 tcl 文件运行

**【New IP】**：新建一个 IP 到当前工程。

**【Add Source】**：为当前工程添加一个新的源文件。

**【Project Setting】**：设置当前工程选项

**【Project Cleanup】**：可以选择性的清理工程运行流程生成的临时文件。

#### (5) 、Process 菜单

Process 菜单的命令包括 Run、Rerun、Rerun All、Stop、Run Simulation 和 Pre Synthesize Constraint Check。Process 菜单如下图所示

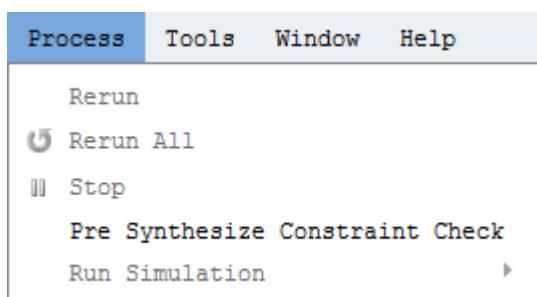


图 2-26 Process 菜单

**【Rerun】**：重新执行当前选中操作之前的所有未执行过的操作（包括前选中操作），如果当前选中操作之前的操作是 out of date 的，则跳过该操作。

**【Rerun All】**：重新执行当前操作及其前序步骤的所有操作。

**【Stop】**：立即停止当前执行的操作。

**【Run Simulation】**：运行仿真。

**【Pre Synthesize Constraint Check】**：约束检查。该按钮的功能是对添加至项目中所有约束文件进行检查（不含 lcf 和 scf 约束文件）。如果需要对单个约束文件进行检查，右键单击项目中约束文件进行单个约束文件检查。ADS 流程使用该功能，将生成约束检查报告(.ccr)，

OEM 流程使用该功能，生成约束检查报告(.rpt)。

#### (6) Tools 菜单

Tools 菜单的命令包括 User Constraint Editor(Timing and Logic)、Physical Constraint Editor (Post-Map)、Route Constraint Editor、Design Editor、Power Calculator、Power Planner、Timing Analyzer、Inserter、Debugger、Configuration、IP Compiler、Synplify Pro、Compile Simulation Libraries、Schematic Viewer (View RTL Schematic、View Technology schematic)。

Tools 菜单如下图所示

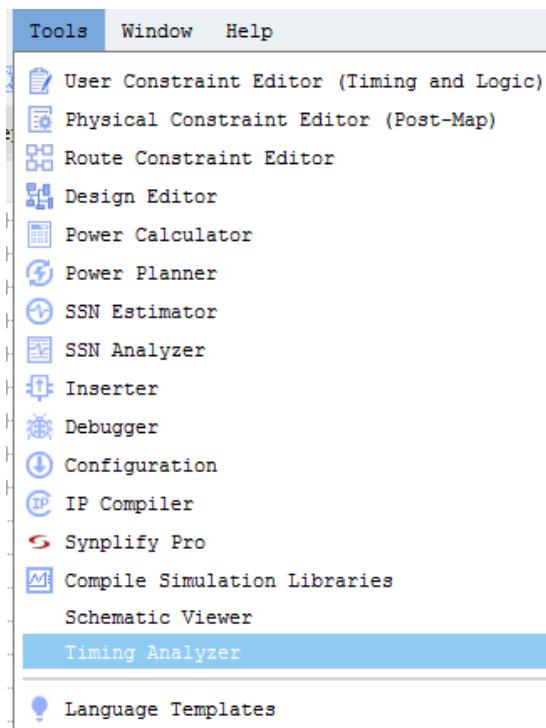


图 2-27 Tools 菜单

【User Constraint Editor (Timing and Logic)】：用户约束

【Physical Constraint Editor (Post-Map)】：打开物理约束

【Route Constraint Editor】：打开布线约束

【Design Editor】：查看 PnR 结果或者进行手动 PnR

【Power Calculator】：功耗分析软件

【Power Planner】：功耗估算软件

【Timing Analyzer】：时序分析工具

【SSN Estimator】：同步噪声估算工具

【SSN Analyzer】：同步噪声分析工具

【Inserter】： Inserter 软件

【Debugger】： Debugger 软件

【Configuration】： Configuration 软件

【IP Compiler】： IP Compiler 软件

【Synplify Pro】： Synplify Pro 软件（仅配置 Synplify Pro 工具的 PDS 有此项）

【Schematic Viewer】： 包含下面两个菜单项

【Compile Simulation Libraries】： 编译仿真库

【View RTL Schematic】： 打开 RTL Schematic Viewer

【View Technology Schematic】： 打开 Technology Schematic Viewer

【Language Templates】： 常用模板信息，下面章节有具体介绍

#### （7）、Window 菜单

Window 菜单如下图所示

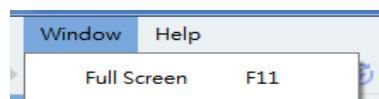


图 2-28 Window 菜单

window 菜单用来控制软件是否全屏显示（Full Screen）。

#### （8）、Help 菜单

Help 菜单的命令包括 Help Topics、Software Manuals 和 About。Help 菜单如下图所示

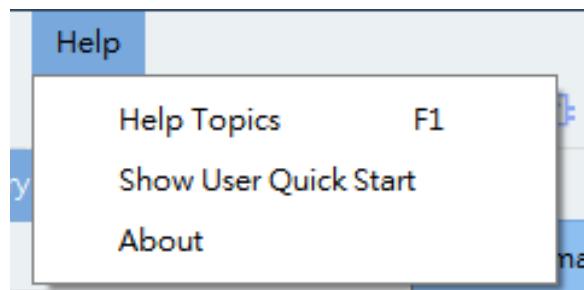


图 2-29 Help 菜单

【Help Topics】： 显示 Help 助手。

【Show User Quick Start】： 查看 Pango Design Suite 用户手册。

【About】： 显示 Pango Design Suite About 信息。

## 2.5 工具介绍

工具栏为常用菜单栏的快捷方式，工具栏右键可直接调出 View 菜单，下面介绍 Language Templates、Report Summary 和 Run 其他请参考菜单栏介绍。

### (1) 、Language Templates

Language Templates 能够快捷的找到一些常用的原语、tcl 命令、GTP 模板、约束模板等信息，也可以实现定义编辑操作，编辑界面支持 CTRL+C 复制操作，如下图：



图 2-30 Language Templates 界面

### (2) 、Report Summary

目前 PDS 的窗口大小有限，文件打开过多的时候，想要显示 report summary 需要一个个移动 tab，不是很方便，故添加 report summary 的快捷方式，如图 2-31：

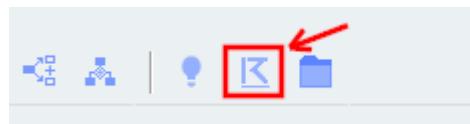


图 2-31 Report Summary 界面

### (3) 、Run

点击菜单栏的 Run 按钮，弹出菜单栏，可选择执行不同的操作，如果该流程已执行完，会重新执行该流程，如图 2-32：

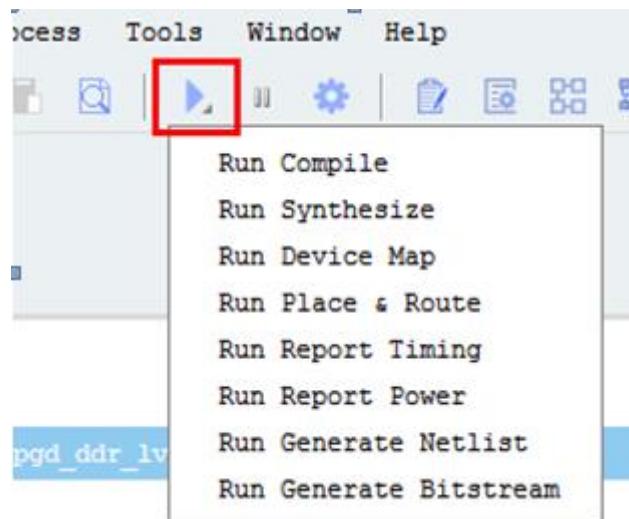


图 2-32 Run 工具

## 2.6 打开工程

打开工程一般可在两种情形下进行，一是在软件启动的初始界面中打开已有工程；二是在已有工程上打开工程。软件启动的初始界面有两种方式可进入，第一种是启动软件方式，另一种是在当前工程页面选择 Project 菜单的【Close Project】项，关闭当前工程后得到。

(1)、打开工程主要是打开选取工程对话框【Open Project】，然后选取已有的工程文件。在已有的工程商打开工程，提供了两种方式如 a,b 所示：

- a. 通过选择菜单项【Project】 / 【Open Project...】打开。如下图所示：

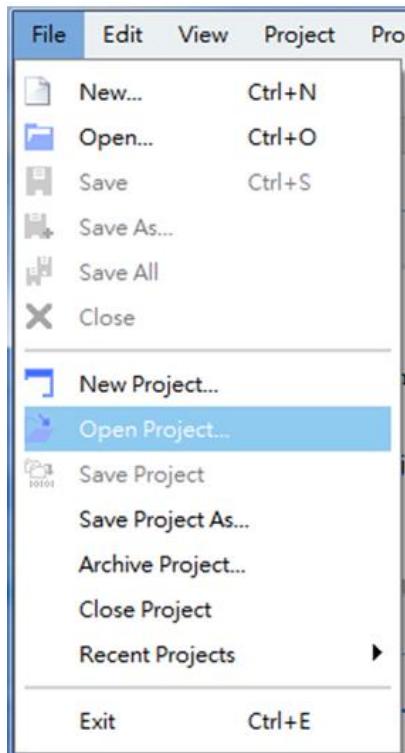


图 2-33 打开工程-菜单

- b. 通过 Open Project 快捷方式命令打开。如下图打开工程-快捷方式：

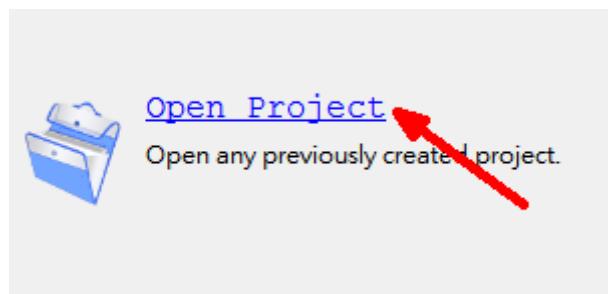


图 2-34 打开工程-快捷方式

- (2)、打开工程对话框【Open Project】。如下图所示：

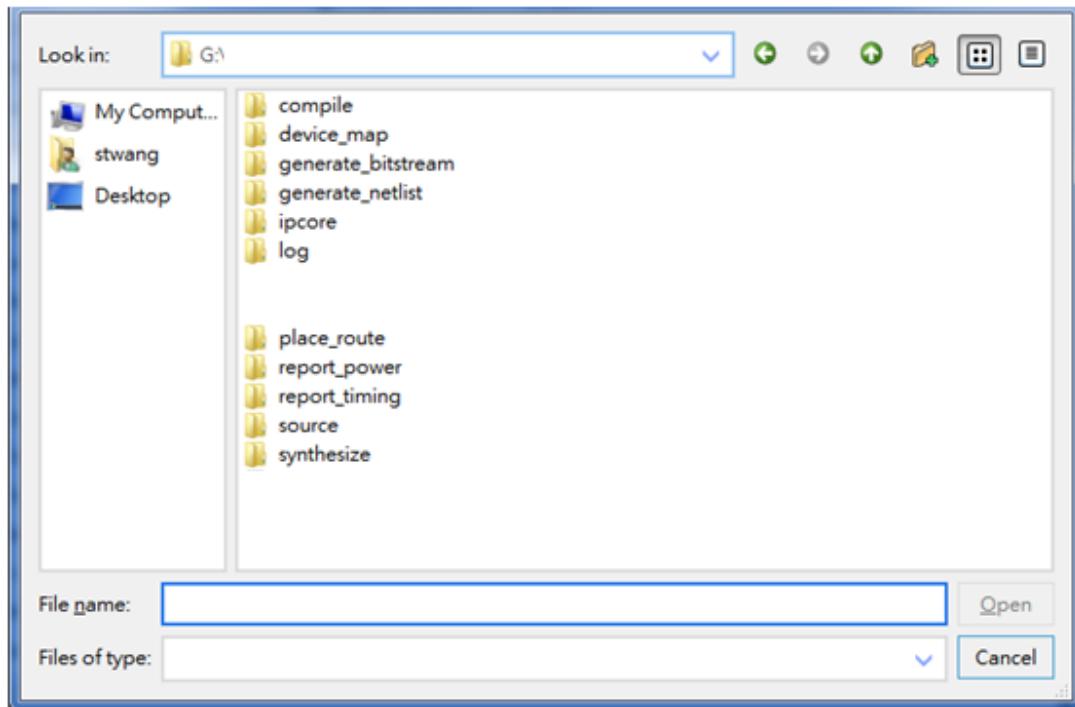


图 2-35 打开工程对话框

(3)、选择完成 .pds 或.prj 工程文件后，点击 【Open】按钮，完成打开工程操作。

工程打开完毕后，软件如下图所示。

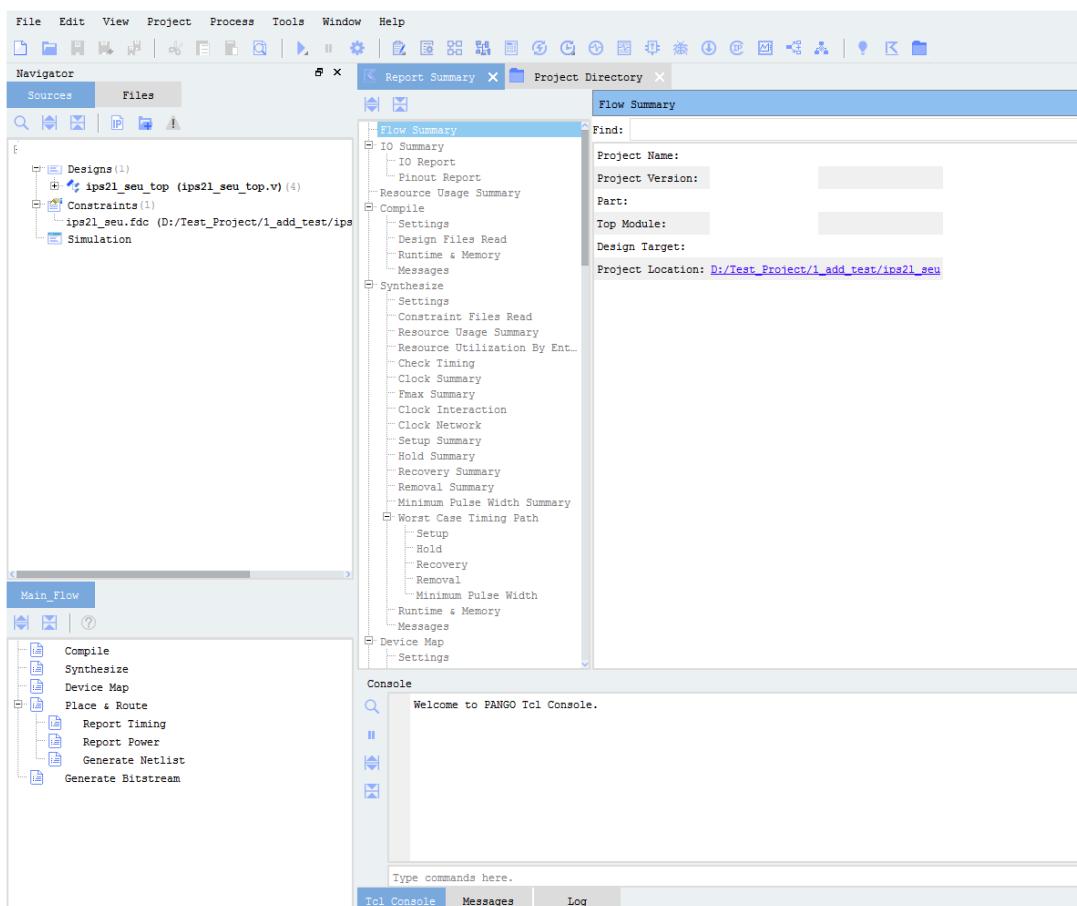


图 2-36 打开工程后的 Pango Design Suite 主界面

特别地，对于【Recent Projects】列表中的工程，则还有两种方式可以打开，主要是通过 Recent Projects 列表来打开。以下示例说明在最近工程类表（最多可容纳十个工程）中打开工程文件的两种方式：

a. 可通过选择菜单项【Project】/【Recent Projects】下的工程列表来打开。如下图所示，打开工程-最近工程-菜单：

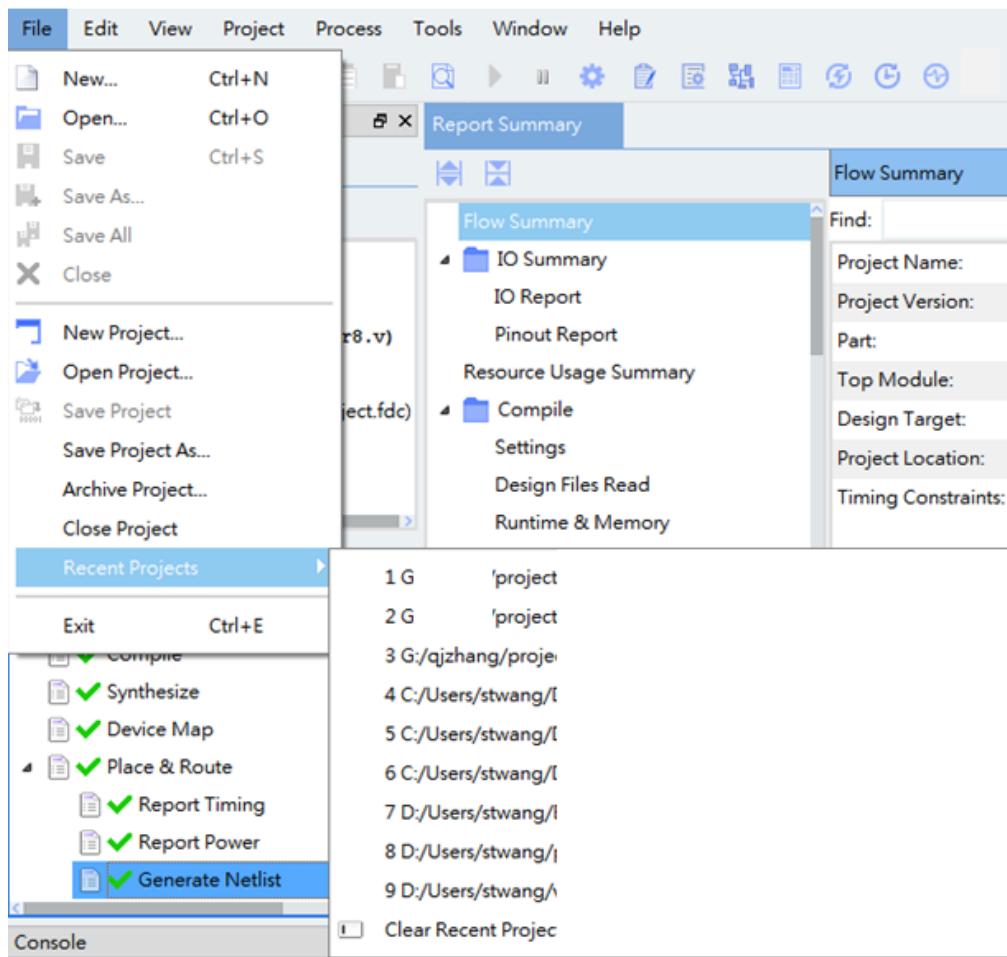


图 2-37 打开工程-最近工程-菜单

当鼠标停留在某个 project 菜单项上面时，在 PDS 主界面下方的状态栏里会显示该 project 的完整路径；

b. 在软件初始界面中，可通过 Recent Projects 工程列表来打开。如下图所示：

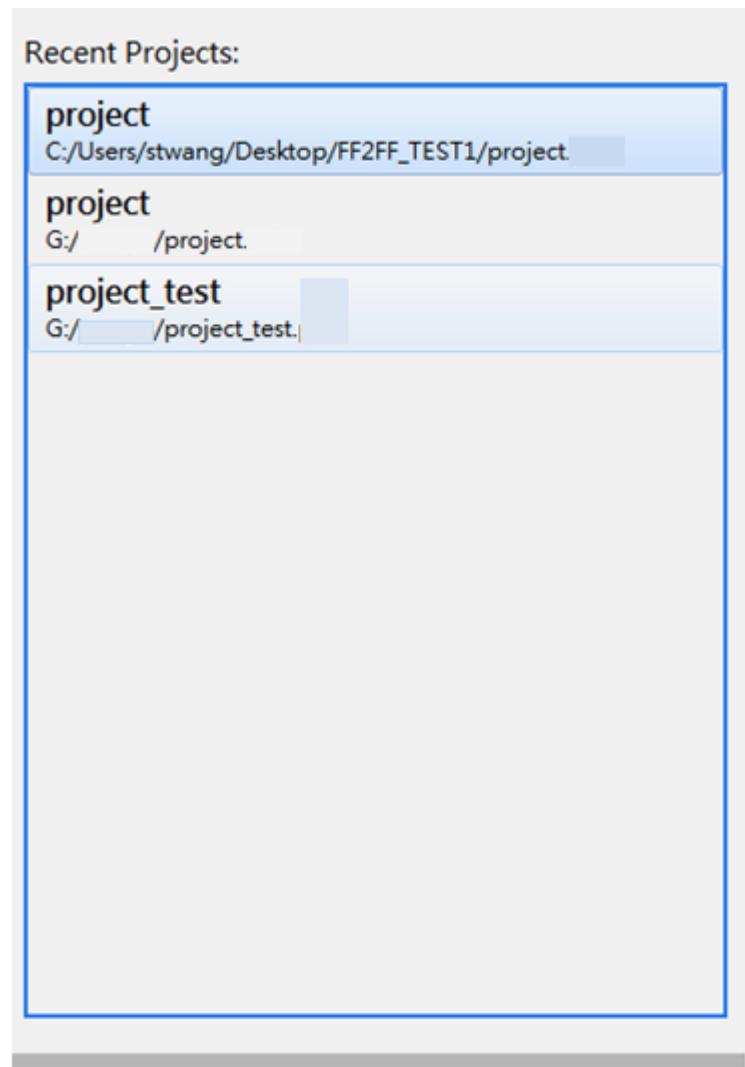


图 2-38 打开工程-最近工程-向导栏

如果打开的工程文件中记录的 device 信息不存在时，会弹出重新选择器件的提示窗口，如下图：

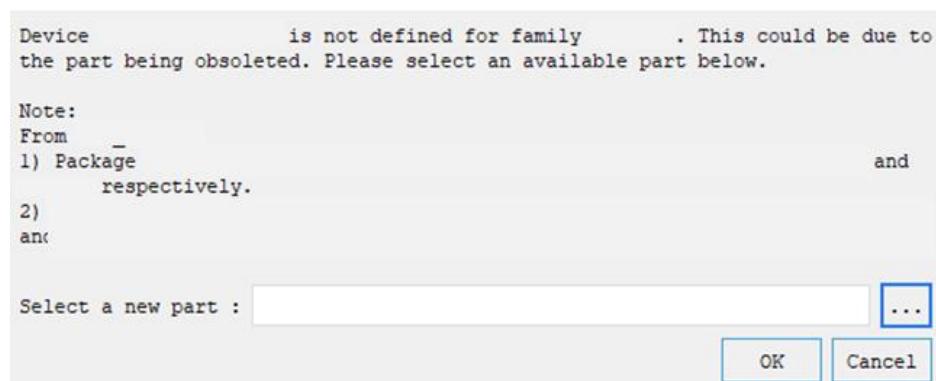


图 2-39 device 信息不存在提示窗口

软件会自动填一个器件，点击右侧“...”按钮可调整具体信息，如下图所示：

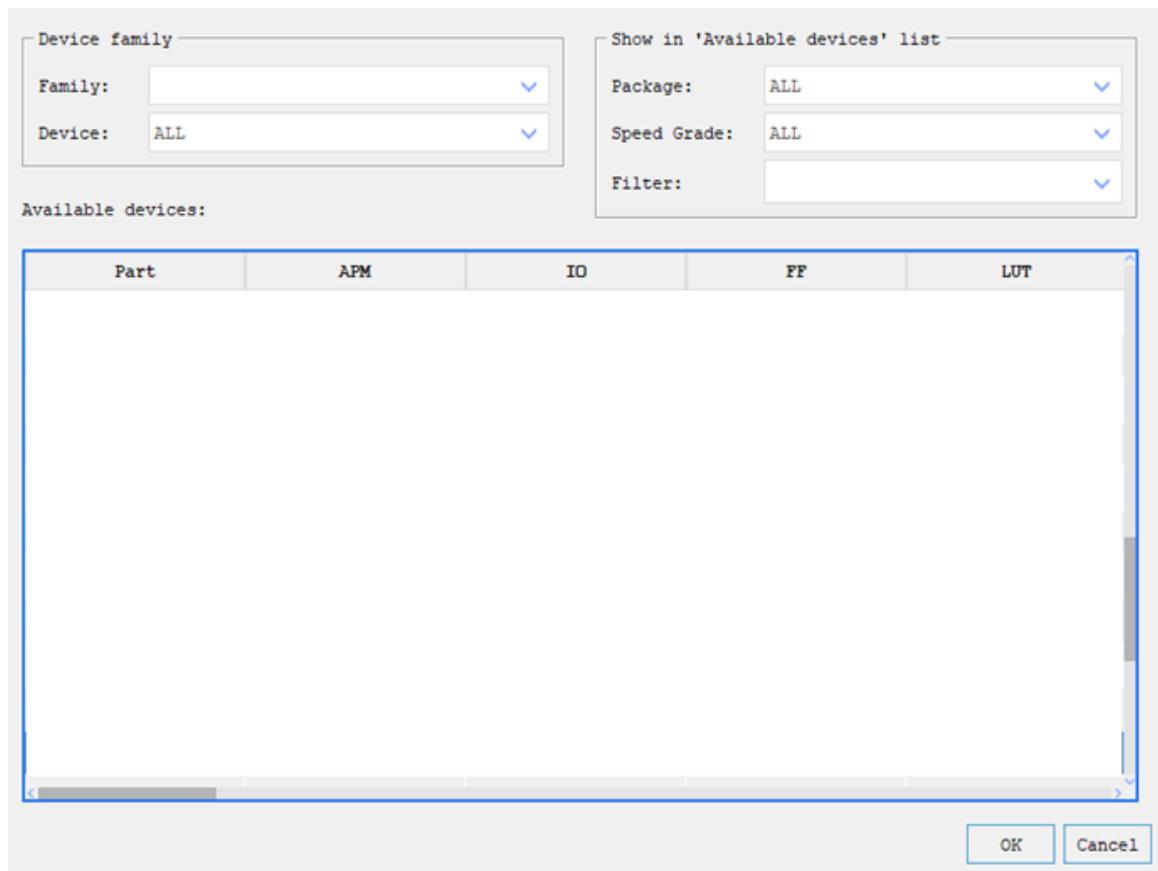


图 2-40 选择 device 信息窗口

当选好器件后，点击 OK 可以用新选择的器件打开工程，点击 cancel，则不打开工程，回到初始界面。

## 2.7 工程管理

Navigator 工程管理区负责执行软件的工作操作流程和添加相关文件。可在工作操作流程名称上双击鼠标左键运行。

Navigator 工程管理区如下图所示

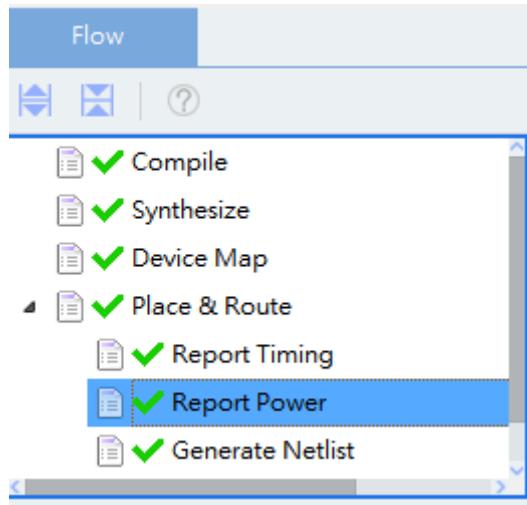


图 2-41 Navigator 工程管理区

Navigator 工程管理区可为 Sources、Files 和 Flow 三个部分。

### 1. Sources 下面的任务或执行体介绍如下：

**【器件信息】**：双击此栏可弹框选择器件；

**【Designs】**：选择添加 Verilog 或者 VHDL 两种格式的文件；

**【Constraints】**：加入逻辑约束文件（时序约束等）。

**【Simulation】**：选择添加 test bench 文件

Source 窗口的右键菜单介绍如下：

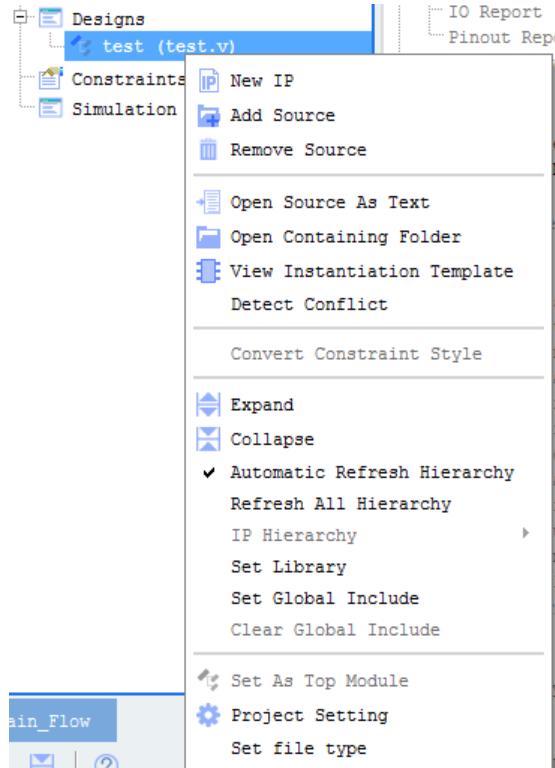


图 2-42 source 窗口右键菜单

- 1) New IP: 新建一个 IP
- 2) Add Source: 添加 hdl, 约束等源文件
- 3) Remove Source: 移除所选的源文件
- 4) Open Source As Text: 以文本方式打开源文件
- 5) Open Containing Folder: 打开源文件所在文件夹
- 6) View Instantiation Template: 例化 design source, IP, simulation 中的 module
- 7) Detect Conflict: 检测是否存在将 ip 的 idf 文件和 ip 的 rtl 文件同时加入工程的场景
- 8) Convert Constraint style: 对目标文件进行约束转换
- 9) Expand: 展开当前节点
- 10) Collapase: 折叠当前节点
- 11) Automatic Refresh Hierarchy: 自动刷新 design hierarchy
- 12) Refresh Hierarchy: 手动刷新 design hierarchy
- 13) IP Hierarchy: 控制是否显示 IP 的层级结构
- 14) Set Library: 为目标文件设置 library name
- 15) Set Global Include: 将目标文件定义成 global include
- 16) Clear Global Include: 将目标文件从 global include 中清除
- 17) Set As Top Module: 将当前 module 设为 top module
- 18) Project setting: 工程设置
- 19) Set file type: 设置文件类型

其中 Set file type 可以设置多种文件类型，具体如下：

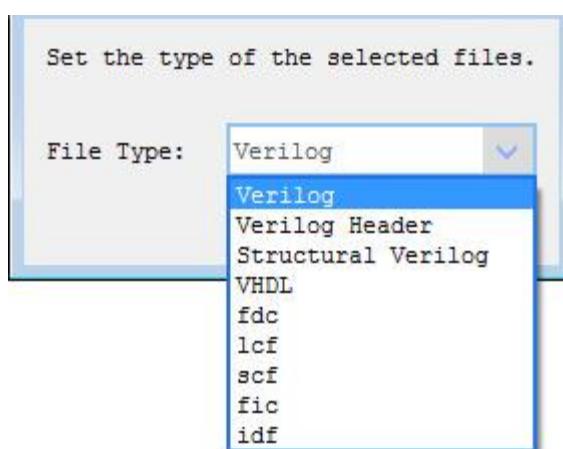


图 2-43 Set file type 界面

2. Sources 区工具栏从左到右介绍如下：



图 2-44 File 窗口工具栏

查找功能：可根据文件名或 module 名搜索

展开功能：将 source 界面的层次结构全部展开

折叠功能：将 source 界面的层次结构全部折叠

新建 IP 功能：新建一个 IP

添加 source 功能：添加各类型源文件

Missing file 功能：显示没有定义的 module

3. Files 下面的内容介绍如下：

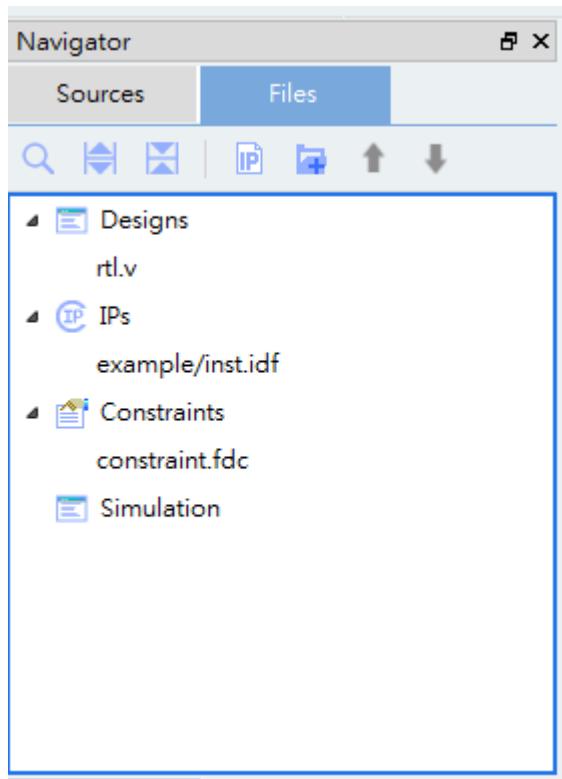


图 2-45 Navigator 工程管理区

【Designs】：选择添加 Verilog 或者 VHDL 两种格式的文件；

【Constraints】：加入逻辑约束文件（时序约束等）。

【Simulation】：选择添加 test bench 文件

File 窗口下工具栏除后面两个为设置 compile order 外，其余的和 Source 窗口的工具栏功能完全一致。

File 窗口的右键菜单介绍如下：

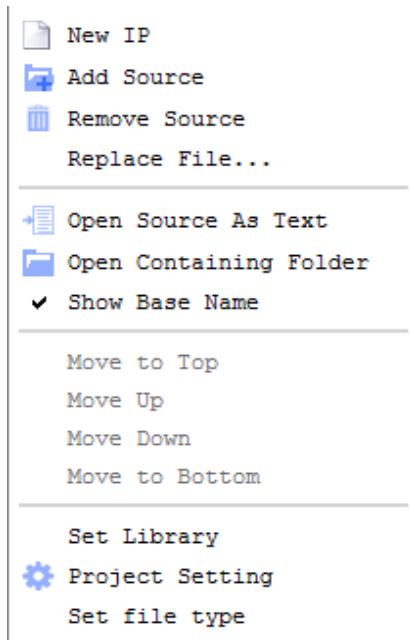


图 2-46 File 右键菜单

- 1) IP: 新建一个 IP
- 2) Add Source: 添加 hdl, 约束等源文件
- 3) Remove Source: 移除所选的源文件
- 4) Replace File...: 替换源文件
- 5) Open Source As Text: 以文本方式打开源文件
- 6) Open Containing Folder: 打开源文件所在文件夹
- 7) Show Base Name: 仅显示文件名, 不勾选则显示文件路径
- 8) Move to Top: 将 hdl 文件移动到所有源文件的顶部
- 9) Move Up: 将 hdl 文件向上移动
- 10) Move Down: 将 hdl 文件向下移动
- 11) Move to Bottom: 将 hdl 文件移动到所有源文件的底部
- 12) Set Library: 为目标文件设置 library name
- 13) Project Setting: 工程设置

4. Flow 下面的任务或执行体介绍如下:

【Compile】: 生成 source code RTL 网表;

【Synthesize】: 综合;

【Device Map】: 映射, 进行 design 网表文件的转化和与器件资源的映射工作;

【Place & Route】: 布局布线;

【Place & Route】 / 【Report Timing】生成布局布线后时序报告；

【Place & Route】 / 【Report Power】生成布局布线后功耗报告；

【Place & Route】 / 【Generate Netlist】生成布局布线后仿真报告；

【Generate Bitstream】：生成位流文件。

工程管理区中的右键菜单：右键菜单中的 Run、Rerun、Rerun All、Stop 和 Project Setting、Select a Seed To Apply，与 Process 菜单对应部分具有相同的功能。如下图所示，Navigator 右键菜单。

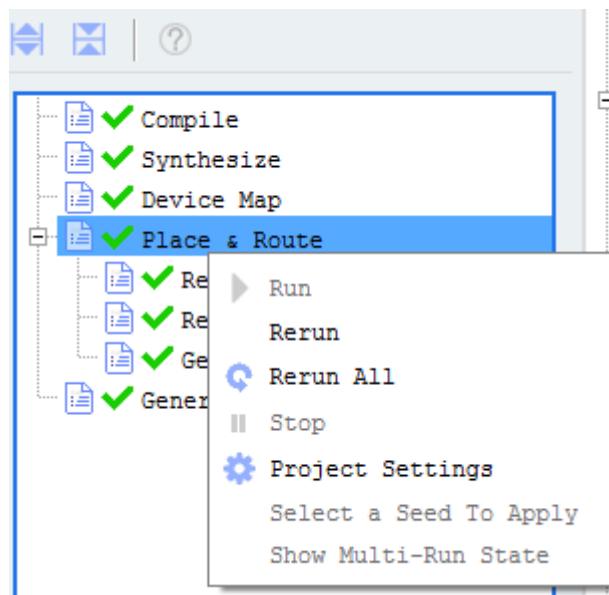


图 2-47 Navigator 右键菜单

在工程管理区中可以通过使用鼠标左键双击来执行某个操作，亦可选中各个功能，然后右键选择【Run】或【Rerun】命令来执行此操作。也可以选择【Rerun All】命令来重新执行该操作之前的所有操作（包括该操作），或在程序运行时选择【Stop】来停止该操作的执行，【Select a Seed To Apply】用户可在 PNR 阶段指定种子运行之后流程。

Flow 上方菜单中可以显示 out of date 信息，点击后可以查看具体的 flow 状态 out of date 原因。如下：

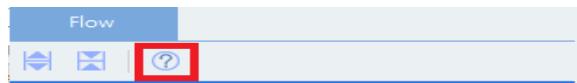


图 2-48 out of date 按钮变亮

## 2.8 设置器件

Part\_settings 介绍如下：双击【Navigator】中【Sources】/【PGL50H】或点击 Project Setting 弹出器件选择窗口，如下图所示：

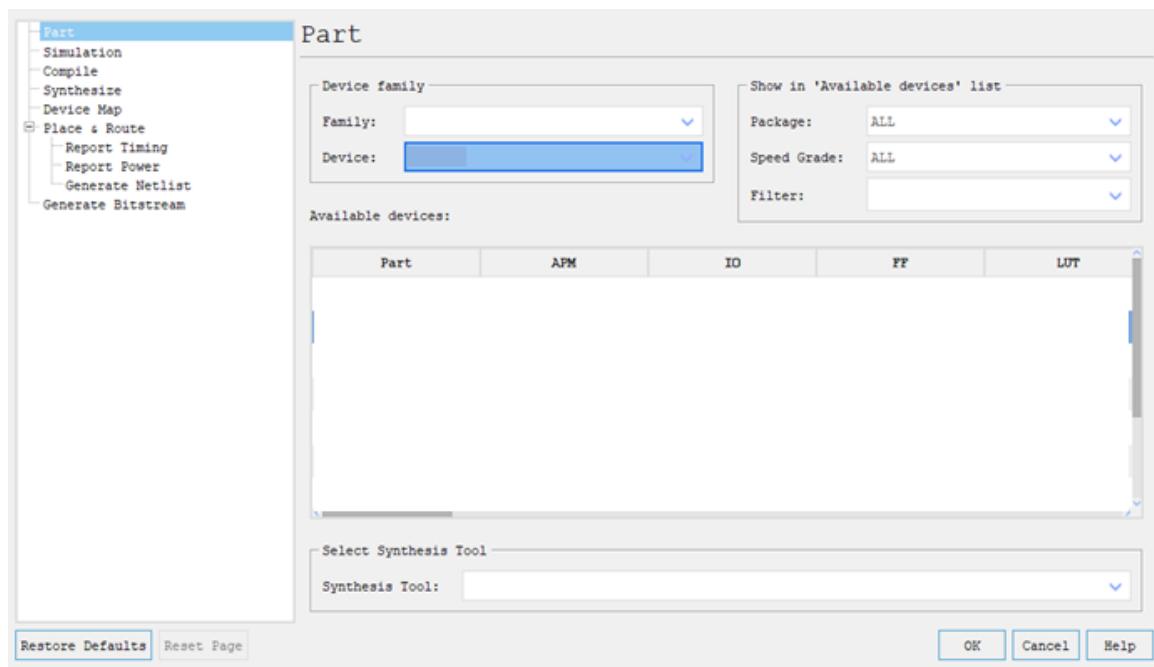


图 2-49 器件选择窗口

【Family】：选择器件系列；

【Device】：选择器件型号；

【Package】：选择封装类型；

【Speed Grade】：选择速度等级；

【Filter】：根据输入筛选器件；

【Synthesis Tool】：选择综合工具。

## 2.9 添加文件

### 2.9.1 添加源文件

单击下图中红色按钮

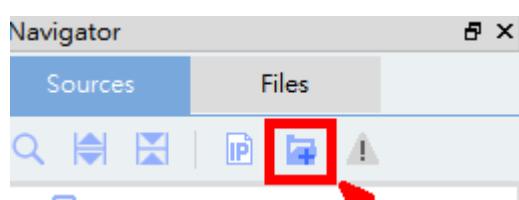


图 2-50 Sources 窗口中单击工具栏 Add Source

或在下图中红色区域右键，点击 Add Source 菜单

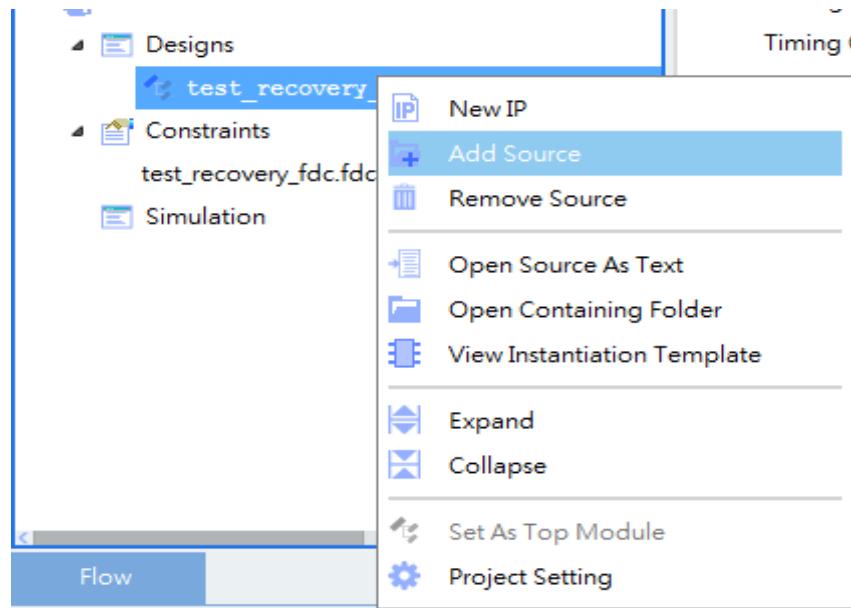


图 2-51 Sources 窗口中右键菜单中 Add Source

出现如下界面：

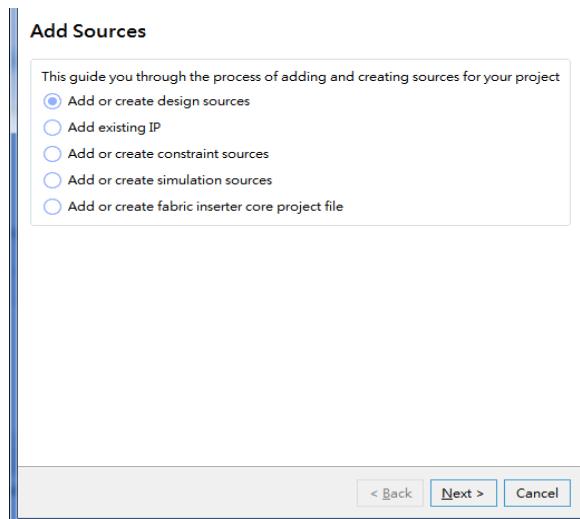


图 2-52 Add Source 界面

单击 Next，出现如下界面：

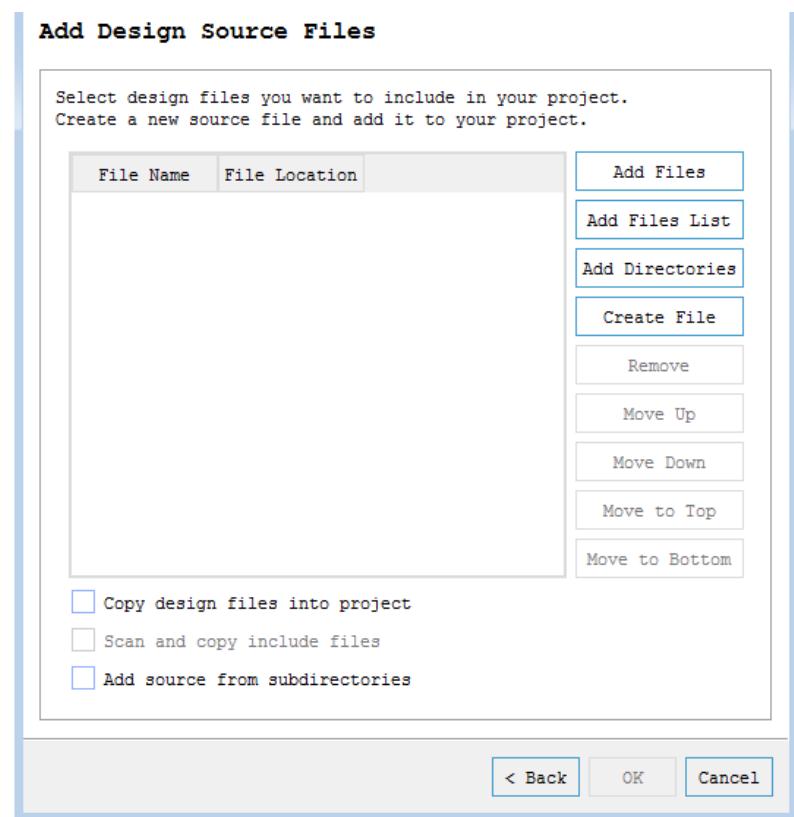


图 2-53 添加 design 界面

在添加文件时，可以通过 Add Files 找到对应的文件添加，也可以通过 Add Directories 来添加整个文件夹下的符合规则的文件，若勾选了下方的“Add source from subdirectories”则会搜索选中目录下的所有子目录下的符合规则的文件全部添加。如果勾选了“Copy design files into project”选项，那么程序会拷贝设计文件（add.v）到工程目录中的 source 文件夹，拷贝时，至少保留源文件所在的一级目录。设计文件添加完成后，将会看到如下图所示的软件界面：

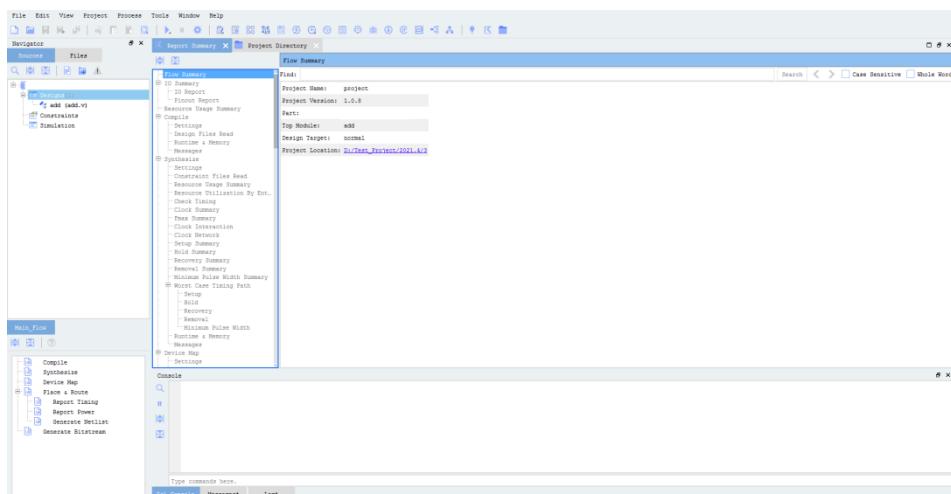


图 2-54 添加设计文件后的软件界面

如果添加文件失败，会出现如下提示框：

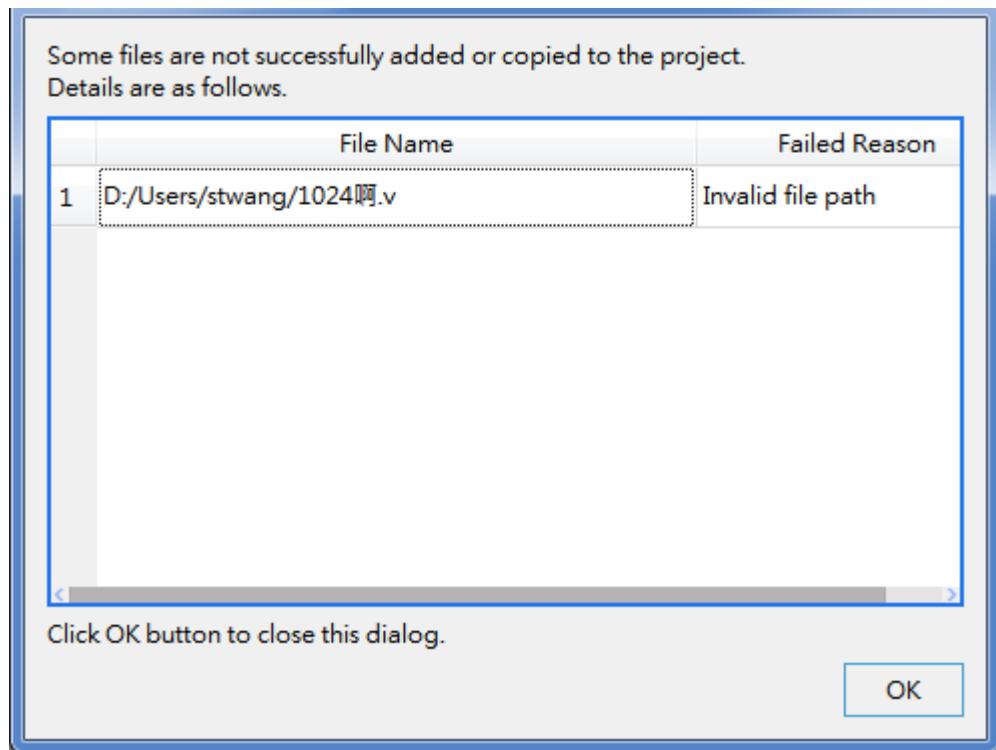


图 2-55 添加文件失败对话框

失败原因：

Invalid file path：文件路径太长或路径中含有非法字符（如汉字）

Copy file failed：复制文件失败

New file failed：新建文件失败

工程管理区【Entry】中【Files】里显示的是工程的输入文件，可以查看相应的文件路径和类型，右键菜单中有 edit file, open containing folder, show base name 等操作，其中 show base name 可以直接显示文件名，方便查看。

### 1) 在用户添加的文件里进行搜索

在用户添加的文件里搜索某关键词时，可以使用工具栏里的“Find In Files”工具进行搜索，如下图：



图 2-56 “Find In Files”工具栏位置

打开该工具后，界面如下图所示：

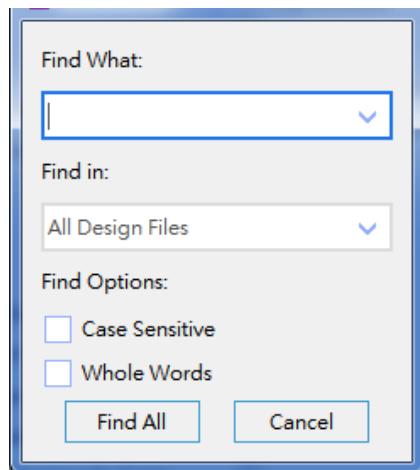


图 2-57 “Find In Files” 工具界面

其中，“Find in:”标题下面的下拉框可以指定搜索的范围，有三种选择范围，“Find Options:”标题下面两个复选框可以指定搜索的方式，包括大小写敏感，以及是否整字匹配，如下图所示：

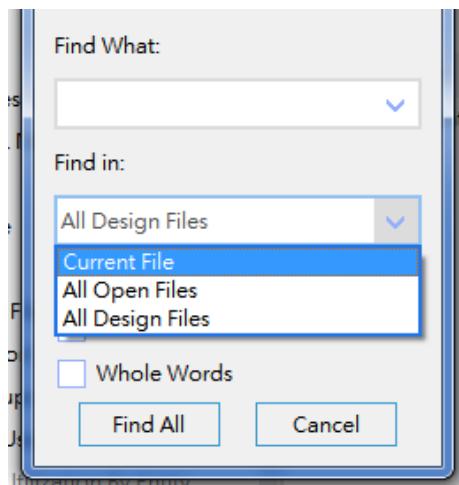


图 2-58 “Find In Files” 工具可搜索范围

可以看到，该工具一共有“Current File” / “All Open Files” / “All Design Files”三种搜索范围：“Current File”和“All Open Files”分别指的在文本编辑器里当前打开和所有打开的文件里搜索，“All Design Files”指的在所有用户添加的源文件里进行搜索，不管这些文件是否在文本编辑器里打开；

搜索的结果都会显示在主界面下面的 Console 窗口里，如下图所示：

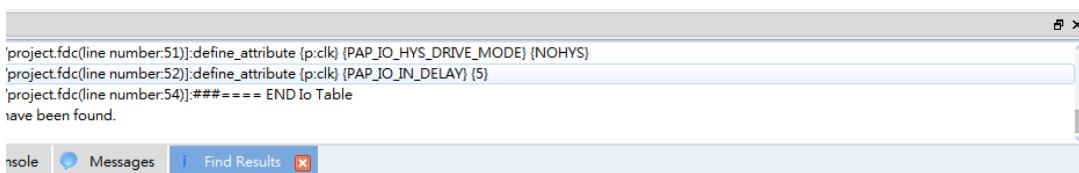


图 2-59 “Find In Files” 工具搜索结果

搜索结果里包含搜索到的含有关键词的文件的名字和行号，以及结果的数目。

## 2) IP 文件

在集成版工程中，在工程管理区【Sources】右键弹出右键菜单，点击 Add Source，在出现的对话框中选择添加 IP 按钮，选择下拉列表中的\*.idf 即可加入相应的 ip 文件。（如以下两幅图）

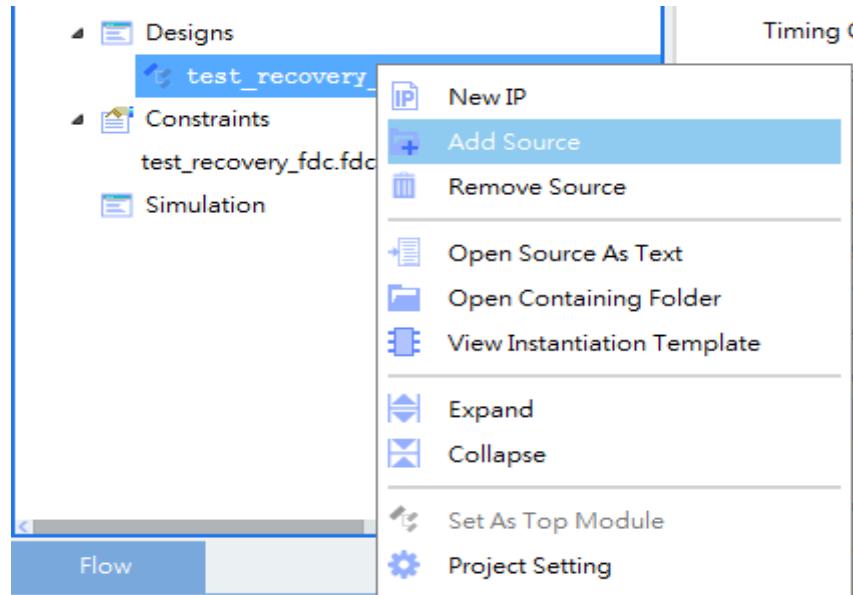


图 2-60 添加源文件右键菜单

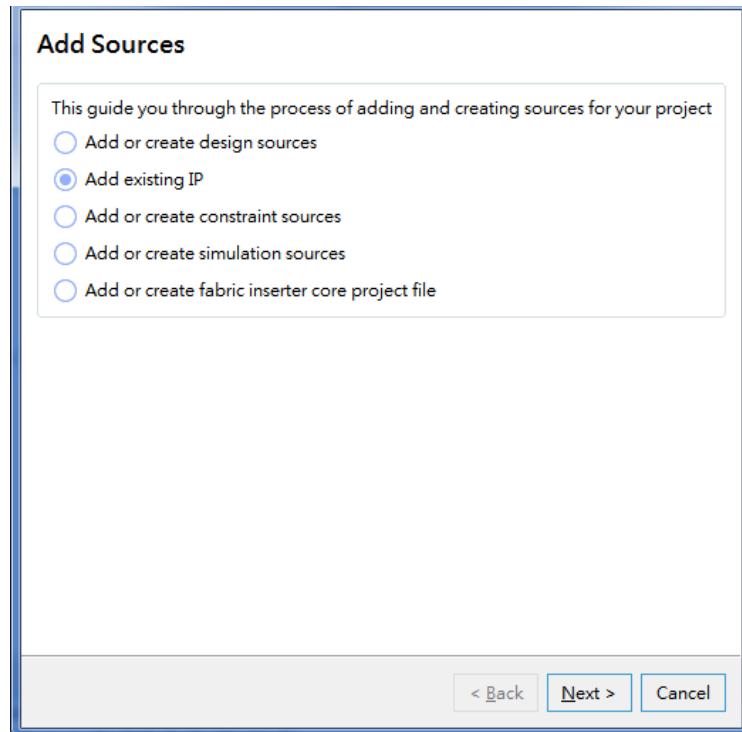


图 2-61 添加 ip 文件对话框

添加 IP 后主界面如下图所示：



图 2-62 添加 IP 后主界面

添加 IP 后，在 sources 界面会显示 IP 的 rtl 文件层级结构，但 rtl 文件只是显示在界面，并未添加到工程中。

IP 包含的所有 rtl 文件会自动传递给综合工具综合，用户不再需要将这些 ip 的 rtl 文件加入到 design 中。

hierarchy 支持标识出需要更新的 IP 实例，体现在 module hierarchy 前面的图标区分，可右键选中 ip 工程文件(可多选)，点击 Regenerate IP 直接更新 IP 文件内容，如下：

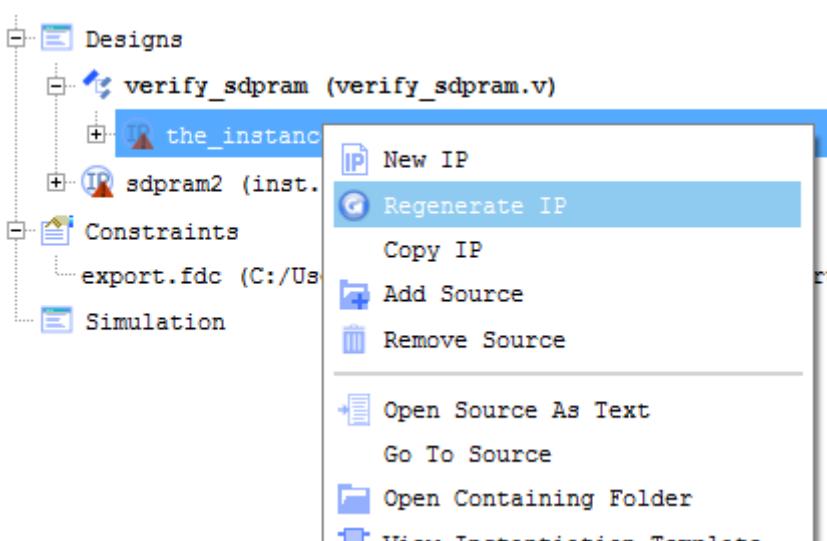


图 2-63 更新 IP 界面

右键选择 ip 工程文件，点击 Copy IP，可以拷贝该 ip 工程文件并添加到工程中。设置好 IP 名字和目标路径后可以执行拷贝操作，拷贝完成后可以对其进行配置，如下：

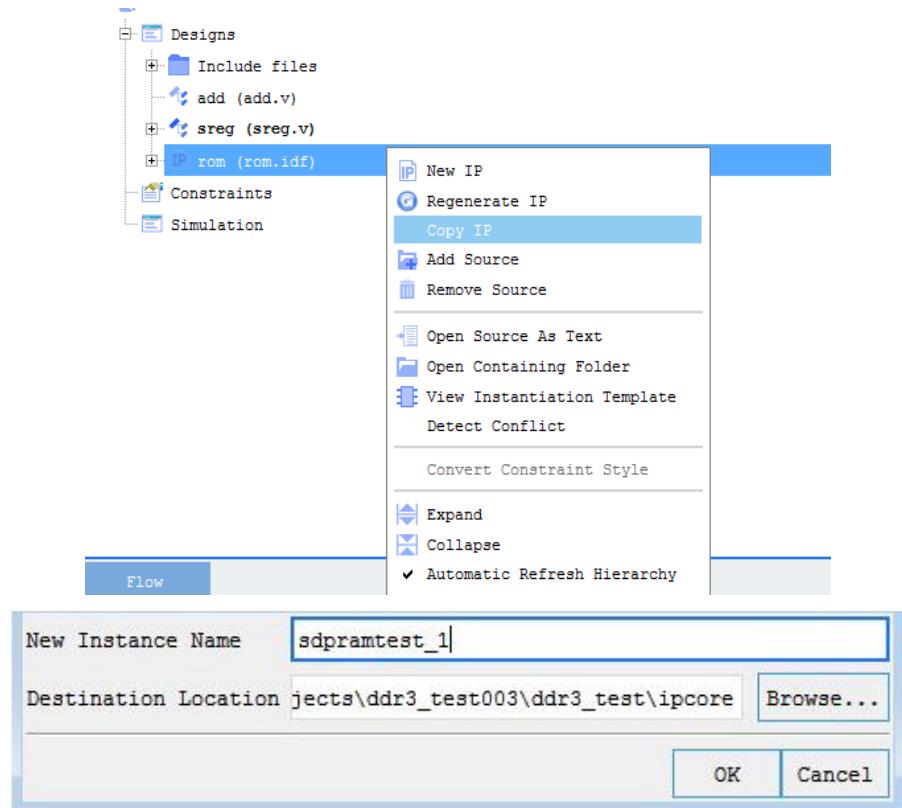


图 2-64 Copy IP 界面

### 2.9.2 新建源文件

前面操作与添加源文件类似，当出现如下界面时：

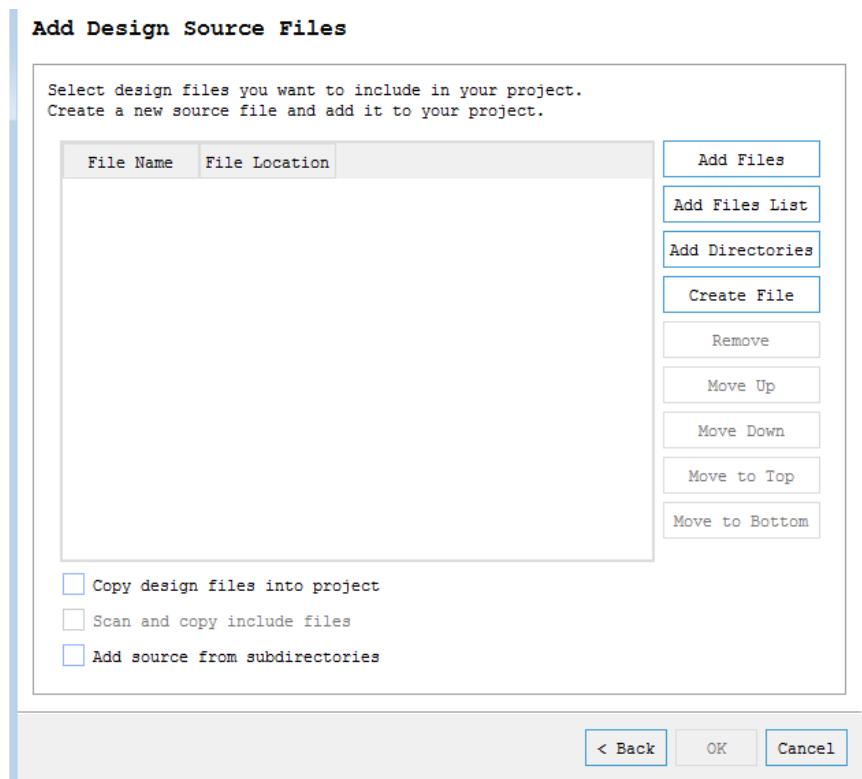


图 2-65 添加源文件主界面

单击 Create File 按钮，将出现如下界面：

点击上方下拉列表可以选择新建文件的类型：

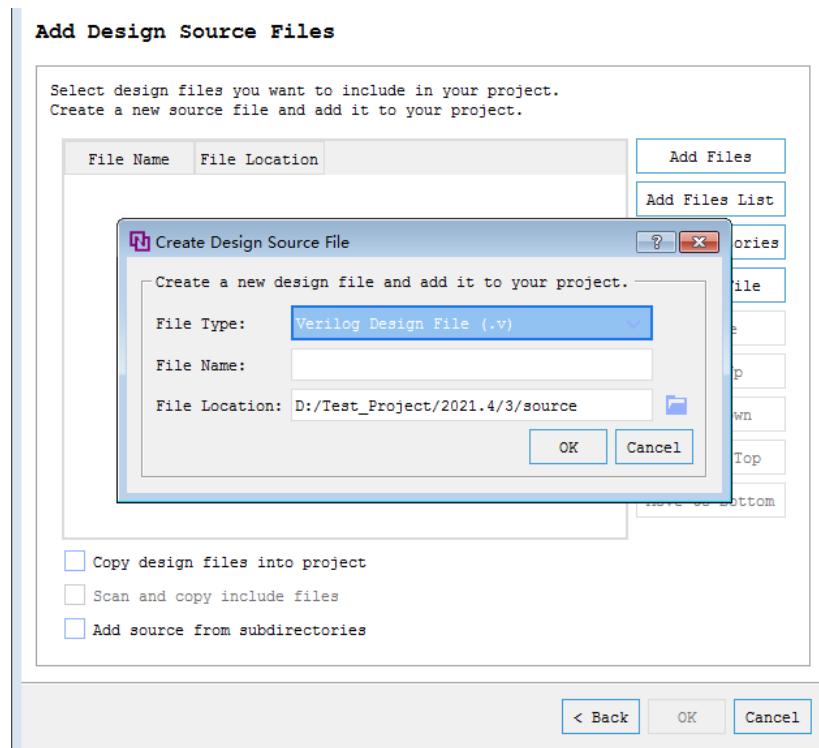


图 2-66 点击 Create File 后弹出界面

选择好文件类型之后（此处选择.V 文件），接着在下面的文本框中输入文件名，并选择文件路径：

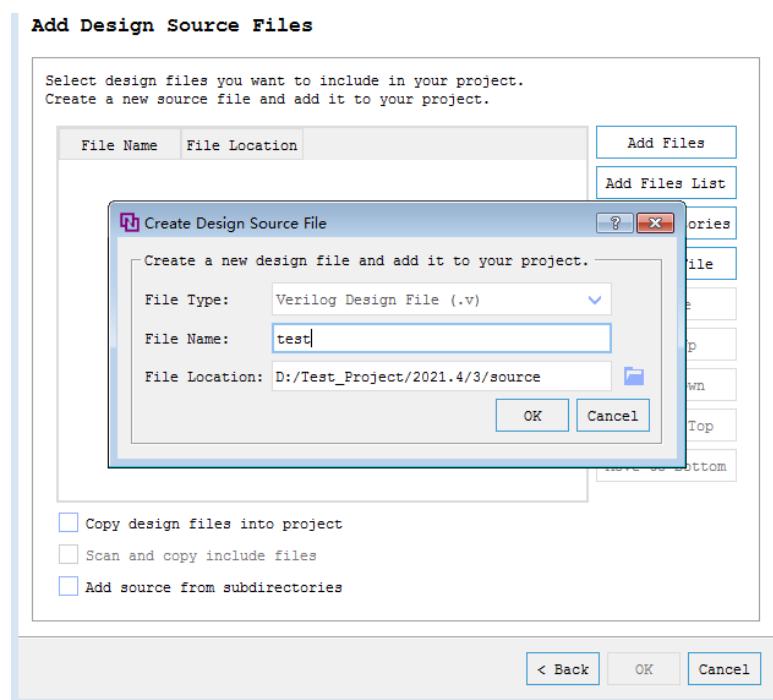


图 2-67 输入文件名界面

若文件名已经存在，则会弹窗提示，并询问是否要重命名如下：

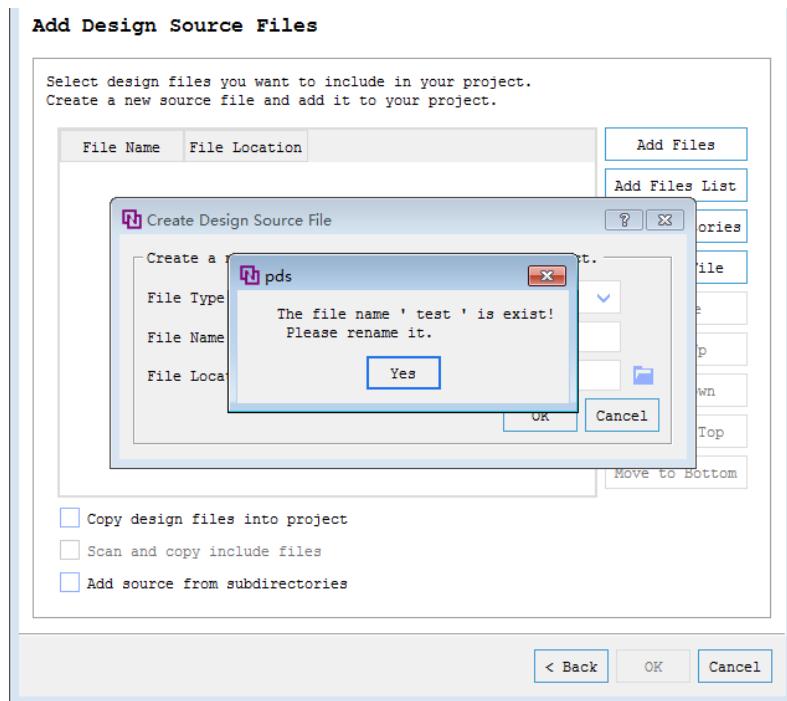


图 2-68 输入文件名重名提示界面

选择‘Yes’，则返回上面界面重命名。

单击OK之后出现如下界面：

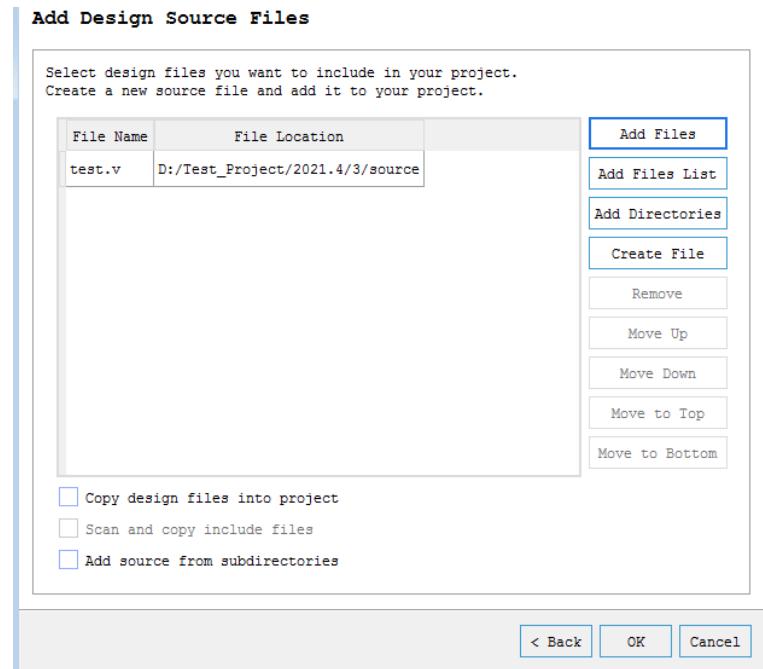


图 2-69 Create File 结束后界面

此处可以添加或者新建多个文件。添加或新建完成之后点击OK：

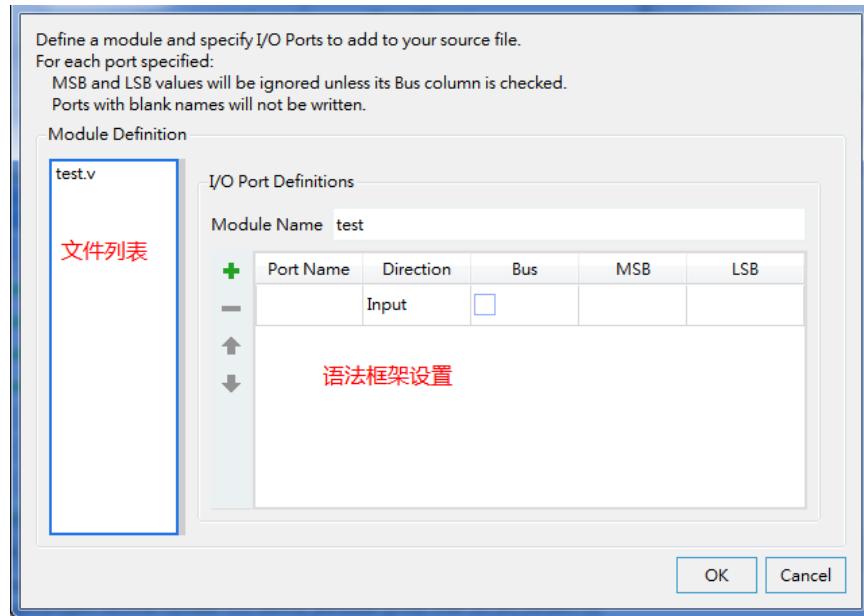


图 2-70 .v 文件语法框架设置界面

在文件列表中可以选择想要设置的文件，在语法框架设置区域可以按照需求进行设置：

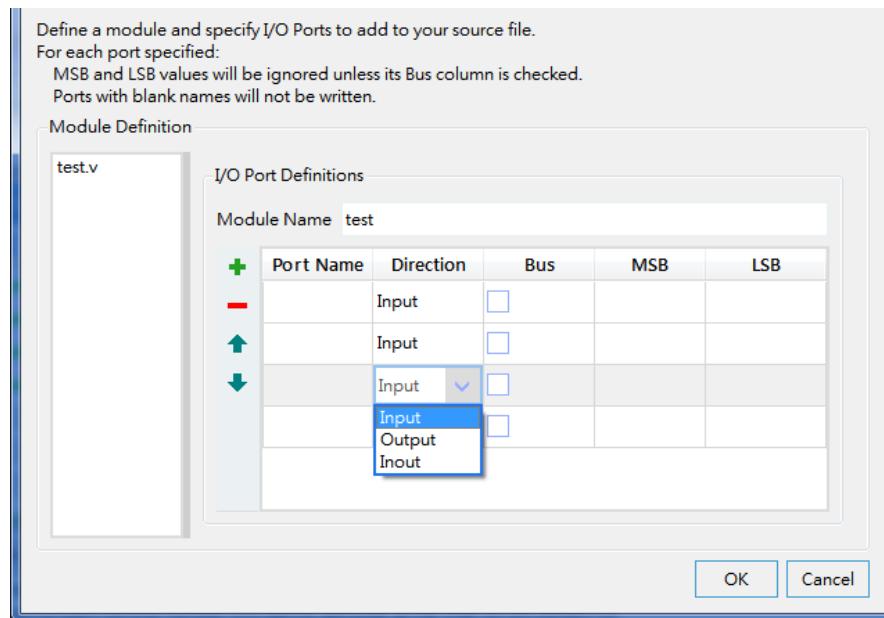
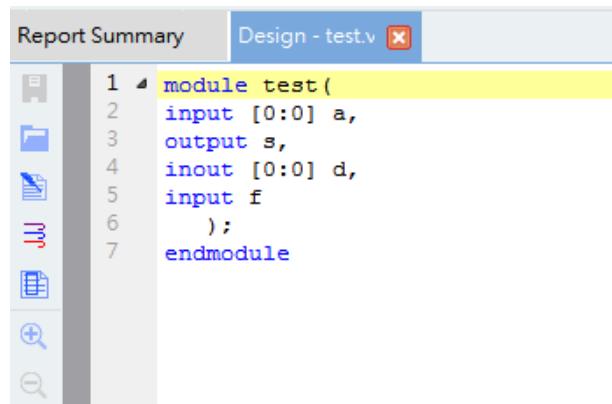


图 2-71 module 设置界面

设置完成之后，单击 OK，将打开新建好的文件：



```

Report Summary Design - test.v
1 module test(
2   input [0:0] a,
3   output s,
4   inout [0:0] d,
5   input f
6 );
7 endmodule

```

图 2-72 .v 文件设置好语法框架后生成界面

.VHDL 文件与.V 文件大体相同，仅在设置界面有所区别：

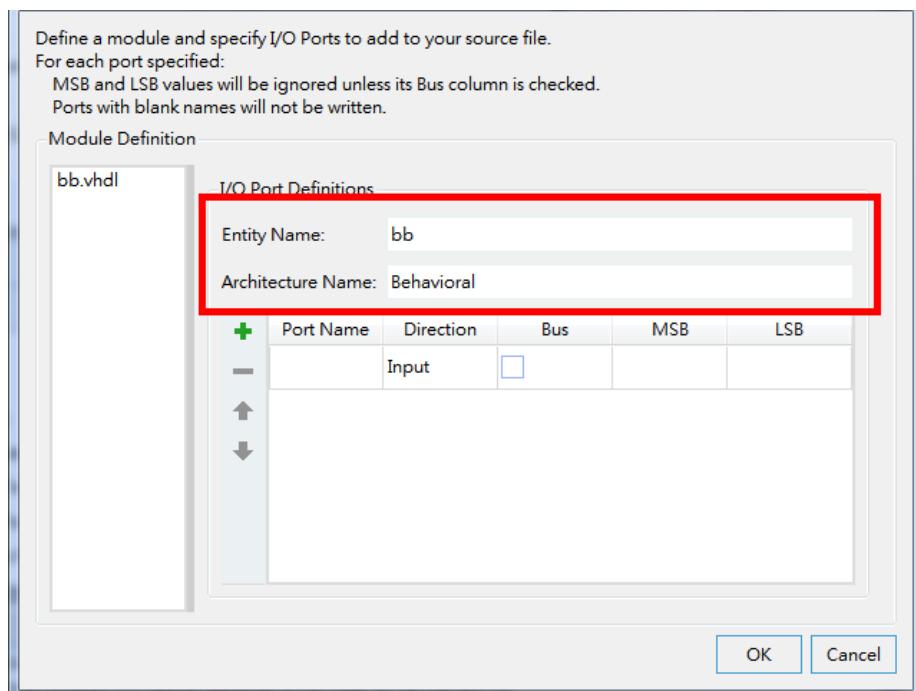
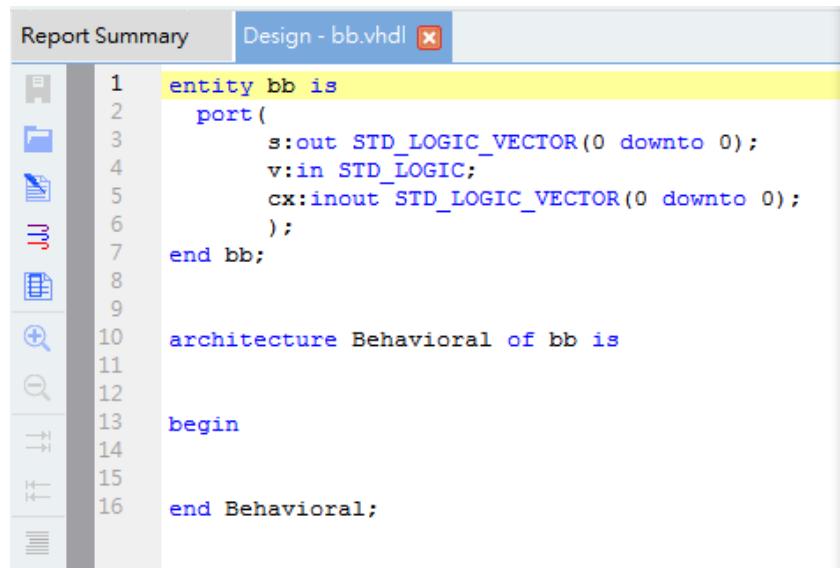


图 2-73 .vhdl 文件的语法框架设置界面

由于语法区别，因此在红色方框中的内容与.V 文件有所不同。最后生成的文件也同样如此：



```

1  entity bb is
2      port(
3          s:out STD_LOGIC_VECTOR(0 downto 0);
4          v:in STD_LOGIC;
5          cx:inout STD_LOGIC_VECTOR(0 downto 0);
6      );
7  end bb;
8
9
10 architecture Behavioral of bb is
11
12 begin
13
14
15 end Behavioral;
16

```

图 2-74 .vhdl 文件设置完成后界面

### 2.9.3 刷新 design hierarchy

在 source 窗口右键，弹出如下界面

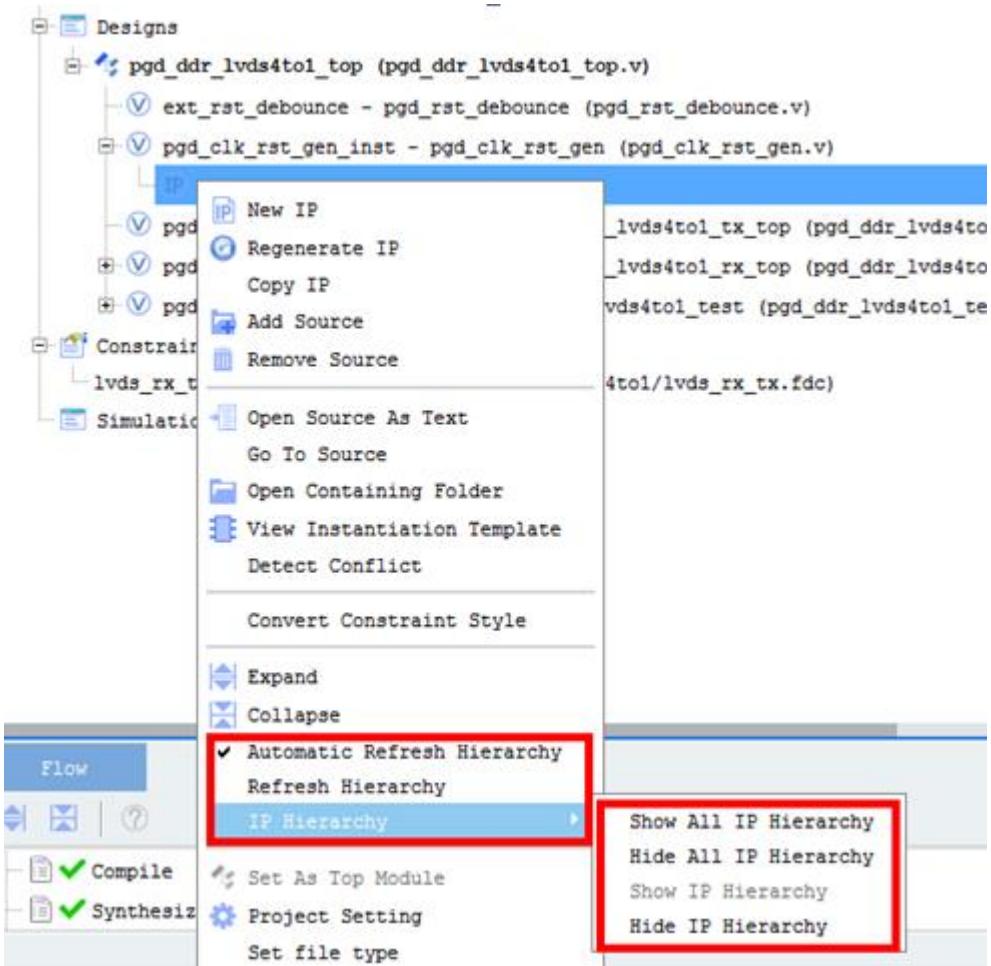


图 2-75 刷新 hierarchy

**【Automatic Refresh Hierarchy】：**当 design 文件存在增加、删除、修改操作的时候自动

刷新 design 的 hierarchy， 默认为勾选。

当取消勾选时，对 design 文件有新保存的时候，在 Sources 下弹出 “Hierarchy is out of date” 提示和 Reload 按钮，点击 Reload 按钮的时候刷新 design hierarchy。

**【Refresh Hierarchy】**：手动刷新 design hierarchy。

**【IP Hierarchy】**：控制是否显示 IP 的层级结构，包括以下四个功能：

- (1) Show All IP Hierarchy: 设置所有的 IP 的层级结构可展开
- (2) Hide All IP Hierarchy: 设置所有的 IP 的层级结构不可展开
- (3) Show IP Hierarchy: 展开选中的 IP 层级结构
- (4) Hide IP Hierarchy: 隐藏选中的 IP 层级结构

#### 2.9.4 Global include 设置

在 PDS 的 source 窗口右键菜单栏中增加 set global include 和 clear global include，对选中的文件进行相关操作；set global include 和 clear global include 入口在 OEM 流程和后端版不可见。

把目标文件设置成 global include，所有的设计文件都能使用目标文件中 define 定义的宏，这会导致设计文件的相关宏生效，从而产生不同解析结果。

把目标文件从 global include 中清除，所有没有 include 目标文件的设计文件的相关宏失效，会产生不同解析结果。

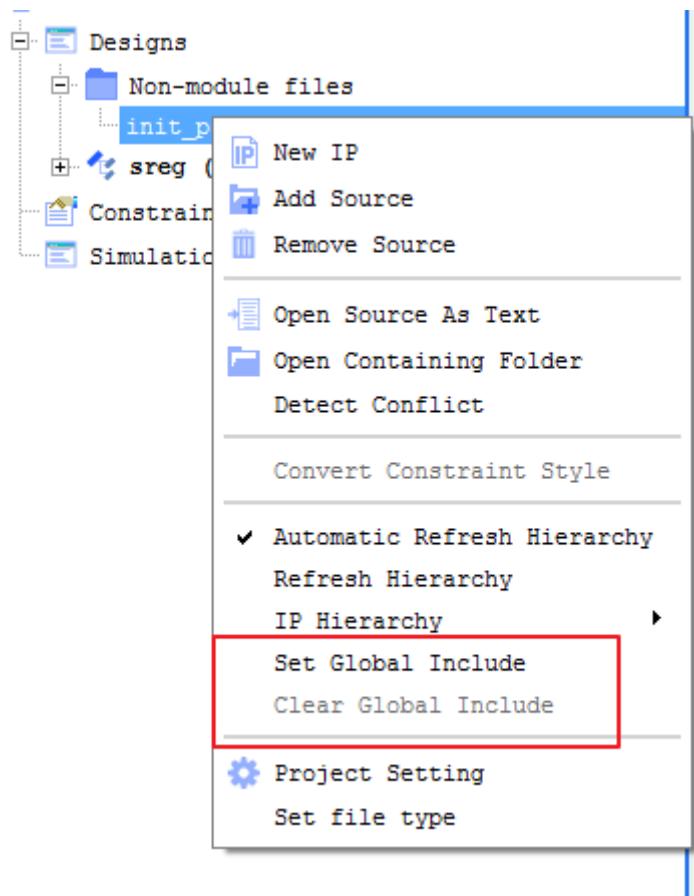


图 2-76 set Global include

### 2.9.5 添加约束文件

用户设计约束包括对时序的约束和对模块位置的约束，对应的约束文件分为两种，时序约束和物理约束。以下是 pds 支持的约束文件格式介绍。

**Sdc** 格式文件：该格式是时序约束文件格式，此格式文件只能在没有使用综合工具的情况下才可加载，里面记录的是 timing 相关的约束信息。

**Lcf** 格式文件：该格式是逻辑约束文件格式，此格式文件只能在没有使用综合工具的情况下才可加载，里面记录的是与逻辑约束相关的信息，如设置属性。Map 过后部分逻辑约束信息会转化为物理约束。

**Scf** 格式文件：该格式文件里面记录的是 timing 相关的约束信息，有无综合工具均可添加，作用于 dev\_map 阶段。

**Fdc** 格式文件：该格式文件中包含时序和逻辑约束，是只有在使用了综合工具的情况下才可以使用。

#### 1) 添加用户约束文件

单击下图中红色按钮

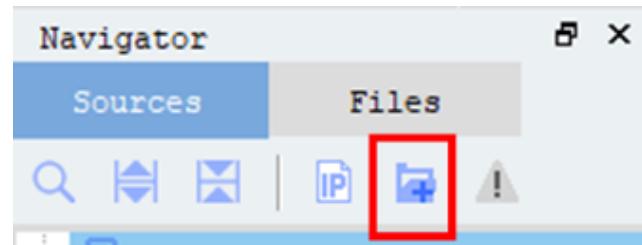


图 2-77 添加约束文件

或再下图中红色区域右键，点击 Add Source 菜单

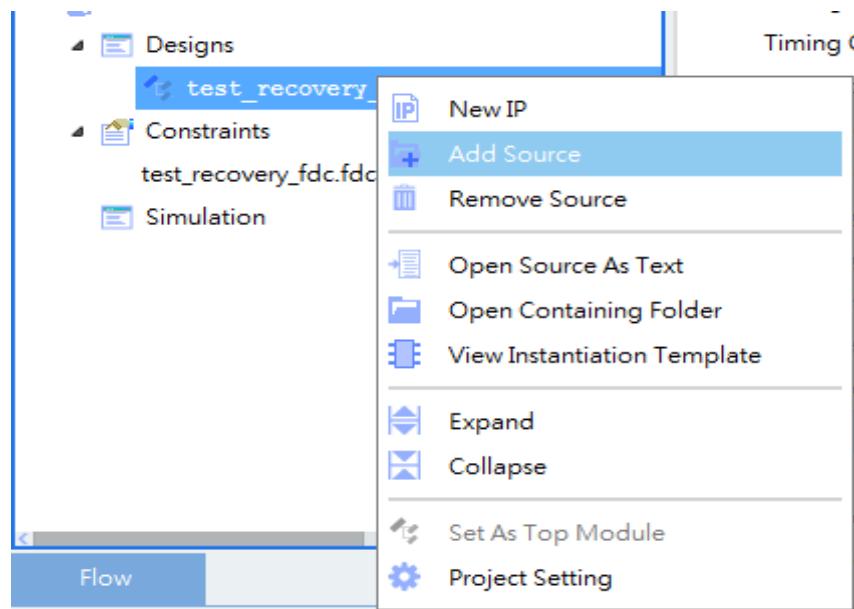


图 2-78 右键菜单添加源文件

出现如下界面：

### Add Sources

This guide you through the process of adding and creating sources for your project

- Add or create design sources
- Add existing IP
- Add or create constraint sources
- Add or create simulation sources
- Add or create fabric inserter core project file

< Back [Next >](#) Cancel

图 2-79 添加源文件界面

选择第三项 Add or create constraint sources，如下图：

### Add Sources

This guide you through the process of adding and creating sources for your project

- Add or create design sources
- Add existing IP
- Add or create constraint sources
- Add or create simulation sources
- Add or create fabric inserter core project file

< Back [Next >](#) Cancel

图 2-80 选择添加约束界面

点击 Next，出现如下界面：

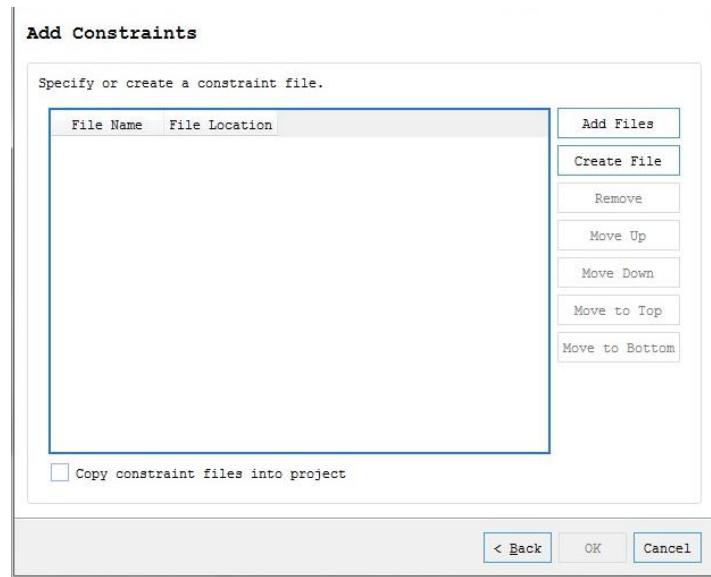


图 2-81 添加用户约束文件

选择好约束文件或新建约束文件后，点击 **OK** 即可加入工程。同样，当勾选“**Copy constraint files into project**”选项时，程序会拷贝约束文件到工程目录中的 source 文件夹。

约束文件（.sdc 文件）格式如下：

命令格式： `create_clock -name clock_name -period clock_period -waveform { 0 wave_period } [ get_ports { clock_port_name } ]`

其中 `clock_name` 是要被约束的时钟端口（design 中的 `clock_port_name`）对应的时钟名字（时序分析中）； `clock_period` 是要被约束时钟的周期； `wave_period` 是要被约束时钟的振幅（一般占空比为 50%），其值为时钟周期的一半（`wave_period=clock_ period/2`）； `clock_port_name` 要被约束的时钟端口（design 中）。

注意：`clock_name` 和 `clock_port_name` 可以是不同的，但在创建时钟约束时，推荐 `clock_name` 取 `clock_port_name` 相同的名字，使得 `clock_name` 和 `clock_port_name` 相同，方便对应和查看。

建议：（特别是针对多时钟约束时）在约束每个时钟前加上“#Begin clock constraint”、结尾加上“#End clock constraint”以示不同时钟约束间的区别。

实例：现要对 design 中的时钟端口 `clk` 进行约束，可以创建一个名为 `clk` 的时钟，其周期为 6.929ns，占空比为 50%，则 sdc 文件可编辑为：

#Begin clock constraint（此语句可省略）

`create_clock -name clk -period 6.929 -waveform { 0 3.4645 } [ get_ports { clk } ]`

#End clock constraint（此语句可省略）

特别地，当时钟名字中存在“[ ]”时（对多位时钟进行取位操作），应用“{}”将整个时钟名字均括起来；

实例：现要对 design 中的时钟端口 clk[0]进行约束，可以创建一个名为 clk 的时钟，其周期为 6.929ns，占空比为 50%，则 sdc 文件可编辑为：

```
#Begin clock constraint (此语句可省略)  
create_clock -name {clk[0]} -period 6.929 -waveform { 0 3.4645 } [ get_ports { clk[0] } ]  
#End clock constraint (此语句可省略)
```

## 2) 添加物理约束文件

物理约束文件通过 Physical Constraint Editor 窗口添加。打开 Physical Constraint Editor 窗口需要运行完 Device Map。所以实际操作添加物理约束文件可以放在 Device Map 运行完成后。

Physical Constraint Editor 窗口可以通过 Tools 菜单下的 Physical Constraint Editor (Post-Map) 打开，或者直接通过工具栏中的 Physical Constraint Editor (Post-Map) 快捷方式打开。菜单栏 Tools 下的二级菜单如下图所示：

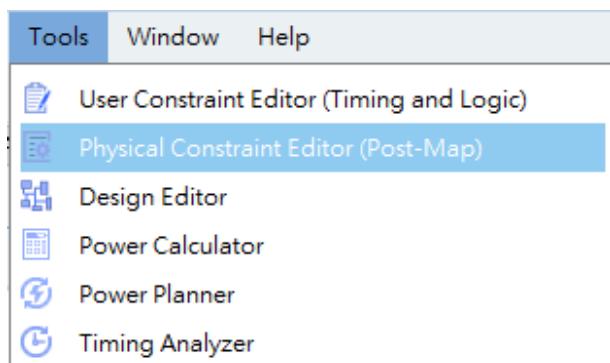


图 2-82 Tools 菜单下 Physical Constraint Editor (Post-Map)

在 PDS 中点击上述菜单，会弹出 Input Physical Constraint File 窗口。该窗口中，Current pcf file 指示当前工程中的 pcf 文件，Delete pcf from project、Select a pcf to edit 等选项提供删除、编辑、添加和替换 pcf 文件的操作。

添加物理约束文件的窗口如下图所示：

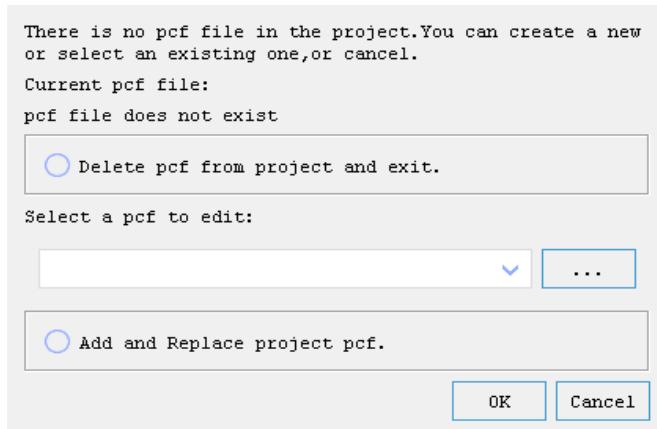


图 2-83 添加物理约束

点击 OK 按钮，将开始启动 Physical Constraint Editor (PCE)。打开后的 Physical Constraint Editor 界面如下图所示：

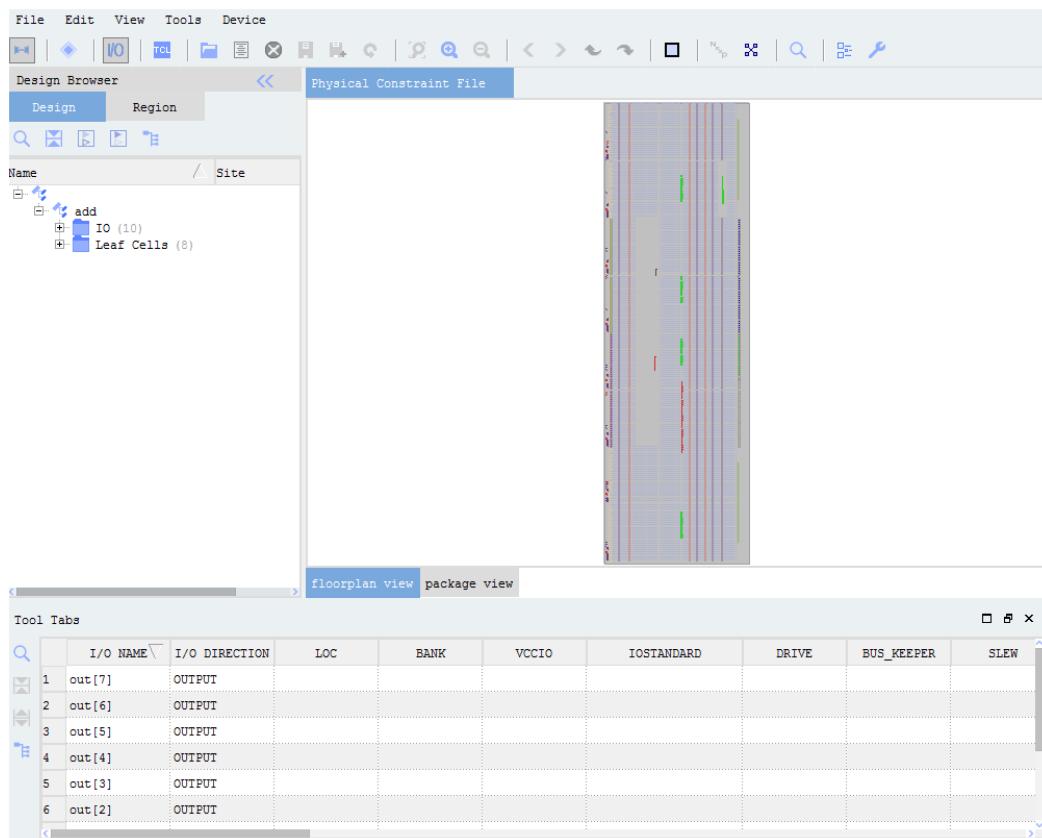


图 2-84 添加物理约束后的 Physical Constraint Editor 界面

物理约束文件添加完成后，关闭 Physical Constraint Editor 界面即可回到主界面。

物理约束可分为管脚约束和布局约束两种类型，软件中提供了管脚约束和布局约束命令，均支持编写物理约束文件和在 Constraint Editor 中图形操作生成相应的物理约束文件。

### i. 管脚约束

#### a. 编辑管脚约束文件

管脚约束文件，能够很好地支持标准形式（不对管脚属性进行设置）约束，而管脚属性（可选项）进行设置功能也逐步完善。管脚约束有两种格式，包括 def\_port 命令和 def\_inst\_site 命令，且两者仅是格式头（def\_port 和 def\_inst\_site）不一样，其余格式均一样。

#### def\_port 命令

```
def_port port_name LOC=pin_loc
[IOSTANDARD=io_standard_name]
[SLEW=SLOW/FAST]
[DRIVE=2/4/6/8/12/16/24]
[PULLDOWN=TRUE/FALSE]
[PULLUP=TRUE/FALSE]
[KEEPER=TRUE/FALSE]
[IODELAY=NONE/BOTH/IBUF/IFD]
```

其中 port\_name 是(design 中)要被约束的 IO 端口的名字； pin\_loc 是芯片上的一个 pin 脚资源位置； io\_standard\_name 是各种端口标准的名字； SLEW 的可取值是 SLOW 和 FAST； DRIVE 的可取值是 2、4、6、8、12、16 和 24； PULLDOWN、PULLUP、KEEPER 的可取值都是 TRUE 和 FALSE； IODELAY 的可取值是 NONE、BOTH、IBUF 和 IFD。该命令中，方括号内的都是可选项。其中，对输入端口， SLEW 和 DRIVE 选项不可用；对输出和双向端口， IODELAY 选项不可用。

#### 实例 1：

一个 counter 的用户设计中，有个时钟输入端口名为 clkin，将其约束到 A11 的管脚上。则使用标准形式（不含可选项）约束，内容如下：

```
def_port clkin LOC=A11
```

注意：对输入端口， SLEW 和 DRIVE 选项不可用；对输出和双向端口， IODELAY 选项不可用。

#### 实例 2：

一个名为 counter 的用户设计中，有个时钟输入端口名为 clkin，将其约束到 A11 的管脚上，并对其管脚属性进行设置。内容如下：

```
def_port clkin LOC=A11 IOSTANDARD=LVTTI IODELAY=NONE
```

若某个输出端口名称为 LED1，将其约束到 B17 的管脚上，并对其管脚属性进行设置。

内容如下：

```
def_port LED1 LOC=B17 IOSTANDARD=LVCMOS18 SLEW=FAST DRIVE=16
```

### **def\_inst\_site 命令**

```
def_inst_site port_name pin_loc
```

其中 port\_name 是(design 中)要被约束的 IO 端口的名字； pin\_loc 是芯片上的一个 pin 脚资源位置，支持 IOB、PIN、PAD 三种描述方式；

#### 实例 1：

一个 counter 的用户设计中，有个时钟输入端口名为 clkin，用 IOB 描述方式将其约束到 IOB\_9\_42 的管脚上。内容如下：

```
def_inst_site clkin IOB_9_42
```

#### 实例 2：

一个 counter 的用户设计中，有个时钟输入端口名为 clkin，用 PIN 描述方式将其约束到 P1 的管脚上。内容如下：

```
def_inst_site clkin P1
```

#### 实例 3：

一个 counter 的用户设计中，有个时钟输入端口名为 clkin，用 PAD 描述方式将其约束到 PAD1 的管脚上。内容如下：

```
def_inst_site clkin PAD1
```

### **b. 图形操作生成管脚约束文件**

软件中提供了底层 IO 约束的功能模块 package，当软件运行完 map 后，已经得到了底层的映射网表，可以直接操作这些底层 IO，实现对底层的 IO 操作。

软件中提供了图形化的管脚约束功能，管脚约束可由 package view 或 floorplan view、I/O Table 和 Design Browser 共三类视图上操作完成，其中 I/O Table 视图可单独操作，而 package view 或 floorplan view 和 Design Browser 视图必须配合使用，两种方式的最终结果是一致的，只是约束的操作形式不一样而已。另外 package view 或 floorplan view 视图中的 IO 类型的 instance、Design Browser>>IO 视图中 io、I/O Table 视图中的 io 三者均是一一对应的。

用户可以通过在视图中进行拖放的方式，进行管脚约束。将 Design Browser>>IO 视图中 io 拖放到 package View 中的 pin 脚上或 floorplan view 中的 IOB 上，也可以在 I/O Table 视图填写 IO(I/O Name)的约束位置 (LOC)，来完成引脚约束。如下图所示：

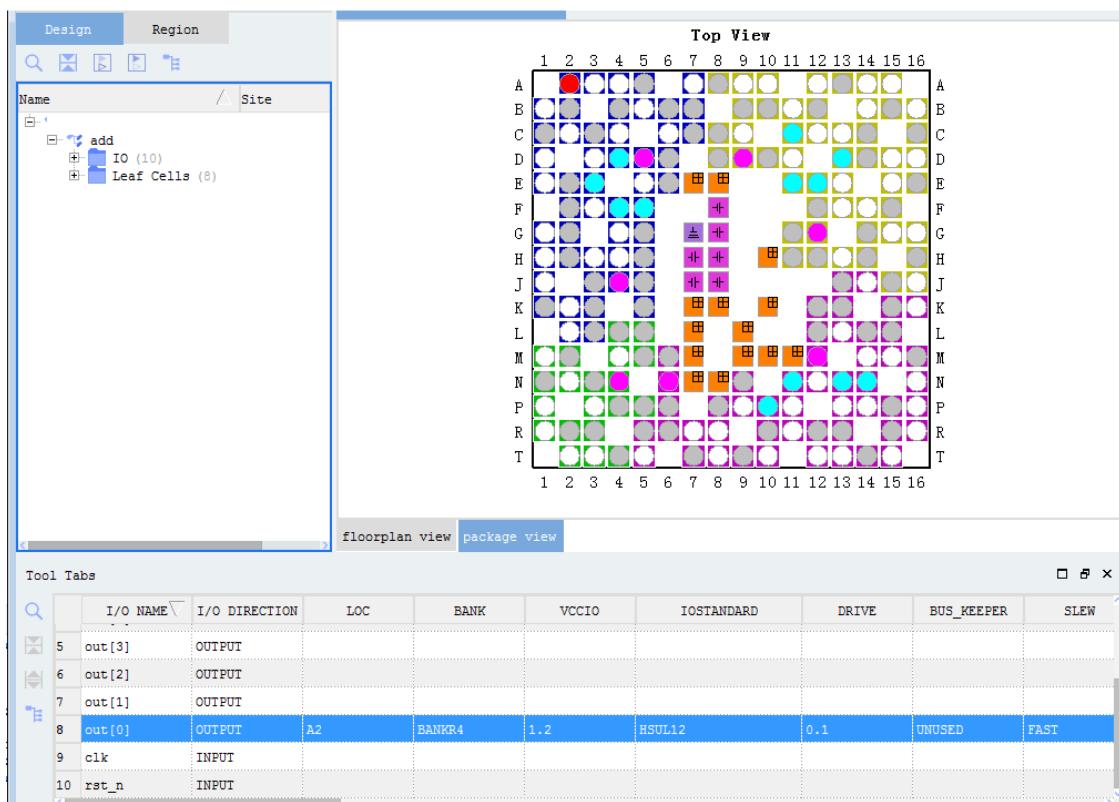


图 2-85 管脚约束视图

注意：无论通过哪种方式来完成引脚约束，所有视图均会按相应的规则反映管脚约束的情况。

## ii. 布局约束

### a. 编辑布局约束文件

编辑布局约束文件一般应用 def\_inst\_site 命令，格式如下：

命令格式：def\_inst\_site inst\_name site\_name [ impl\_name ]

其中 inst\_name 是要被约束的设计实例的名字，若其中的设计实例为 IO 类型，则其和管脚约束一致（这也是管脚约束和布局约束的联系）； site\_name 是 FPGA 芯片中器件资源的名字，如 CLMA\_74\_14, MUL\_71\_22 等； impl\_name 是可选项，表示对应的设计实例必须使用的基本物理单元（如 CLMA 中的 FGA,FGB,FF1 等）。

实例 1：

假设 TempInst 是设计网表中一个类型为 LUT4 的实例，想把它放到 CLMA\_77\_38 上。则命令为：

```
def_inst_site TempInst CLMA_77_38
```

由于 LUT4 在一个 CLMA 上有 FGA 和 FGB 两种实现方式，如果还想指明用 F 来实现 TempInst。则命令为：

```
def_inst_site TempInst CLMA_77_38 F
```

实例 2:

假设 TempFF 是设计网表中一个寄存器类型的实例，想把它放到 CLMA\_77\_35 上，并且用 FGA 来实现它。命令为：

```
def_inst_site TempFF CLMA_77_35 FGA
```

实例 3:

假设 TmpMem 是设计网表中一个 BRAM 类型的实例，想把它放到 BRAM\_71\_39 上。则命令为：

```
def_inst_site TmpMem BRAM_71_39
```

实例 4:

假设 TmpMul 是设计网表中一个 MUL 类型的实例，想把它放到 MUL\_71\_38 上。则命令为：

```
def_inst_site TmpMul MUL_71_38
```

### b. 图形操作生成布局约束文件

软件中提供了底层单元布局的功能模块 floorplanner，当软件运行完 map 后，生成底层的映射网表，可以直接操作这些底层单元，实现对底层的布局操作。

布局约束由 floorplan View 和 Design Browser 两个视图操作完成，用户可以通过拖放的形式把 Design Browser>>Instance 视图中的 instance 拖放到 floorplan View 视图中，就能将 instance 约束到芯片合适的（资源）位置上。如下图所示：

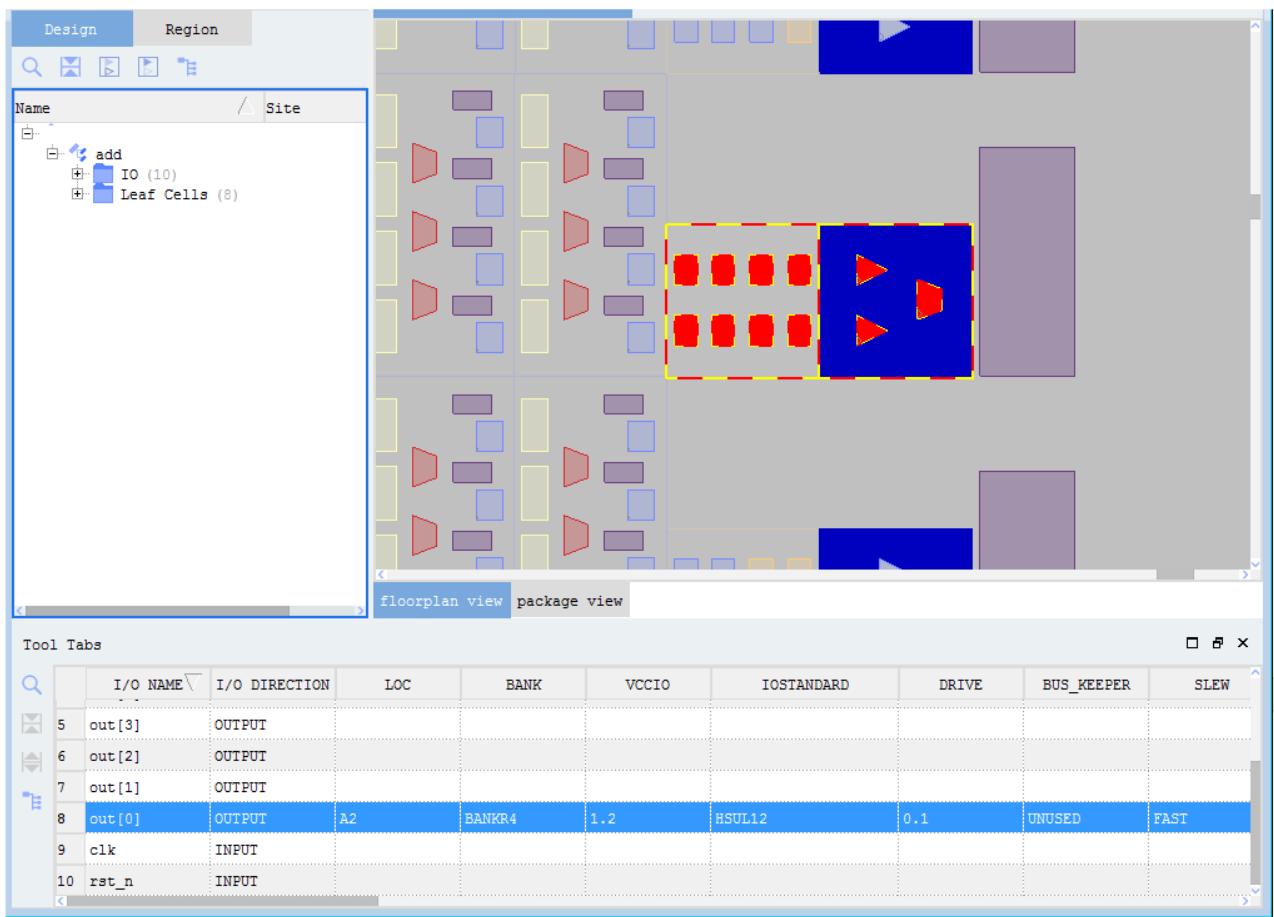


图 2-86 布局约束视图

### iii. 区域约束

区域约束其实是布局约束的变形，即不直接布局约束到某一个具体的物理单元上，而是预先指定一个区域，然后软件根据实际情况自动布局约束到指定的区域中的任意物理单元上。

区域约束共有 2 种命令形式，包括 `def_region` 命令和 `def_inst_region` 命令。其中 `def_region` 命令是用来定义在当前 FPGA 芯片中的布局区域；`def_inst_region` 命令用来将当前设计中的设计实例约束到某个区域中。

#### a. 定义区域

##### `def_region` 命令

区域约束需要遵循，先声明后使用的原则，即：先定义区域后使用该区域。比如先通过 `def_region` 命令创建区域，再通过 `def_inst_region` 命令将 instance 放入区域中。

```
def_region region_name xmin xmax ymin ymax
```

其中 `region_name` 是区域的名字，方便以后引用；`xmin, ymin` 是区域左下角 XY 坐标，`xmax, ymax` 是区域右上角 XY 坐标。在 FPGA 芯片中，每一个物理单元都有一个坐标，这个坐标可以在 floorplan View 视图中可以看到。如下图所示：

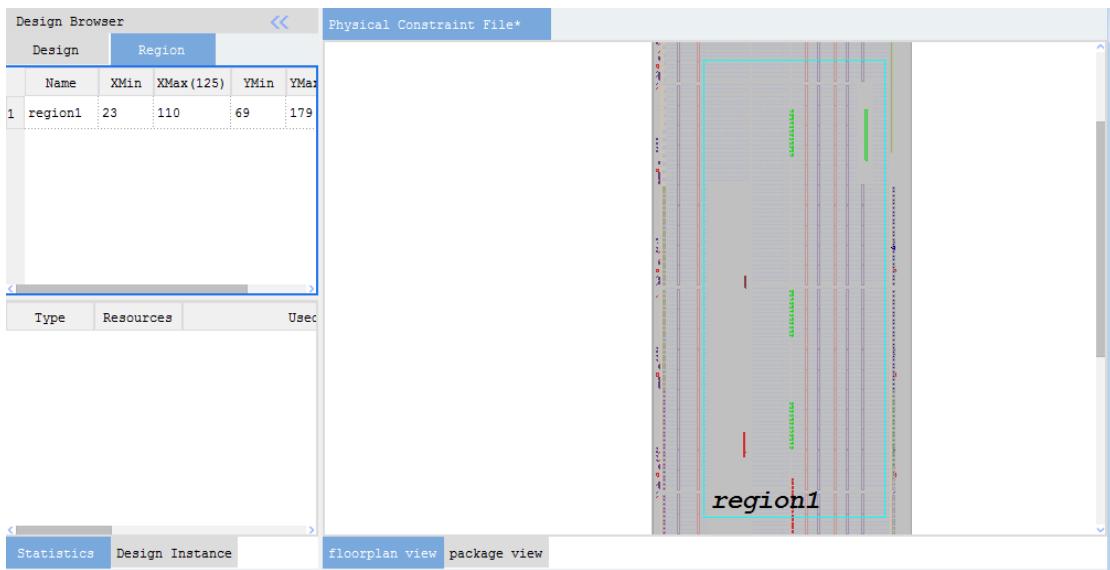


图 2-87 布局约束视图

当把鼠标放到一个 CLMA 单元里的空白处，鼠标提示中给出 CLMA 单元的名字 CLMA\_5\_133 和它的坐标 (4,44) ，这个坐标就是 def\_region 命令中所使用的坐标。

实例：

定义一个长方形区域 reg1, 该区域的图形显示左下角为 CLMA\_190\_11, 坐标为 (62,5) ; 长方形右上角为 CLMA\_201\_12, 坐标为 (65,6) 。

用 def\_region 命令定义区域的命令为：

```
def_region reg1 62 65 5 6
```

### b. 约束设计实例

def\_inst\_region

使用该命令时，用户需要在该命令之前先定义该命令，方才可以使用，命令格式如下：

```
def_inst_region inst_name region_name
```

其中 inst\_name 是网表文件中要被约束的设计实例的名字， region\_name 是一个区域的名字（被上面两个命令定义）。该命令在布局的时候将实例 inst\_name 放到区域 region\_name 里。

实例：假设 TmpInst 是设计网表中一个类型为 LUT4 的实例，想把它放到上两个命令定义的 reg1 中。命令为：

```
def_inst_region TmpInst reg1
```

## 2.10 移除文件

对于添加的文件，可以采用选中该文件，点击右键，通过弹出的右键【Remove Source】

菜单移除该文件。如下图所示

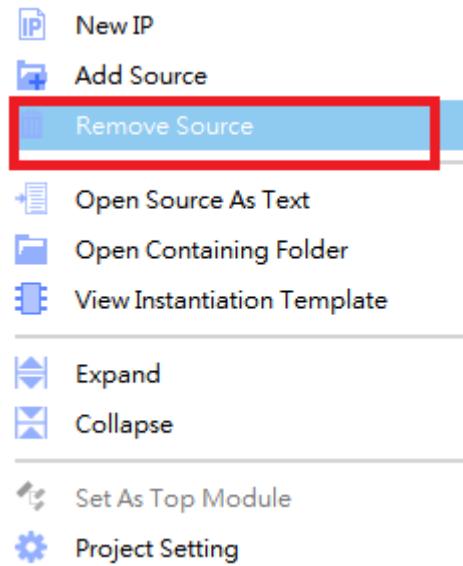


图 2-88 文件移除菜单

点击【Remove Source】将弹出如下图所示对话框，如果勾选“Also delete the project local file/directory from disk”，删除的文件将从工程中删除后，并将磁盘中的文件一并删除。

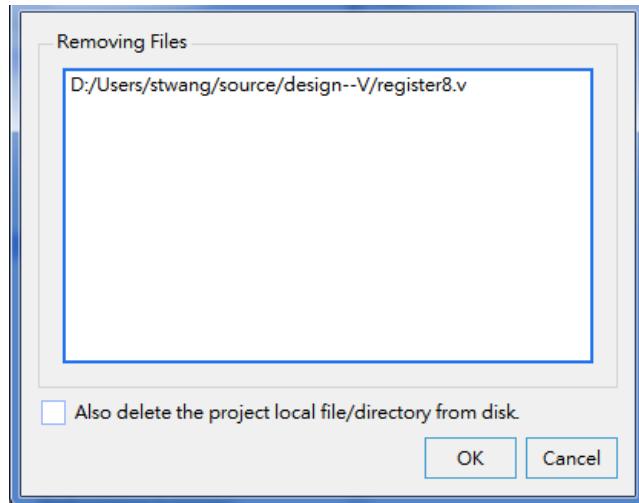


图 2-89 文件移除对话框

在图 2-85 中选中 design source 文件，右键下拉菜单中除了移除文件的操作外，还有其他功能选项。功能介绍如下：

**New IP:** IP 文件添加；右击下拉菜单选中 New IP，弹出 New IP 对话框，完成 IP 的添加。

**Add Source:** 添加 source 文件或 constraint 文件；右击下拉菜单中选择 Add Source 或 Project->Add Source 打开 Add Sources 对话框添加文件；

**Open Source As Text:** 使用 PDS 内置的文本编辑器打开文件；

**Remove Source:** 删除 source 或 constraint 文件；选中 source 或 constraint 文件，右击下拉

菜单中选择 Remove Source，删除文件。

Open Containing Folder: 打开文件所在的目录。

Expand All: 展开所有文件。

Collapse All: 文件折叠显示。

Set As Top Module: 设置 top module; 选中 Designs 结构中 source 文件的 module，右击下拉菜单选择 Set As Top Module，可以将选中的 module 设置为 top module。

物理约束文件的移除通过打开 Physical Constraint Editor 窗口的 Delete from project 选项移除。如下图所示：

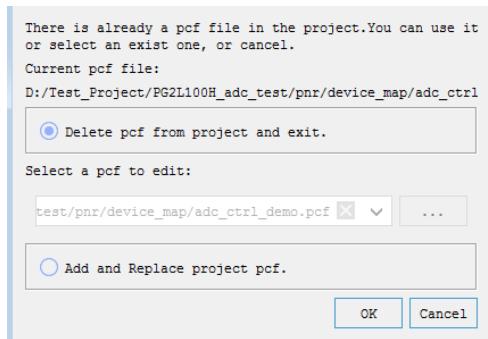


图 2-90 pcf 文件移除

## 2.11 参数设置

若要进行参数设置，则需打开软件 flow 的选项配置窗口【Project Setting】，然后才能进行具体的设置。一般有两种方式可以打开【Project Setting】。

a. 点击 Project 菜单中的 Project Setting，可以进行参数设置。如下图所示：

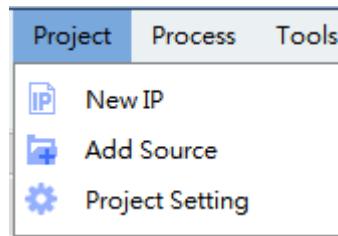


图 2-91 Configure 菜单

b. 在工程管理区的右键菜单中，选择 Project Setting 也可以进行参数设置。如下图所示：

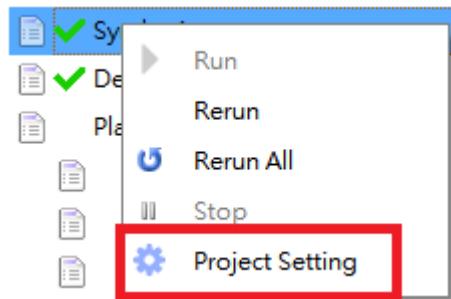


图 2-92 工程管理区 configure 菜单

打开软件 flow 的选项配置窗口【Project Setting】，则如下图所示：

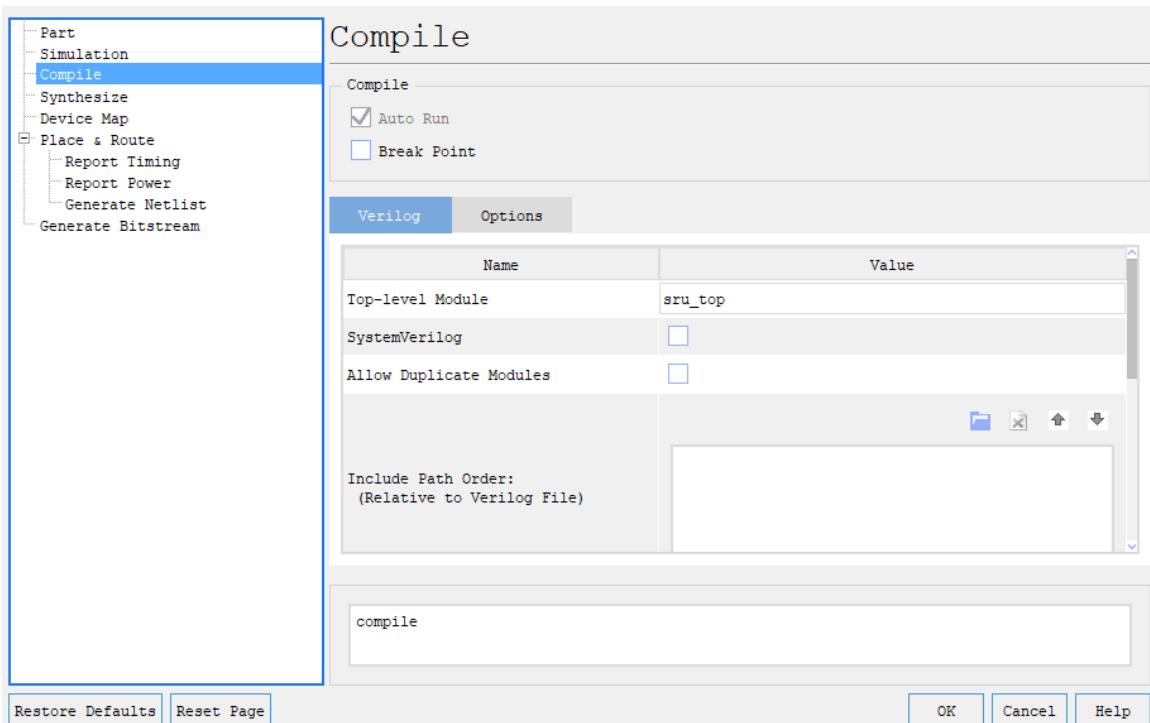


图 2-93 选项配置窗口

## 2.12 报告系统

软件提供了较为完整的报告机制：如下图所示，图中显示了报告系统的 Report Summary。Report Summary 在主页上优先显示了比较常用且重要的信息，报告系统对软件产生的信息进行分类并且在 Report Summary 中提供了相关链接。

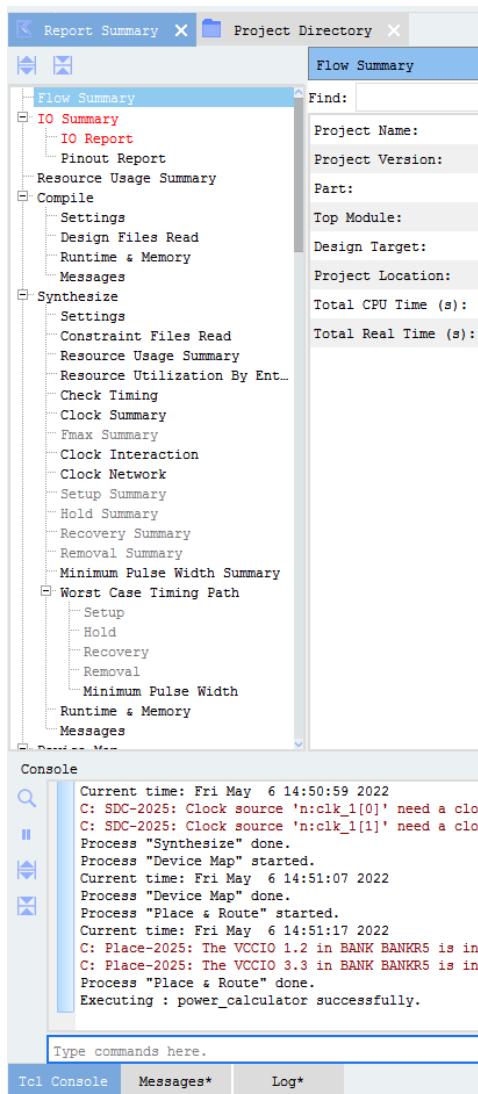


图 2-94 报告 Report Summary

**Synthesize Report** :综合后报告，主要是介绍综合过程使用的器件资源信息。

**Device Map Report** :设备映射报告，主要设备映射有关的信息。

**Place & Route Report** :布局布线报告，主要是布局布线使用的器件资源和布局布线所用时间信息。

**Report Timing Report** :布局布线后时序报告，主要是介绍与布局布线有关的时序信息。

**Report Power Report** :布局布线后功耗报告，主要介绍与布局布线有关的功耗信息。

**Generate Bitstream Report** :位流报告，主要是介绍与位流有关的信息。

操作运行完成后，以上介绍的重要报告均可以点击链接打开相应报告，除可以点击链接打开的报告之外，软件还提供了以文件描述的报告，这些报告文件都放在软件的实现路径中。

其中修改设计文件保存后，在 PDS 界面的 Report Summary 中还是能够阅读之前运行完成所生成的报告，在重新开始编译后，之前运行完成所生成的报告将置灰，不能阅读。

其中 Flow Summary 下面的“Project Location”链接用于打开工程文件所在目录的文件浏览器。打开 PDS 工程的时候会默认打开 Report Summary 右边的 Project Directory 列表下图。双击列表中对应的文件可以在 PDS 的文本编辑器内打开该文件。

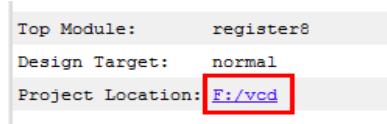


图 2-95 “Project Location” 链接

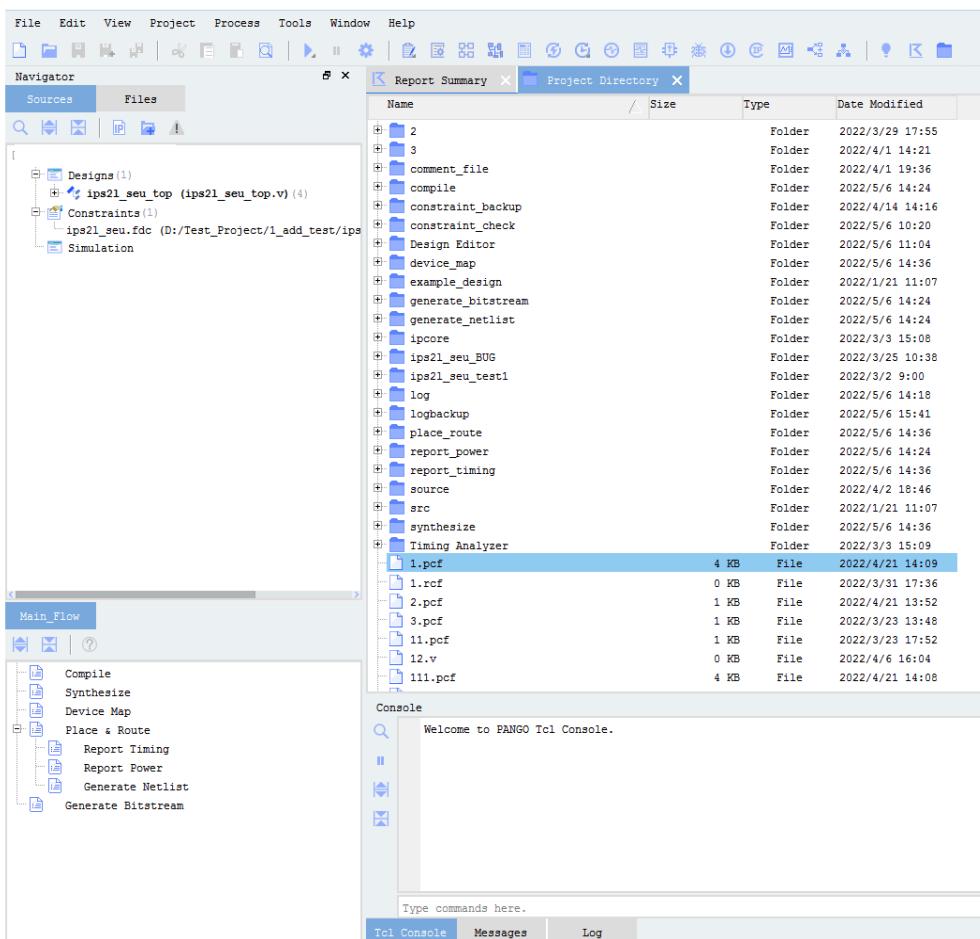


图 2-96 “Project Directory” 界面

## 2.13 文本编辑工具

软件提供了默认的文本编辑功能，如下图所示。

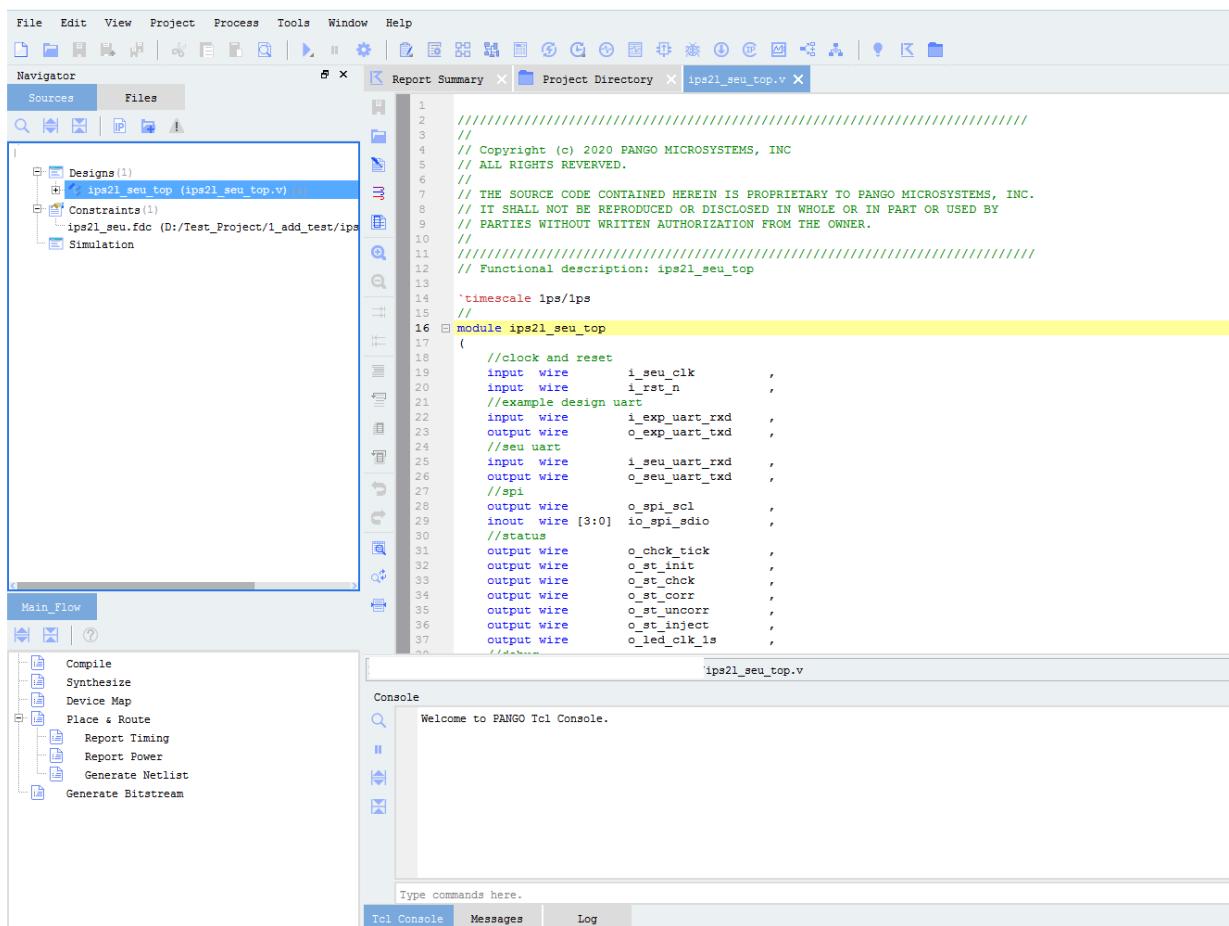


图 2-97 软件提供的默认文本编辑工具

如上图中左侧工具栏中提供了一些常用的编辑操作。

**Save:** 保存文件。

**Open Containing Folder:** 打开文件所在目录；点击工具栏按钮，可以打开文件所在的目录。

**Read Write:** 文本可读写；点击工具栏按钮，可以激活 read only 模式，激活后不能对文本进行编辑，在 read only 模式下再点击该按钮，即可取消 read only 模式。

**No Wrap:** 超出换行；默认情况下，如果不输入换行，编辑器不会给换行显示，点击工具栏按钮，超出了编辑区域就会自动换行显示。

**Column Selection Model:** 进入列选择、列编辑模式。

**Zoom In:** 放大；点击工具栏按钮或者使用快捷键 Ctrl+鼠标滚轮向上滚动，可以对文本字体放大显示。

**Zoom Out:** 缩小；点击工具栏按钮或者使用快捷键 Ctrl+鼠标滚轮向下滚动，可以对文本字体缩小显示。

**Increase Indent:** 向右缩进文本；选中文本中的某行，点击工具栏中的  按钮，实现文本向右缩进，相当于 Tab 键。

**Decrease Indent:** 向左缩进文本；选中文本中的某行，点击工具栏中的  按钮，实现文本的向左缩进。

**Comment Line:** 注释文本；选中文本中的某行，点击工具栏中的按钮或者使用快捷键 Alt+C，可以将该行注释。

**Uncomment Line:** 取消行注释；选中文本中被注释的行，点击工具栏中的  按钮或者使用快捷键 Shift+Alt+C，可以取消行注释。

**Comment Selection:** block 注释；选中文本中某块内容，点击工具栏中的 (Comment Selection)，可对选中的内容添加/\* \*/注释。

**Uncomment Selection:** 取消 block 注释；选中有/\* \*/注释的块，点击工具栏中的 (Uncomment Selection)，就会消掉选中部分的/\* \*/注释。

**Undo:** 撤销修改；点击工具栏中的  按钮，可以撤销上一次的修改操作。

**Redo:** 恢复修改；点击工具栏中的  按钮，可以恢复上一次的修改操作。

**Find and Replace:** 查找和替换；点击工具栏中的按钮或者使用快捷键 Ctrl+R，弹出查找和替换对话框对文本进行查找和替换。如下图中输入查找的 name 和替换后的 name，点击 Replace 按钮可以逐一替换，点击 Replace All 可以将全部 top 替换为 out，选择 Replace All 替换完成后会弹出提示框显示替换的位置和结果。可以通过 Match Whole Word、Case Sensitive 和 Wildcard Character（通配符匹配）选项卡对搜索条件进行设置。

**Line Search:** 行号搜索；点击对话框中的按钮或使用快捷键 Ctrl+G，弹出如下对话框中输入行号，点击 OK 可跳转到相应的行，并高亮显示。

除上述工具栏功能外，右键菜单栏还提供了以下功能：

**快捷键 CTRL+D:** 复制光标所在行的内容，并另起一行将其粘贴。

**快捷键 CTRL+L:** 剪切光标所在行的内容。

**书签功能:** 在编辑器的左侧灰色区域左键单击可以设置书签，方便来回切换位置。

**折叠功能:** 对于 verilog 格式的文件，支持以 module 为单位进行折叠展开。

**Check Syntax 功能:** 支持右键点击 Check Syntax 查看非当前 Top Module 设计文件的语法错误信息。

括号匹配：编辑器支持“（”、“[“、“{“的匹配，即在编辑器中鼠标选中上述括号符的附件，编辑器会自动标识出与之对应的括号间的文件。

修改标识：编辑器支持标识文本的修改痕迹，即文本如果有修改，会在编辑器的左侧显示红色的标识，保存后会变成绿色标识。

除了软件提供的默认文本编辑工具，通过 Edit 菜单下的 Preferences 可以指定第三方文本编辑功能，如下图。

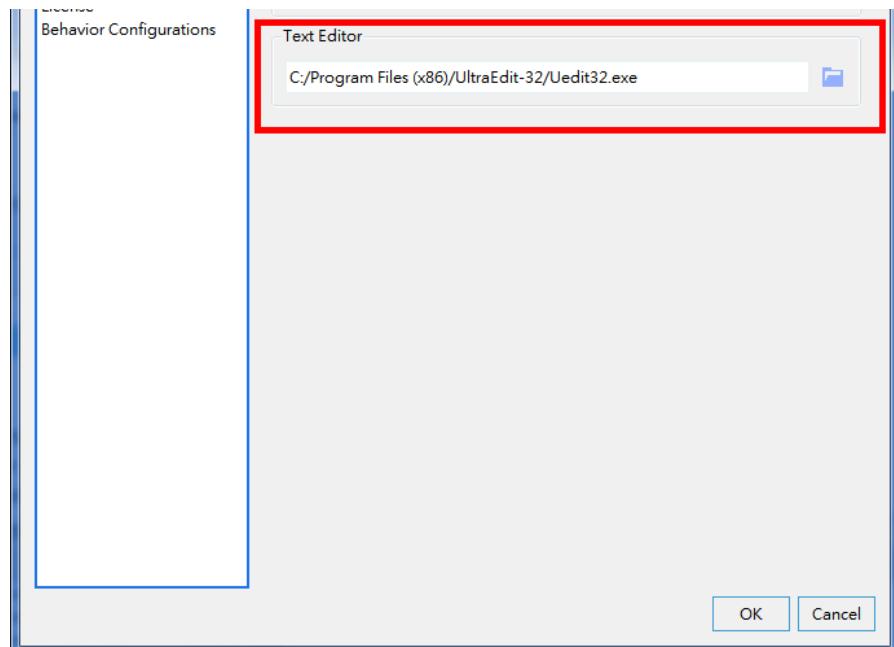


图 2-98 指定文本编辑工具

如上图中指定了 Ultra Editor 作为文本编辑工具，这样用户下次打开文件时就是使用 UE 打开和编辑了。

## 2.14 Messages 管理

PDS 软件运行过程中生成的 messages 显示在 Messages 窗口，在该窗口可以查看各个 action 运行阶段生成的 messages，还可以对 messages 进行按等级过滤、隐藏、切换 message 等级操作。



图 2-99 Messages 窗口

Errors、critical\_warning、warning、infos 前的复选框勾选时，显示勾选等级的 message 信息，取消勾选则不显示。

Hide All: 在 Messages 窗口隐藏所有的 message 信息。

Hide IP: 在 Messages 窗口隐藏 IP 相关的 message 信息，需要显示 IP 的 message 时可以再次点击 Show IP 按钮。

切换 message 等级: 在 Messages 窗口选中一条 message，右键有 Switch to Error, Switch to Warning, Switch to Information 选项，可以根据需要执行等级切换。切换等级后的 message 右键有 Reset Current, Reset All 选项，可以将 message 等级重置到默认值。

Manage Message Level : 对 message 进行等级切换后，点击此按钮，可以显示已切换 message 等级的具体信息，并可以有选择的进行等级重置。

### 3 开发流程

通过前两章的介绍，可以对 PDS 软件有一个基本的了解，接下来以一个实例进行简单的开发，演示软件的开发过程。

Pango Design Suite 软件一般的开发流程，主要包括七个步骤：

1. New Project（新建工程）；
2. Add Design/Constraints（添加设计和约束）；
3. Compile（编译）；
4. Synthesize（综合）；
5. Device Map（映射）；
6. Place & Route（布局布线）；
7. Design Editor（查看布局布线和时序分析结果，以及手动布局布线）；
8. Generate Bitstream（配置位流）。

下面用一个简单的实例说明 Pango Design Suite 的开发流程，如何用 Pango Design Suite 软件生成配置位流文件。由于步骤 1 和 2 已经在前面章节介绍了，所以这里我们直接以第 3 步 Compile 开始介绍。

以 add.v 为例，并对设计进行时序约束和物理约束为例进行说明。实例中选用 C:\pango\PDS\_2021.3\example\add\add.v。在该文件夹中还有一个 add.sdc（时序约束文件）和 add.vm。

以 Logos 系列下的器件 PGL50H，对设计进行时序约束和物理约束为例进行说明。

注意： 1. 实例中的基本设置一般均采用默认情况；

2. 实例中的所有配置选项均用默认设置。

3. 实例中采用 ADS 为综合工具；PDS 支持的综合工具包括 ADS 和 Synplify\_Pro，

相关的文档可见本文档第四章相关插件中的 4.1 ADS 和 4.2 Synplify\_Pro。

#### 3.1 Compile

Compile\_settings 介绍如下：Compile 将用户的 verilog 设计代码编译成 RTL 网表。

在主界面的 Compile 处右键，弹出如下图所示右键菜单：

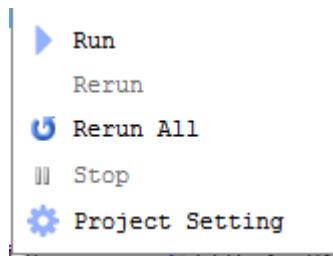


图 3-1 Compile 右键菜单

点击右键菜单中的 Project Setting 会弹出如下界面：

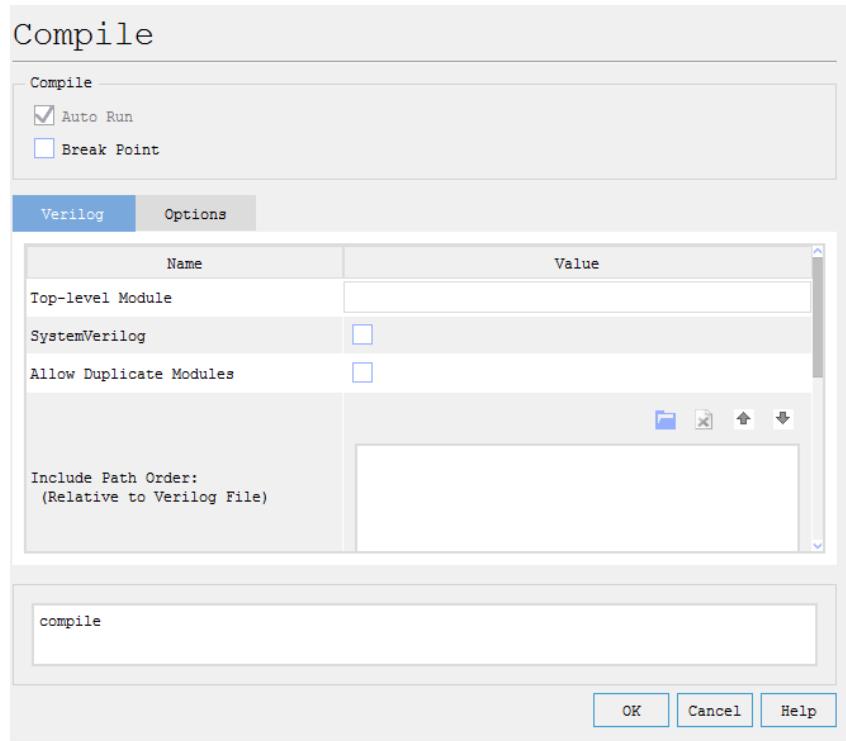


图 3-2 Compile 的选项设置

Compile 的配置选项详情可见《ADS\_Synthesis\_User\_Guide》的第二章第 3 小节内容。

设置完成后点击右键菜单中的 Run 运行 Compile，结果如下图所示：

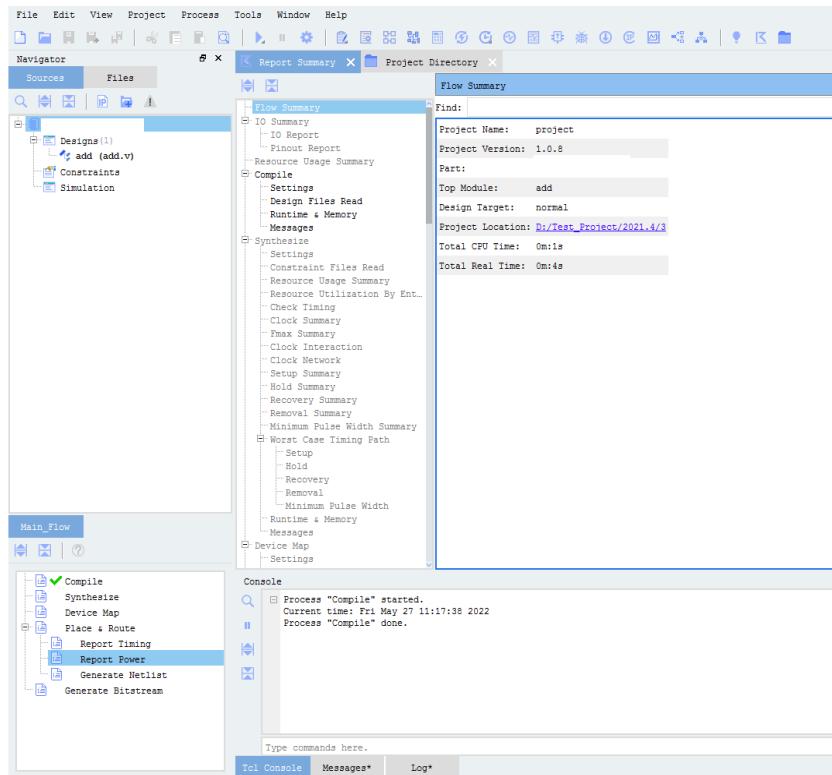


图 3-3 Compile 结果界面

## 3.2 Synthesize

### 3.2.1 ADS

Synthesize 阶段基于 Compile 的 DB, mapping 和 optimization 后生成 technology 网表。

在主界面的 Synthesize 处右键，弹出如下图所示右键菜单：

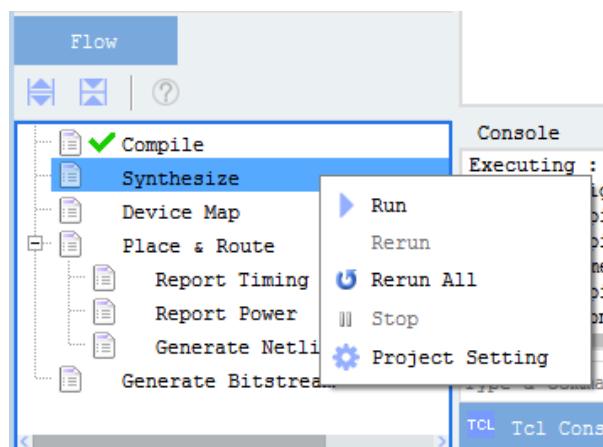


图 3-4 Synthesize 右键菜单

点击右键菜单中的 Project Setting 会弹出如下界面：

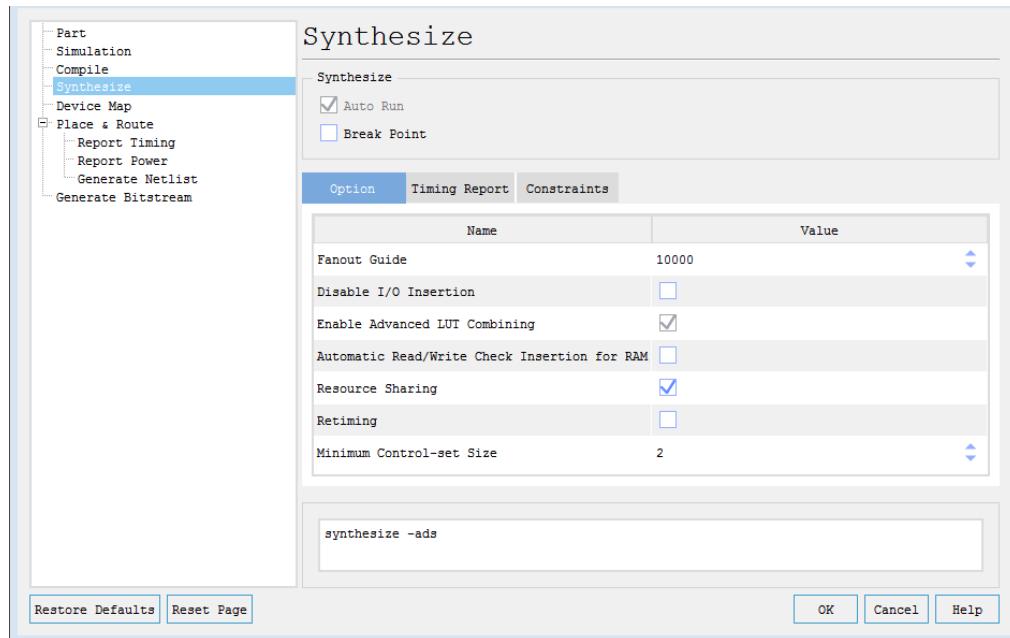


图 3-5 Synthesize 选项设置

Synthesize 的配置选项详情可见《ADS\_Synthesis\_User\_Guide》的第二章第 3 小节内容。

设置完成后点击右键菜单中的 Run 运行 Synthesize，结果如下图所示：

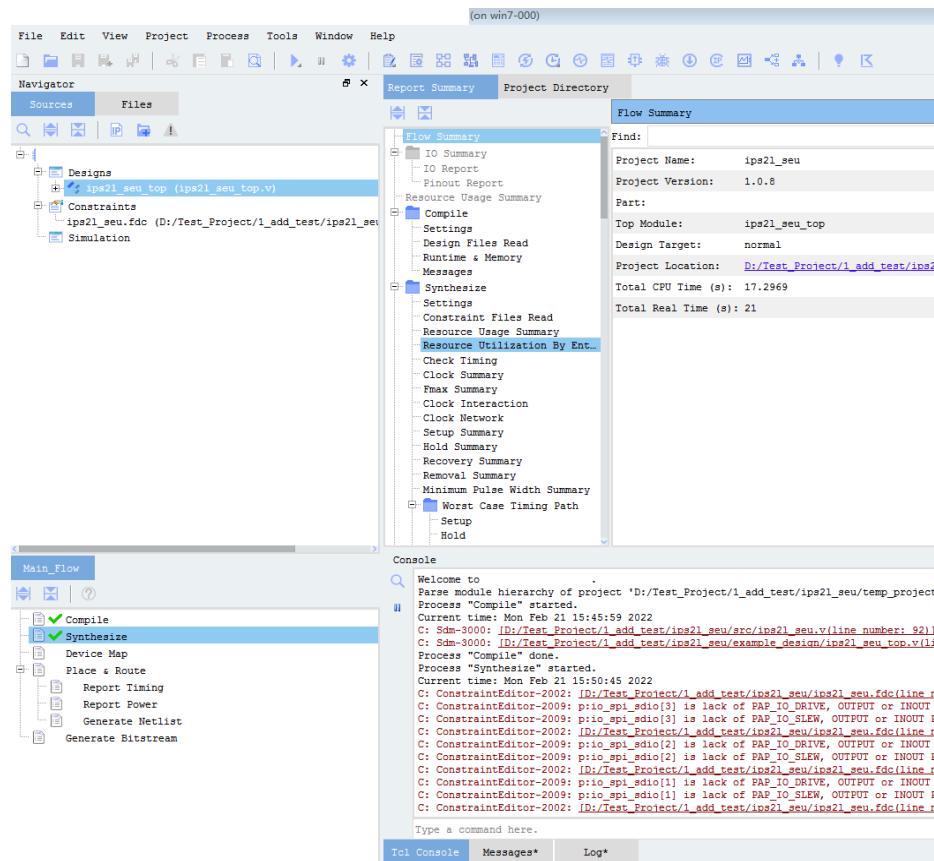
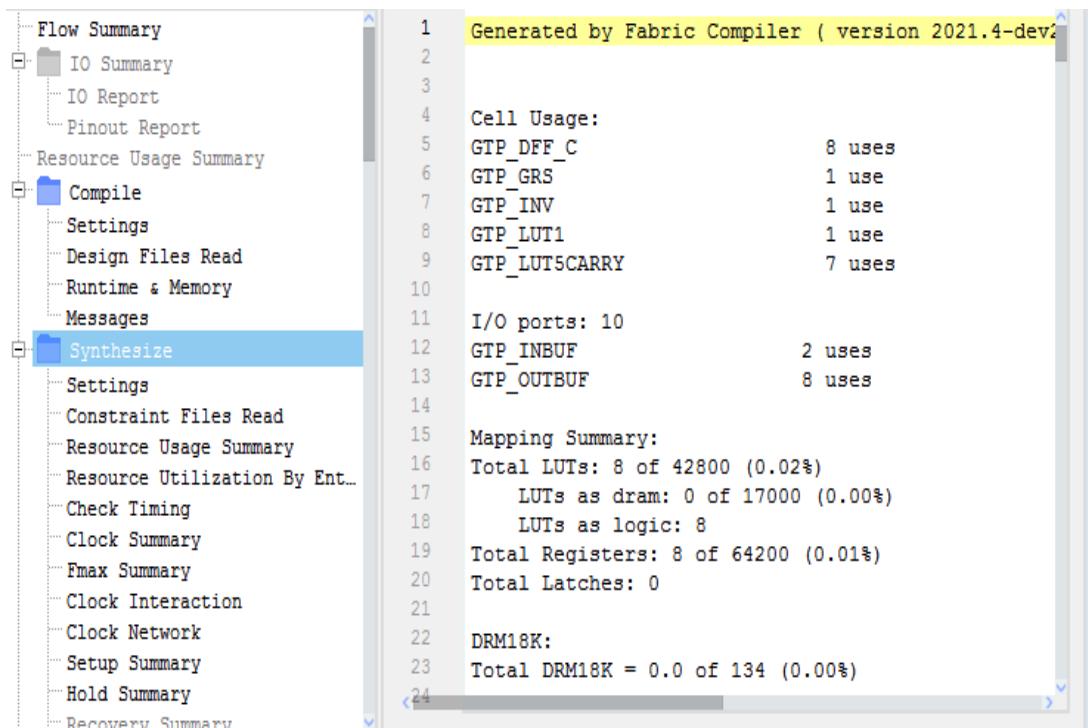


图 3-6 运行 Synthesize 结果界面

从上图中可以看到 Flow Summary 中 Synthesize 相关 Report 的链接已经可以点击打开。鼠

鼠标单击 Synthesize 就能看到 synthesize 的文本报告，如下图。



```

1 Generated by Fabric Compiler ( version 2021.4-dev)
2
3
4 Cell Usage:
5 GTP_DFF_C           8 uses
6 GTP_GRS              1 use
7 GTP_INV              1 use
8 GTP_LUT1              1 use
9 GTP_LUT5CARRY         7 uses
10
11 I/O ports: 10
12 GTP_INBUF            2 uses
13 GTP_OUTBUF           8 uses
14
15 Mapping Summary:
16 Total LUTs: 8 of 42800 (0.02%)
17     LUTs as dram: 0 of 17000 (0.00%)
18     LUTs as logic: 8
19 Total Registers: 8 of 64200 (0.01%)
20 Total Latches: 0
21
22 DRM18K:
23 Total DRM18K = 0.0 of 134 (0.00%)
  
```

图 3-7Synthesize 文本报告

也可以在下图所示位置右键，点击 Open Report 也能打开上图所示报告。另外，如下图，右键还能通过点击 Open Containing Folder 可以打开报告所在文件夹。

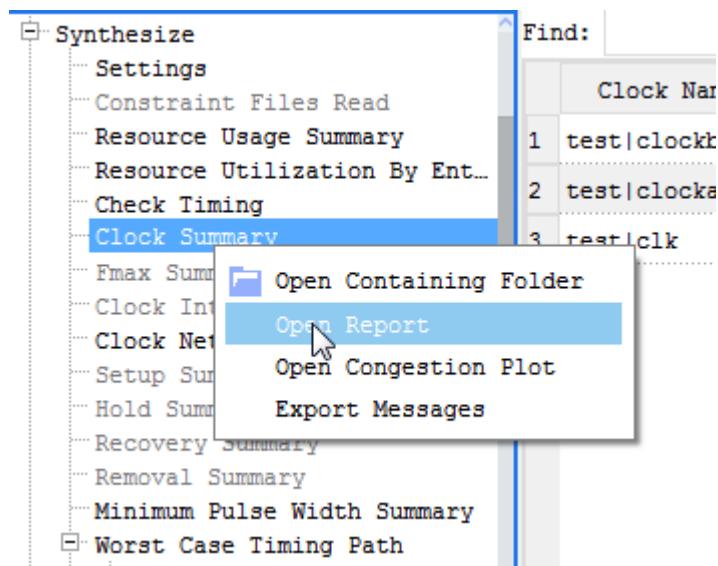
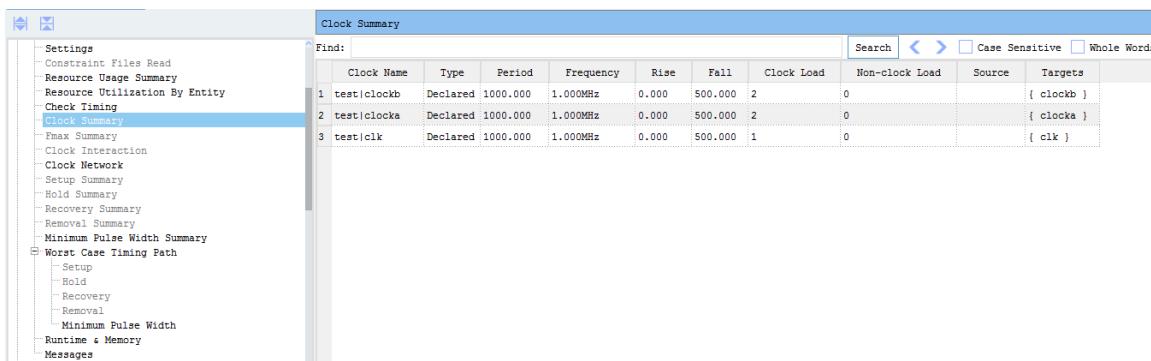


图 3-8 报告区右键菜单

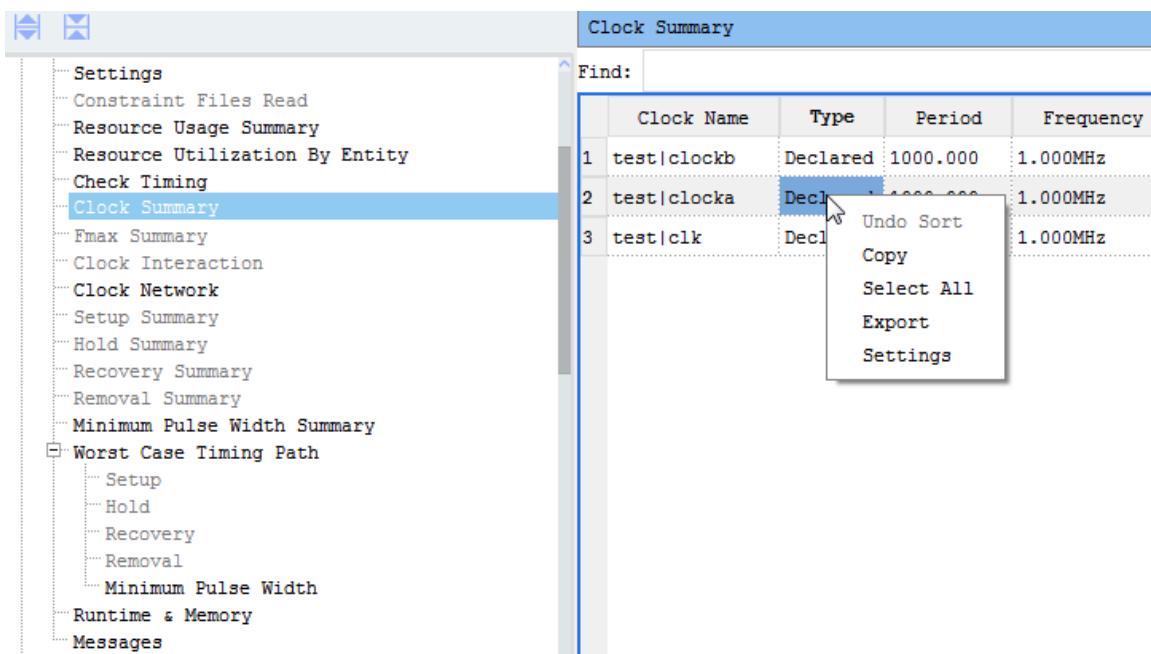
鼠标单击 Clock Summary 看到如下界面(查看其它 report 类似，单击对应项即可):



Clock Name	Type	Period	Frequency	Rise	Fall	Clock Load	Non-clock Load	Source	Targets
1 test clockb	Declared	1000.000	1.000MHz	0.000	500.000	2	0	{ clockb }	
2 test clocka	Declared	1000.000	1.000MHz	0.000	500.000	2	0	{ clocka }	
3 test clk	Declared	1000.000	1.000MHz	0.000	500.000	1	0	{ clk }	

图 3-9 Synthesize Report

上图中，Clock Summary 下方有个搜索条，在文本输入框中可以输入需要搜索的字符，点击后面的 Search 按钮可以执行搜索操作。如果搜索结果有多个，后面的“前一个”或“后一个”按钮会高亮，点击“previous”可以跳到上一个搜索结果，点击“next”可以跳到下一个搜索结果。搜索时还可以选择是否匹配大小写，或匹配整字。在表格报告中右键，可以看到如下右键菜单。



Clock Name	Type	Period	Frequency
1 test clockb	Declared	1000.000	1.000MHz
2 test clocka	Declared	1000.000	1.000MHz
3 test clk	Declared	1000.000	1.000MHz

图 3-10 表格右键菜单

点击每一列的表头（如上图中的 Clock Name）可以排序，Undo Sort 可以恢复产生报告时表格内容的顺序。当表格中有选中的内容时，可以通过 Copy 拷贝选中的内容。Select All 可以选中整个表格。Export 可以导出整个表格内容。Settings 中可以设置表格数据的行高，如下图

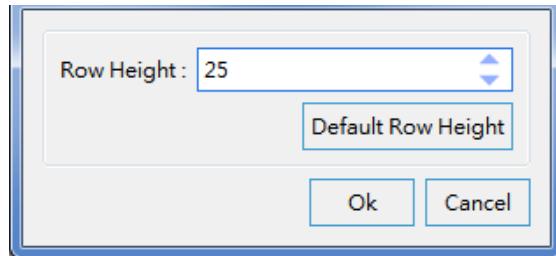


图 3-11 设置表格行高界面

表格行高默认为 25 pixel。

### 3.2.2 Synplify\_Pro

Synthesize\_settings 介绍如下：用户的 verilog 或者 VHDL 设计代码还可以使用第三方综合工具进行综合，综合时基于目标 FPGA 的库，使用生成的 VM 文件作为 Pango Design Suite 的设计输入。第三方综合工具生成的 VM 文件格式是 verilog。

前仿库目录为\$InstallDir\arch\vendor\pango\verilog\simulation，如：

C:\pango\PDS\_2021.3\arch\vendor\pango\verilog\simulation。

在主界面下图 Synthesize 处右键，弹出如下图所示右键菜单

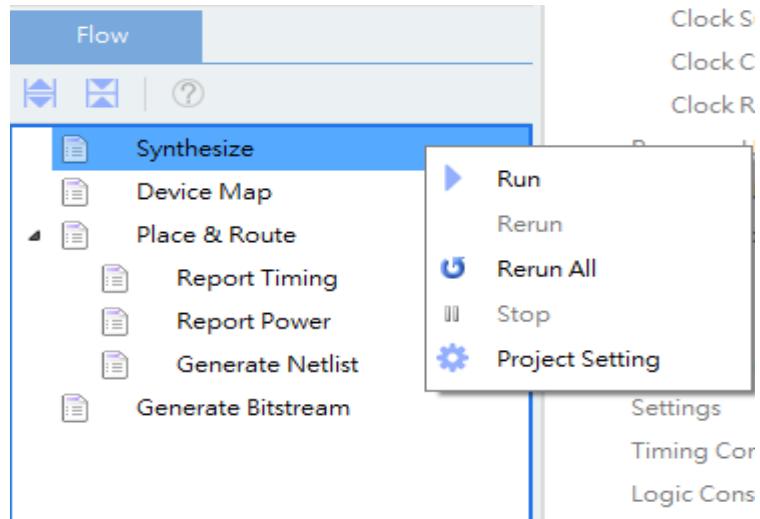


图 3-12 Synthesize 右键菜单

点击右键菜单中的 Project Setting 会弹出如下界面。

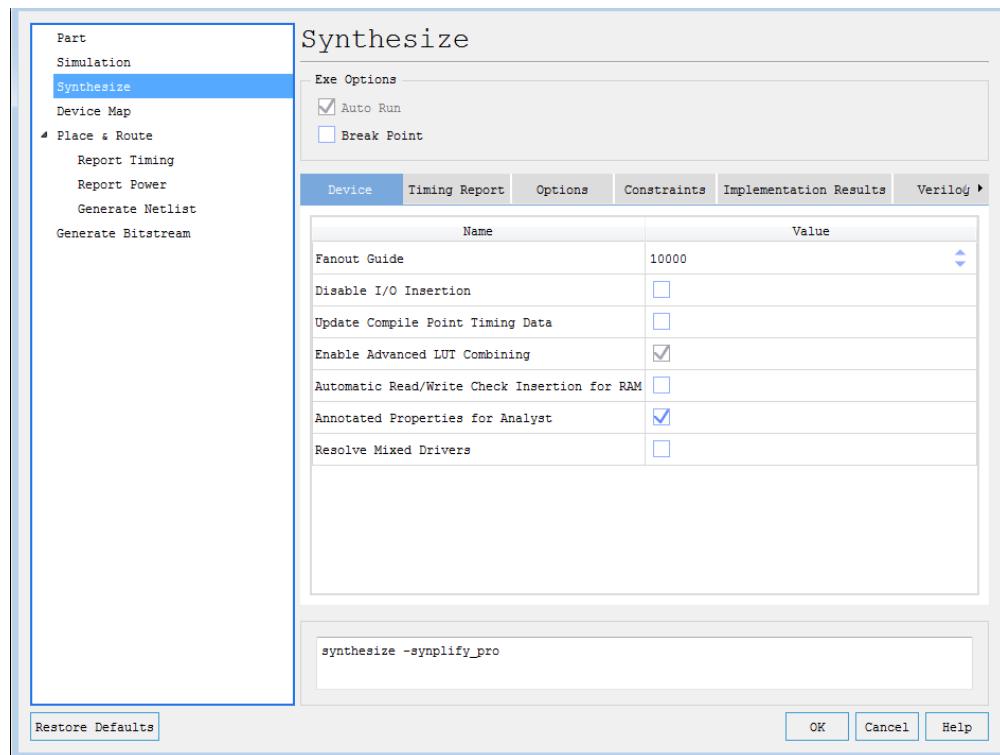


图 3-13synthesize-device 选项卡配置选项界面

上图中涉及到如下选项(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的 fpga\_reference.pdf 文档第 319 页中的 Pango Device Mapping Options 小节中的内容):

**Fanout Guide** : Sets a guideline for the fanout limit. The default is 10000. For tips on setting and using fanout limits, see Setting Fanout Limits and Controlling Buffering and Replication.

**Disable I/O Insertion** : When enabled, it prevents automatic I/O insertion during synthesis. By default, it is disabled.

**Update Compile Point Timing Data** : Determines whether changes to a locked compile point force its parents to be remapped and updated with the new timing model of the child. See Compile Point Basics, for details.

**Enable Advanced LUT Combining** : Prepare netlist for advanced LUT combining.

**Automatic Read/Write Check Insertion for RAM**: Lets the synthesis tool insert bypass logic around the RAM to prevent a simulation mismatch between the RTL and post-synthesis simulations. The synthesis software globally inserts bypass logic around the RAM that reads and writes to the same address simultaneously. Disable this option, when you cannot simultaneously read and write to the same RAM location and want to minimize overhead logic.

**Annotated Properties for Analyst** : Annotates the design with properties after the design is

compiled. You can view these properties in the schematic views

**Resolve Mixed Drivers :** When a net is driven by a VCC or GND and active drivers, enable this option to connect the net to the VCC or GND driver.

点击上图中的 timing Report 选项卡，出现如下界面：

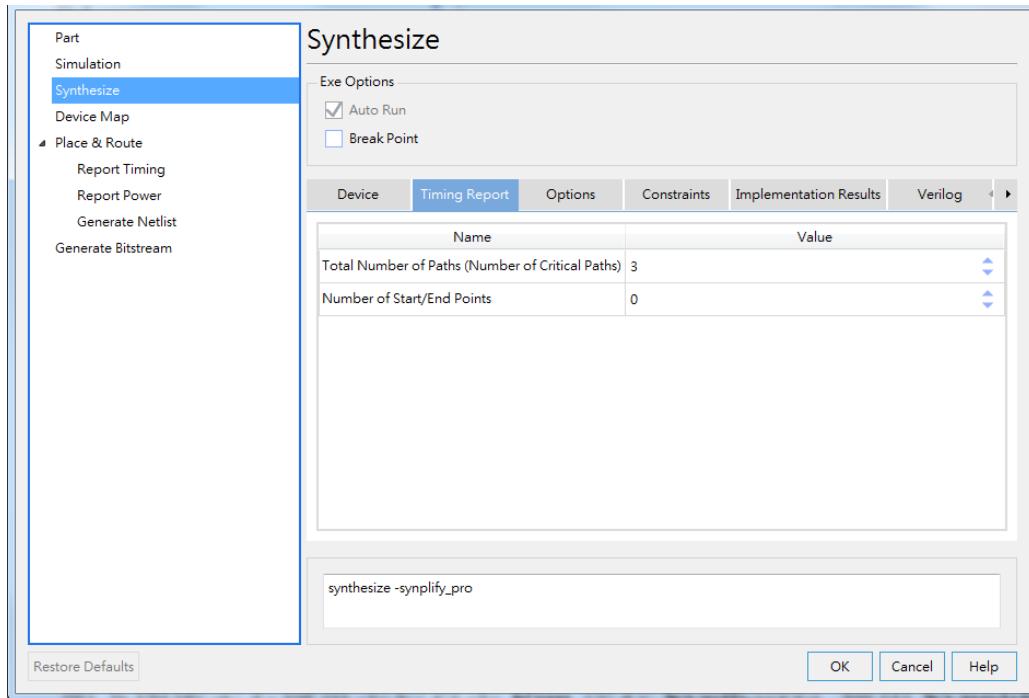


图 3-14 synthesize-Timing Report 选项卡配置选项界面

上图中涉及到如下综合选项(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的 fpga\_command\_reference.pdf 文档第 316 页中的 Timing ReportPanel 小节中的内容):

**Total Number of Paths (Number of Critical Paths):** Set the number of critical paths for the software to report.

**Number of Start/End Points:** Specify the number of start and end points to see reported in the critical path sections.

点击上图中 Options 选项卡，出现如下界面

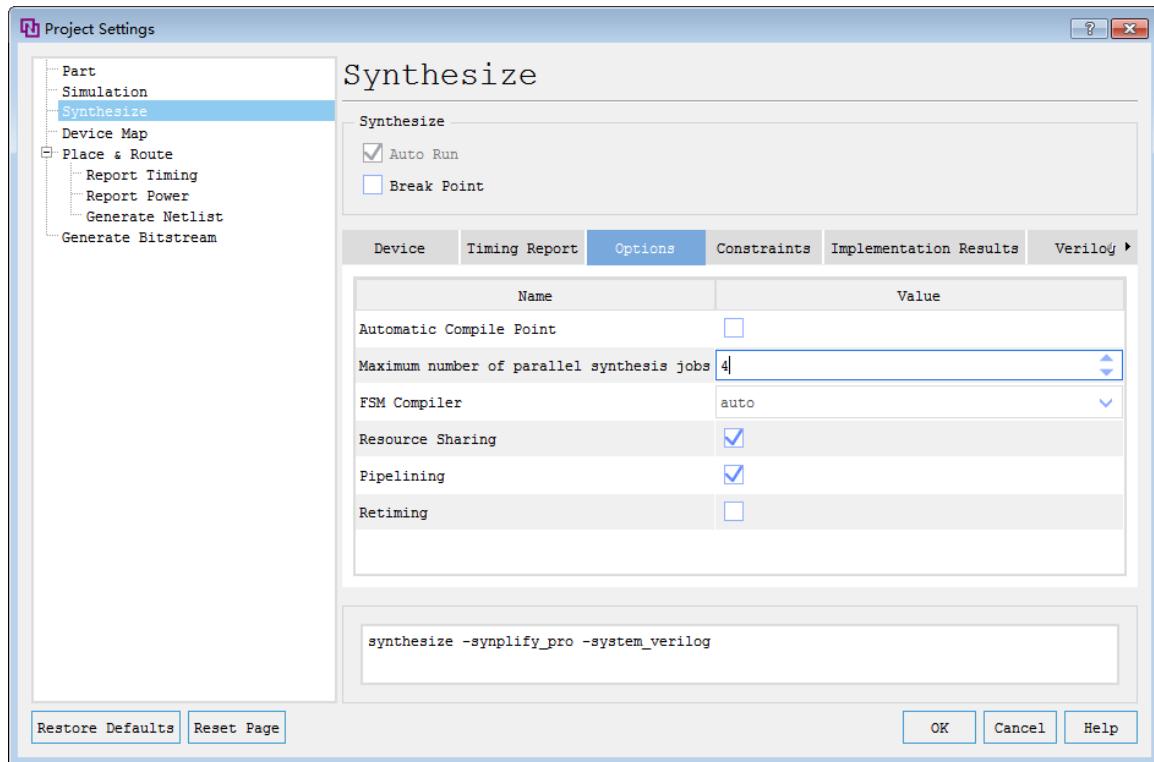


图 3-15 synthesize-Options 选项卡配置选项界面

上图中涉及到的综合选项如下(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的 fpga\_command\_reference.pdf 文档第 311 页中的 Options Panel 小节中的内容):

Automatic Compile Point: Specify boundary constraints for each compile point individually.

Maximum Number of parallel synthesis jobs: Specifies the maximum number of concurrent processes used for synthesis.

FSM Compiler: Determines whether the FSM Compiler is run

Pipelining: Runs designs at a faster frequency by moving registers after the multiplier or ROM into the multiplier or ROM

Resource Sharing: Determines whether you optimize area by sharing resources. When enabled, this optimization technique runs during the compilation stage of synthesis. Even if it is disabled, the mapper can still flatten the netlist and re-optimize adders, multipliers as needed to improve timing, because this setting does not affect the mapper

Retiming: Determines whether the tool moves storage devices across computational elements to improve timing performance in sequential circuits. Note that the tool might retime registers associated with RAMs, DSPs, and generated clocks, regardless of the Retiming setting.

在上图中点击 constraints 选项卡，出现如下界面

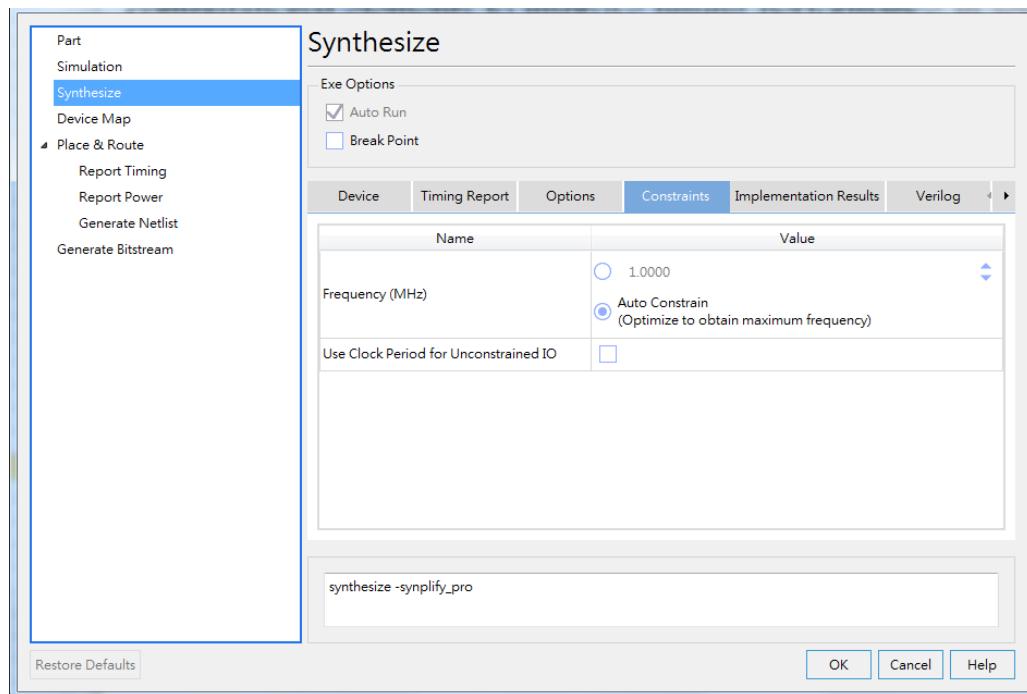


图 3-16 synthesize-Constraints 选项卡配置选项界面

上图中涉及到的综合选项如下(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的 fpga\_command\_reference.pdf 文档第 313 页中的 Constraints Panel 小节中的内容):

**Frequency (MHz)** : Sets the default global frequency. You can either set the global frequency here or in the Project view. To override the default you set here, set individual clock constraints from the SCOPE interface.

**Auto Constrain**: When enabled and no clocks are defined, the software automatically constrains the design to achieve the best possible timing. It does this by reducing periods of each individual clock and the timing of any timed I/O paths in successive steps.

**Use clock period for unconstrained IO**: Determines whether default constraints are used for I/O ports that do not have user-defined constraints. When disabled, only define\_input\_delay or define\_output\_delay constraints are considered during synthesis or forward-annotated after synthesis. When enabled, the software considers any explicit define\_input\_delay or define\_output\_delay constraints. In addition, for all ports without explicit constraints, it uses constraints based on the clock period of the attached registers. Both the explicit and implicit constraints are used for synthesis and forward-annotation. The default is off (disabled) for new designs.

在上图中点击 Implementation Results 选项卡，出现如下界面：

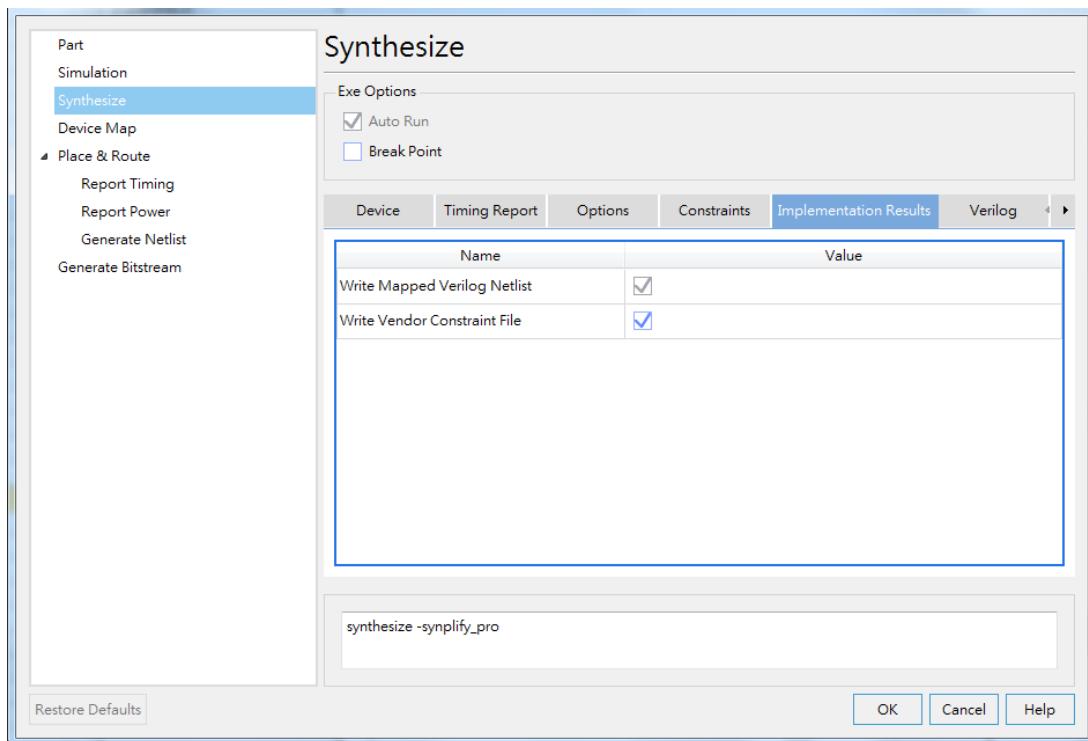


图 3-17 synthesize-Implementation Results 选项卡配置选项界面

在上图中涉及到如下综合选项(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的 fpga\_command\_reference.pdf 文档第 314 页中的 Implementation Results Panel 小节中的内容):

Write Mapped Verilog Netlist: Generates mapped Verilog or VHDL netlist files.

Write Vendor Constraint File: Generates a vendor-specific constraint file for forward annotation.

在上图中点击 verilog 选项卡标签, 出现如下界面:

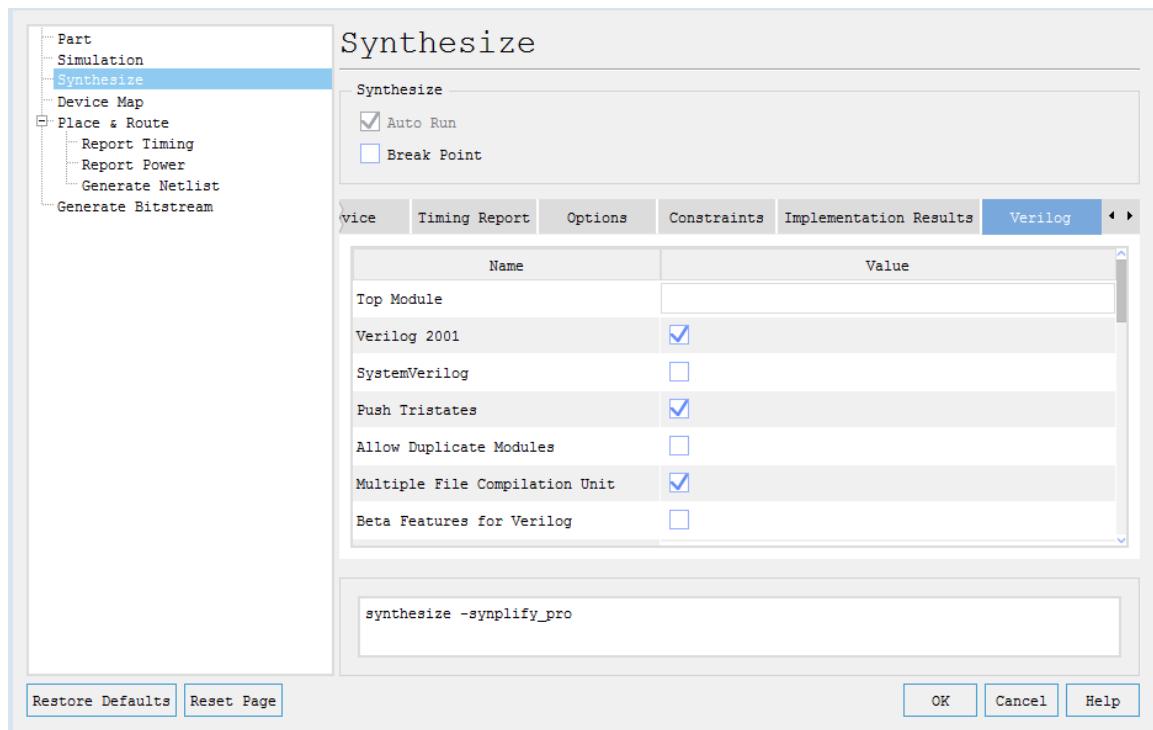


图 3-18 synthesize-Verilog 选项卡配置选项界面

上图中涉及到如下综合选项(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的 fpga\_command\_reference.pdf 文档第 320 页中的 Verilog Panel 小节中的内容):

Top Module : The name of the top-level module of your design.

Verilog 2001 : When enabled, the default Verilog standard for the project is Verilog 2001.

When both Verilog 2001 and SystemVerilog are disabled, the default standard is Verilog 95.

SystemVerilog : When enabled, the default Verilog standard for the project is SystemVerilog which is the default standard for all new projects. Enabling SystemVerilog automatically enables Verilog 2001.

Push Tristates : When enabled (default), tristates are pushed across process/block boundaries.

Allow Duplicate Modules : Allows the use of duplicate modules in your design. When enabled, the last definition of the module is used by the software and any previous definitions are ignored. You should not use duplicate module names in your Verilog design, therefore this option is disabled by default. However, if you need to, you can allow for duplicate modules by enabling this option.

Multiple File Compilation Unit : When enabled (the default), the Verilog compiler uses the compilation unit for modules defined in multiple files.

Beta Features for Verilog : Enables use of any Verilog beta features included in the release.

Loop Limit :Allows you to override the default compiler loop limit value of 2000 in the RTL.

Include Path Order (Relative to Verilog File) : Specifies the search paths for the include commands in the Verilog design files of your project. Use the buttons in the upper right corner of the box to add, delete, or reorder the paths. The include paths are relative.

Library Directories orFiles : Specifies all the paths to the directories which contain the Verilog library files to be included in your design for the project.

Library Extensions (space separated) : Adds library extensions to Verilog library files included in your design for the project and searches the directory paths you specified that contain these Verilog library files.

Generics : Shows generics extracted with Extract Generic Constants. You can override the default and set a new value for the generic constant. The value is valid for the current implementation.

Compiler Directives : Provides an interface where you can enter compiler directives that you would normally enter in the code with ‘ifdef and ‘define statements.

在上图中点击 VHDL 选项卡，出现如下界面

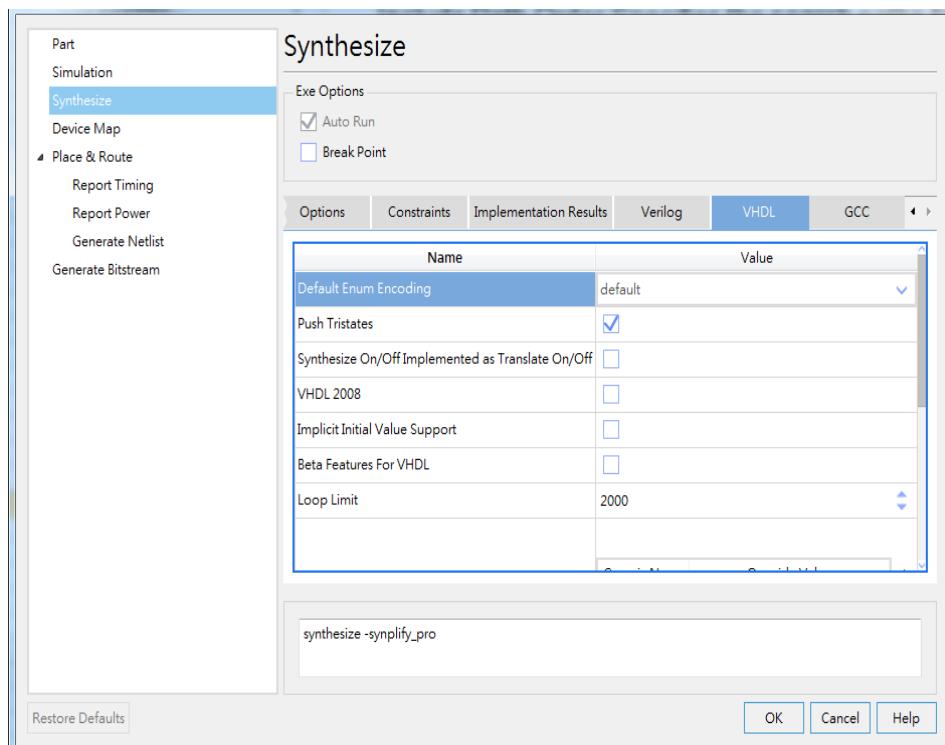


图 3-19 synthesize-VHDL 选项卡配置选项界面

上图中涉及如下综合选项(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的

fpga\_command\_reference.pdf 文档第 317 页中的 VHDL Panel 小节中的内容):

**Default Enum Encoding:** The default enumeration encoding to use. This is only for enumerated types; the FSM compiler automatically determines the state-machine encoding, or you can specify the encoding using the syn\_encoding attribute.

**Push Tristates:** When enabled (default), tristates are pushed across process/block boundaries.

**Synthesis On/Off Implemented as Translate On/Off:** When enabled, the software ignores any VHDL code between synthesis\_on and synthesis\_off directives. It treats these third-party directives like translate\_on/translate\_off directives (see translate\_off/translate\_on for details).

**VHDL 2008:** When enabled, allows you to use VHDL 2008 language standards.

**VHDL 2019:** When enabled, allows you to implement VHDL 2019 conditional analysis identifiers and values.

**Implicit Initial Value Support:** When enabled, the compiler passes init values through a syn\_init property to the mapper.

**Beta Features for VHDL:** Enables use of any VHDL beta features included in the release.

**Loop Limit :** Allows you to override the default compiler loop limit value of 2000 in the RTL. You can apply loop limits using the Verilog loop\_limit or the VHDL syn\_looplmt directive.

**Generics :** Shows generics extracted with Extract Generic Constants. You can override the default and set a new value for the generic constant. The value is valid for the current implementation.

在上图中点击 GCC 选项卡，出现如下界面：

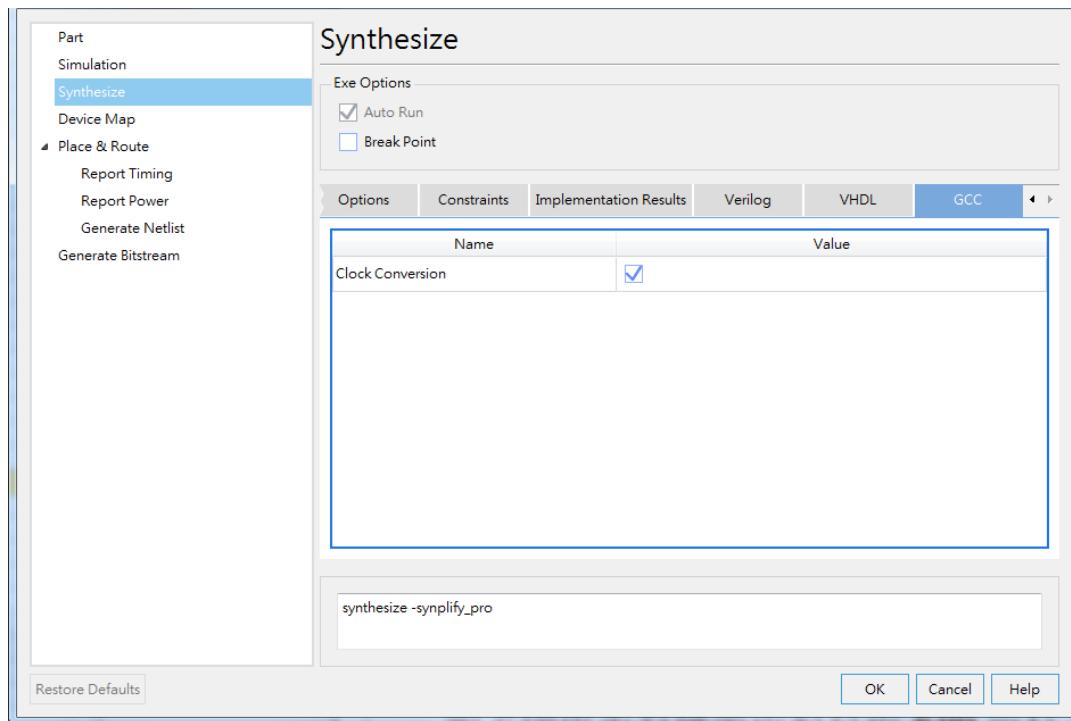


图 3-20 synthesize-GCC 选项卡配置选项界面

上图中涉及到如下综合选项(上图所示的选项详细描述可见安装目录中子目录 syn/doc 中的 fpga\_command\_reference.pdf 文档第 334 页中的 GCC Panel 小节中的内容):

Clock Conversion : Performs gated and generated clock optimization when enabled.

单击后如下图所示右键点击 Run 运行 Synthesize, 使用 Synplify Pro 综合工具编译.v 文件产生.vm 网表文件。

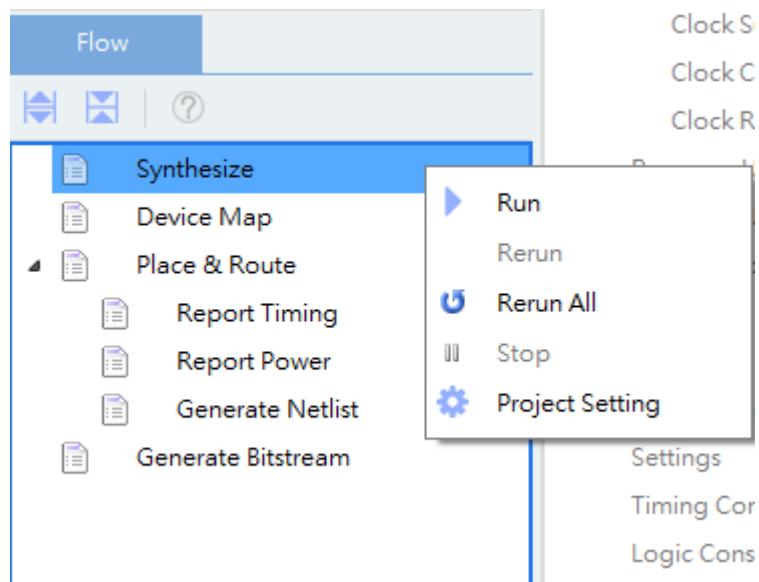


图 3-21 运行 Synthesize

完成以上操作，将会看到如下图所示的界面

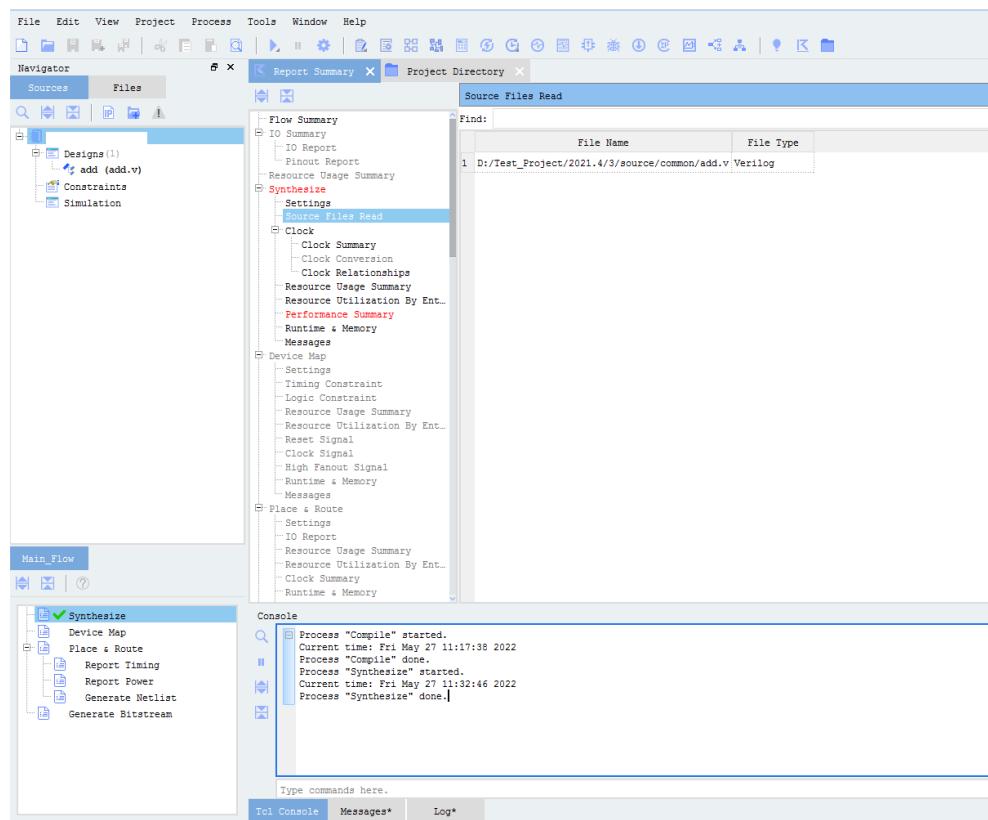


图 3-22 运行 Synthesize 后软件界面

### 3.3 Device Map

Device\_Map\_settings 介绍如下：Map 映射的主要作用是将设计映射到具体型号的子单元上（LUT、FF、Carry 等）。

Device Map 的选项设置 如下图所示：

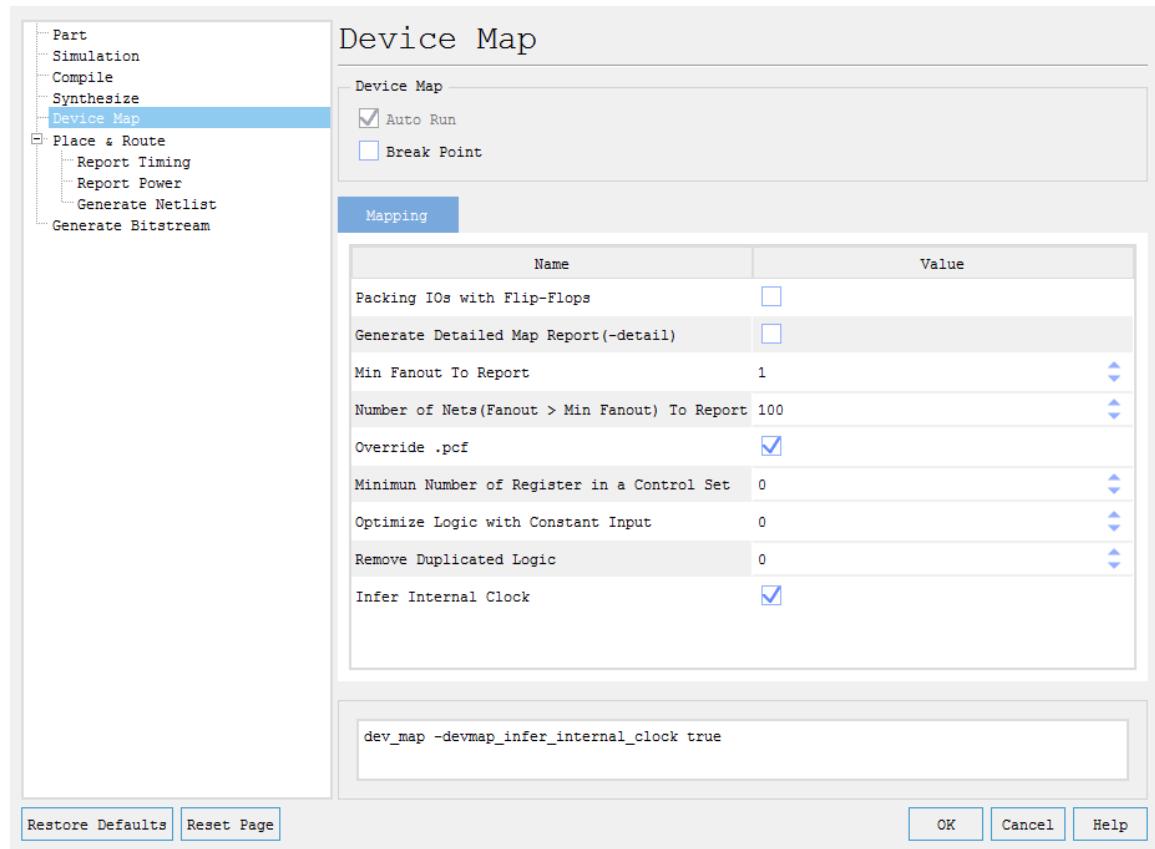


图 3-23 Device Map 的选项设置

Device Map 配置选项：

**【Packing IOs with Flip-Flops】** 设置在映射 IO 时是否加 Flip-Flops。表示是否将与 IO 端口直接相连的 FFs Pack 到相对应的 IO 端口中。此 IO 端口输入输出模式可以是 input、output、inout。默认不选中该选项，即 FF 不和 IO Pack 在一起。

**【Generate Detailed Map Report(-detail)】** 设置是否在 map 的 report 文件中打印每一个 module 实例中的资源使用率。默认不选中该选项。

**【Minimum Fanout To Report】** 设置在 map report 中打印 net 的 fanout 数目的最小值，目前仅针对报告中 High Fanout Signal 项。默认值是 1。

**【Number of Nets(Fanout >= Min Fanout) To Report】** 设置在 map report 中打印 net 的数目，打印的 net 的 fanout 要大于最小 fanout 设置值。默认值是 100。

**【Override .pcf】** 该项用来设置当用户约束文件(fdc/lcf)产生的 pcf 约束信息与已有的 pcf 文件约束信息不一致时，是否覆盖已有的 pcf。如果值为 true，执行 device map 时会自动覆盖 pcf；如果值为 false，则不会覆盖 pcf。默认选中该选项。

**【Enable timing commands commented by Synplify Pro】** 该项用来设置使能状态，即被 OEM

标注为注释的时序命令能够在 PDS 流程中生效。默认不选中该选项。（ADS 综合不支持）

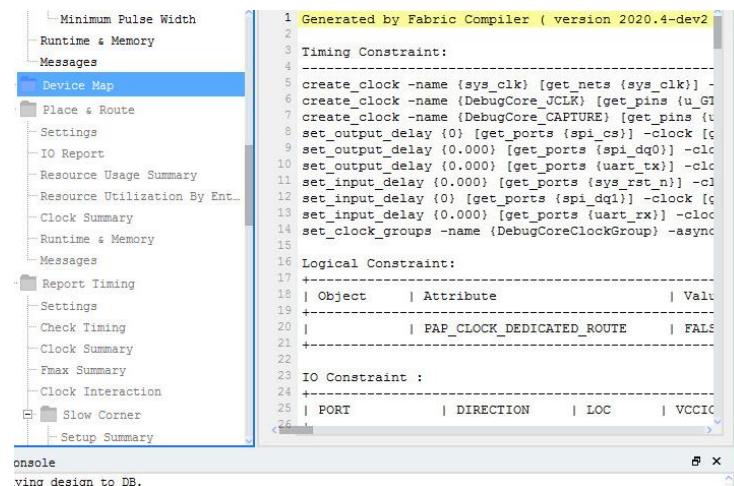
【Minimun Number of Register in a Control Set】用于控制同一种 control set 下，最小触发器的数目。但触发器的数目小于门限时，则将门控信号作为数据通路用 LUT 来实现。Control set 是指触发器的时钟、使能信号，RST 信号。优化该控制信号可以增加资源使用率。默认值是 0。

【Optimize Logic with Constant Input】主要用于优化输入的常量逻辑。常量逻辑主要是指 VCC、GND 逻辑。该优化操作主要通过电路固有的特性，将原来的 VCC、GND 断开或优化到最优状态。默认值是 0。

【Remove Duplicated Logic】该功能主要作用是将部分重复逻辑合并，属于面积优化选项。但是该操作同样会影响到综合阶段的复制工作，使时序性能下降。默认值是 0。

【Infer Internal Clock】该选项控制是否在 map 阶段 infer 内部时钟（比如内部触发器产生的分频时钟，不影响对 pll 时钟的 infer），在 PGC 器件中默认打开，其他器件默认关闭。

完成设置后执行 device map 后结果如下：

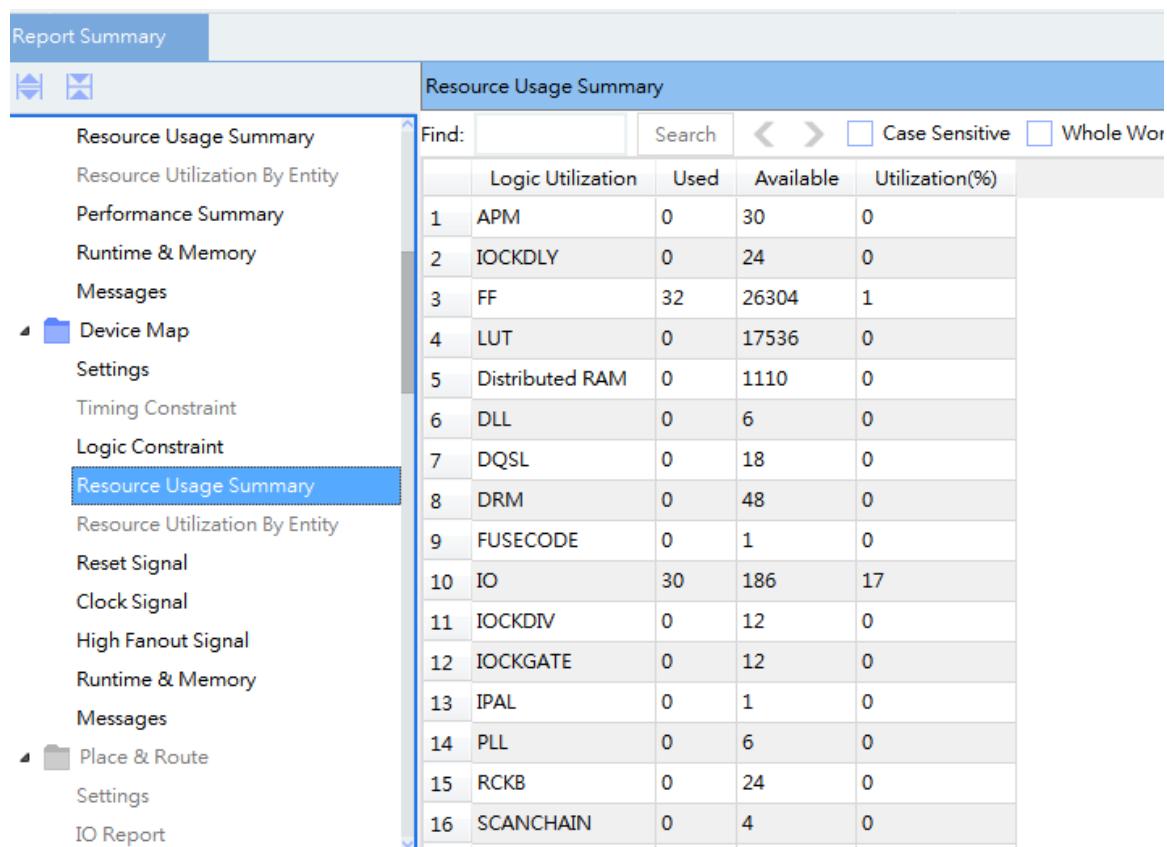


```

1 Generated by Fabric Compiler ( version 2020.4-dev2
2
3 Timing Constraint:
4 -----
5 create_clock -name {sys_clk} [get_nets {sys_clk}] -
6 create_clock -name {DebugCore_JCLK} [get_pins {u_G1}
7 create_clock -name {DebugCore_CAPTURE} [get_pins {u_G2}
8 set_output_delay {0} [get_ports {spi_cs}] -clock [{sys_clk}]
9 set_output_delay {0.000} [get_ports {spi_dq0}] -clock [{sys_clk}]
10 set_output_delay {0.000} [get_ports {uart_tx}] -clock [{sys_clk}]
11 set_input_delay {0.000} [get_ports {sys_rst_n}] -clock [{sys_clk}]
12 set_input_delay {0} [get_ports {spi_dq1}] -clock [{sys_clk}]
13 set_input_delay {0.000} [get_ports {uart_rx}] -clock [{sys_clk}]
14 set_clock_groups -name {DebugCoreClockGroup} -async
15
16 Logical Constraint:
17 -----
18 | Object | Attribute | Value
19 +-----+-----+
20 |       | PAP_CLOCK_DEDICATED_ROUTE | FALSE
21 +-----+
22
23 IO Constraint :
24 -----
25 | PORT | DIRECTION | LOC | VCCIO
26
  
```

图 3-24 Device Map 后软件界面

单击 Resource Usage Summary，结果如下：



	Logic Utilization	Used	Available	Utilization(%)
1 APM	0	30	0	0
2 IOCKDLY	0	24	0	0
3 FF	32	26304	1	0
4 LUT	0	17536	0	0
5 Distributed RAM	0	1110	0	0
6 DLL	0	6	0	0
7 DQSL	0	18	0	0
8 DRM	0	48	0	0
9 FUSECODE	0	1	0	0
10 IO	30	186	17	0
11 IOCKDIV	0	12	0	0
12 IOCKGATE	0	12	0	0
13 IPAL	0	1	0	0
14 PLL	0	6	0	0
15 RCKB	0	24	0	0
16 SCANCHAIN	0	4	0	0

图 3-25 Device Map Resource Usage Summary 界面

### 3.4 Place & Route

#### 3.4.1 Place & Route 设置

Place\_Route\_settings 介绍如下：布局布线（Place & Route）根据用户约束和物理约束，对设计模块进行实际的布局及布线。

Place & Route 的选项设置如下图所示：

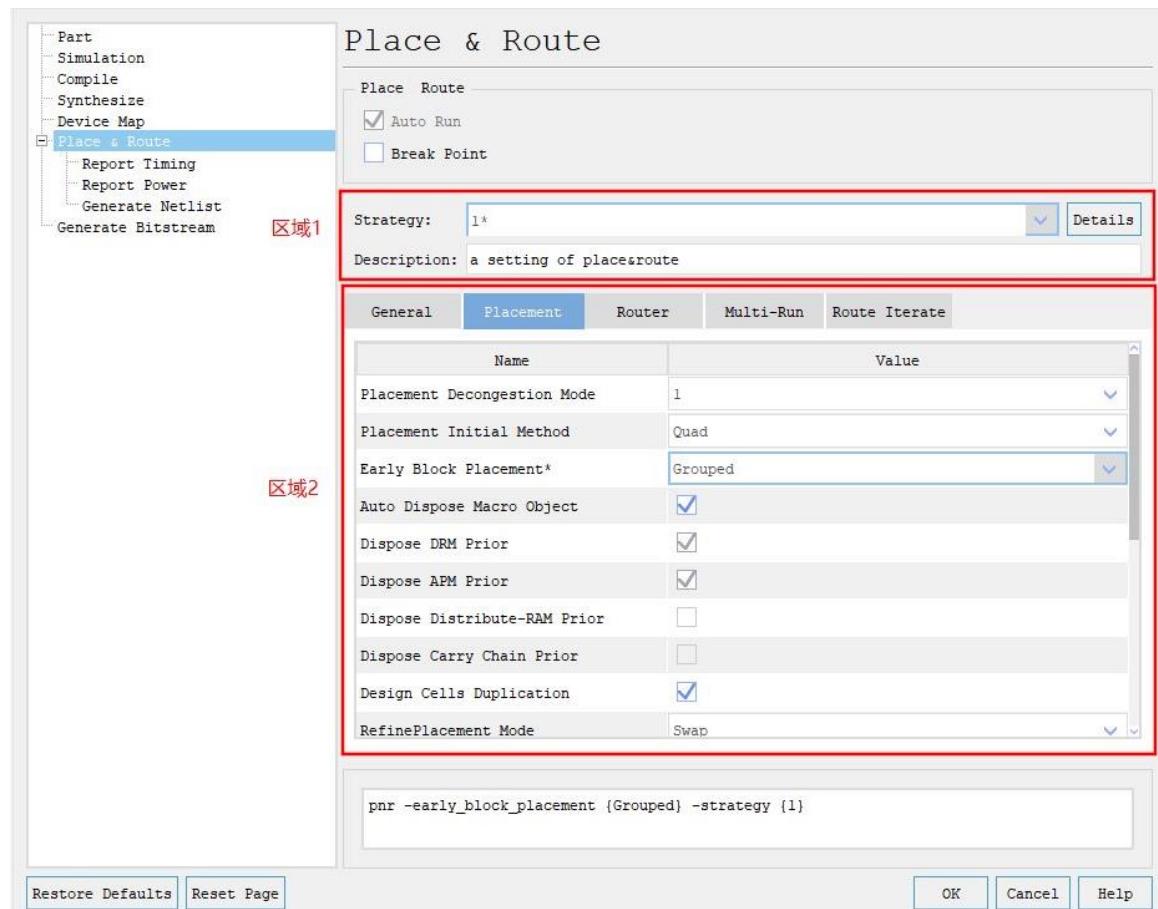


图 3-26 Place & Route 的选项设置

区域 1 为策略选项：

**【Strategy】** 用户可根据设计选择对应的策略进行调试，下拉框选项包括 0~7，其中 0 为默认值，表示未使用策略，1~7 分别对应 7 种策略，选择某个策略时，策略所涉及的选项会在下方的选项名称后标记“\*”，如果对应选项的值修改后与策略默认值不同时，则会在 Strategy 中的策略名后标记“\*”；（仅支持 Titan2 系列）；

**【Details】** 用于按钮显示策略中包含的选项的值及其当前实际值；

**【Description】** 对所选策略的使用场景进行描述；

区域 2 共包括 5 个模块（tab 页）：

该部分的不同器件间选项的支持情况与默认值详情在本节末尾的表格中体现

A、常规模块：

**【Mode】** 有 fast、nomal 和 performance 三个选项，fast 模式，运行速度最快，性能最差；normal 模式，兼顾性能和速度，能满足普通设计要求；performance 模式，性能最优，但是运行时间最长。

【Optimize Multi-Corner Timing】对 hold 违例进行修复时，是否打开 multi-corner 功能，只有 Optimize Hold Timing 打开后有效。启用后，hold fix 对 slow corner 和 fast corner 的 hold 违例都进行修复。不启用，hold fix 默认对 fast corner 的违例进行修复。

【Place Only】只布局而不进行绕线。

【Optimize Hold Timing】当设计有 hold 违例时，是否对违例进行修复。

【Max Optimize Hold Timing Iterations】当 Optimize Hold Timing 打勾时，该选项表示对违例进行修复的最大迭代次数，当达到最大迭代次数时停止修复 Hold 违例。

【Fix Hold Violation Threshold(ps)】当 Optimize Hold Timing 打勾时，可以使用该选项对 slack 小于该值的时序路径进行 Hold 时序优化，从而为 Hold 时序留有一定余量（单位：ps）。

【Optimize Hold Timing In Route】当设计有 hold 违例时，是否在布线过程中对违例进行修复。

【Fix Hold Violation Threshold For Clock Path Cross SRB(ps)】当 Optimize Hold Timing In Route 打勾时，可以使用该选项对 slack 小于该值的时序路径进行 Hold 时序优化，从而为 Hold 时序留有一定余量（单位：ps），该参数仅对违例 path 对应的时钟 path 通过 SRB 时生效。如果将该参数设置得过大，可能由于需要修复的违例较多，修复时间较长。如果将该参数设置得过小，则存在 hold 时序违例的风险。建议参数值设置区间为[0, 1800]。

【Fix Hold Violation Threshold For Clock Path On Clock Tree(ps)】当 Optimize Hold Timing In Route 打勾时，可以使用该选项对 slack 小于该值的时序路径进行 Hold 时序优化，从而为 Hold 时序留有一定余量（单位：ps），该参数仅对违例 path 对应的时钟 path 不通过 SRB 时生效，设置区间为[0, 200]。

【Max Number Of Hold Violated Paths To Fix】当 Optimize Hold Timing In Route 打勾时，可以使用该选项对 hold fix 过程中可处理的 path 数量进行限制，超过限制时，将不进行 hold 违例修复。设置区间为[0, 100000]。

特别的对于修复 hold 违例功能会报告下述 warning:

1、C: Route-2015

1)、触发场景：当最差的 hold slack 值超过 5000 ps 时就会报告此 Critical Warning，并不进行 hold 修复；

2)、示例：C: Route-2015:Hold violation is 6000 ps over the critical value of 5000 ps, beyond the critical value will not be repaired. 上述 warning 就表示工程中最差的 slack 值为 6000ps，超

过了 5000ps;

3)、解决方法：当报告 C:Route-2015 时，需要用户重新确认工程的时钟规划是否合理，超过了 5000ps 的 hold 违例，此时需要用户根据实际情况调整工程。

## 2、C: Route-2037

1)、触发场景：当工程中 hold 违例 path 数量超过"Max Number Of Hold Violated Paths To Fix"选项设置的值，就不执行 hold 时序修复流程，并报告此 Critical Warning，并不进行 hold 修复；

2)、示例：C: Route-2037: Number of hold violated paths is 15012, over the limit 15000 and hold violation fix will be stopped, you can change the value of option "Place & Route" -> "Max Number Of Hold Violated Paths To Fix". 上述 warning 就表示工程中 hold 违例的 path 数量为 15012，而选项中设置的值为 15000. 此时就不进行 hold 修复；

3)、解决方法：当报告 C:Route-2037 时，用户可以根据 warning 中提示的 path 数量，修改 option "Place & Route" -> "Max Number Of Hold Violated Paths To Fix"中的值，然后重新运行布局布线。

## 3、C: STA-3017 和 W: Timing-4105

1)、触发场景：当 hold fix 开关打开，修复 hold 违例后，若 endpoint 上的 hold slack 不满足用户设置的 hold violation threshold，会报告这两个 Warning 进行提示(只针对 launch clock 或 capture clock 走 SRB 的 path)；当 hold fix 关闭，不报告该这两个 Warning；C: STA-3017 用于提示用户存在不满足余量的 path 条数，W:Timing-4105:用于提示用户不满足余量的 hold 最差的 path 的 endpoint、clock 及对应 slack；

### 2)、示例：

C: STA-3017: There are 10 clock cross SRB paths do not meet the required hold violation threshold (500ps) .上述 warning 表示有 10 条时钟走 SRB 的 paths 的 hold 余量不满足用户设置的值（默认值为 500ps，可更改）

W:Timing-4105: The worst slack of endpoint t1daau\_dp/dsmnbop[5]/opit\_0/D of clock lclkp is -115ps (fast corner) , but required hold violation threshold is 500ps. 上述 warning 表示不满足余量的 hold 最差的 path 的 endpoint 为 t1daau\_dp/dsmnbop[5]/opit\_0/D、clock 为 lclkp、hold slack 为 fast corner 下-115ps、而用户设置的余量为 500ps；

### 3)、解决方法：当报告 C: STA-3017 和 W:Timing-4105 时，用户可以查看 warning 中提示

的 path 数量和 slack 值，来确定当前工程的 hold 余量是否满足预期，可根据实际工程情况来修改期望的 hold 余量值即：option "Place & Route" -> " Fix Hold Violation Threshold For Clock Path Cross SRB(ps)"中的值，然后重新运行布局布线。

**【Action Timeout(min)】** 单种子工程超时参数，为 0min 表示无超时机制，设置值大于 0 则启用超时机制，例如-action\_timeout 60 表示设置超时时限为 60min，当 PNR 运行超过 60min 时单种子被杀死。

## B、布局模块

**【Placement Decongestion Mode】** 由于用户设计复杂，各逻辑间连接（Net）关系强，可能导致布局算法无法有效扩散，布局密度偏高，对时序优化以及布线时长造成不利影响。根据用户设计主要资源（通常指 LUT、FF、APM、DRM）最大占比，设置面积扩展系数。设置值可选[0,1,2,3,4,5,6,7,8,9]中任意一整数。一般情况下，模式设置值越大，布局范围越大，布线速度越快，但时序将会变差。当资源（LUT、FF、APM、DRM）使用率超过 70% 时，模式设置值越大时，布线速度将会变得不确定，建议该参数的值可选为 0, 1, 2, 3, 4。

**【Placement Initial Method】** 该选项用于选择全局布局时的起始位置。该参数设置为 Center 时，表示起始位置从芯片中间开始；该参数设置为 Quad 时，表示布局起始位置取决于位置约束和 IO 布局结果。

**【Early Block Placement】** 该选项用于设置 Macro Instance 优先放置的模式，选定操作为下拉框操作。选择 Greedy 选项时，按照网表时序特征优先；选择 Global 选项时，按照布局距离特征优先；选择 Grouped 选项时，按照网表设计特征优先；当选择 Close 选项时，表示不允许 Macro Instance 优先放置。选择除 Close 外的选项时，才能对 Auto Dispose Macro Object 选项进行操作。

**【Auto Dispose Macro Object】** 该选项是 PDS 软件自适应选择宏模块优先放置开关。当该选项为不勾选时，用户可以通过 Dispose DRM Prior、Dispose APM Prior、Dispose Distribute-RAM Prior 和 Dispose Carry Chain Prior 选项分别选择 DRM、APM、DRAM、Carry Chain 是否优先放置。

**【Dispose DRM Prior】** 该选项表示布局时优先放置 DRM。当不勾选 Auto Dispose Macro Object 时，可以操作该选项。

**【Dispose APM Prior】** 该选项表示布局时优先放置 APM。当不勾选 Auto Dispose Macro Object 时，可以操作该选项。

**【Dispose Distribute-RAM Prior】** 该选项表示布局时优先放置 Distribute-RAM。当不勾选 Auto Dispose Macro Object 时，可以操作该选项。

**【Dispose Carry Chain Prior】** 该选项表示布局时优先放置 Carry Chain。当不勾选 Auto Dispose Macro Object 时，可以操作该选项。

**【Design Cells Duplication】**： 布局算法时序优化，采用自适应插入和复制逻辑单元算法优化布局时序。

**【RefinePlacement Mode】** 布局算法时序优化，设置值可选[Swap, Bump]两种模式。Bump 模式下仅以 instance 为单位进行细调，较适用小规模器件。Swap 模式下会以 Site 为单位进行粗调，结合以 instance 为单位的细调，较适用于大规模器件。

**【RefinePlacement Cluster】** 使能详细布局过程中打包开关。当打开时，有利于布局时序性能收敛。

**【RefinePlacement Iteration】** 该选项用于设置详细布局收敛迭代次数。当详细布局时序未能充分收敛时，建议将该参数值设置得更大，与此同时，详细布局收敛时间更长。

**【Enable Recovery Analysis in GP】** 在 global placement 中使能 recovery 时序分析。

**【Enable Recovery Analysis in DP】** 在 detail placement 中使能 recovery 时序分析。

**【Enable Recovery Analysis in RP】** 在 replication placement 中使能 recovery 时序分析。

**【Placement Logic Optimization】** 使能优化布局性能的布局逻辑优化开关。当勾选该选项时，程序将进行逻辑优化，当前逻辑优化仅包含 LUT6D pack 功能。该选项选中时，才能对 LUT6D Packing Strategy 选项进行操作。

**【LUT6D Packing Strategy】** LUT6D pack 功能实现策略，布局阶段使用 LUT6D 合并符合条件的 LUT，降低 LUT 资源，设置值可选[0,1,2,3]表示不同的 pack 策略。针对 PG2T390H 和 PG2T390HX 器件在原有的功能策略基础上进行了时序方面的优化，其他器件未进行优化，保持之前的逻辑策略。

[0]: 关闭 LUT6D pack 功能；

[1]: 进行 LUT6D pack（对性能有较大影响）

[2]: 进行 LUT6D pack（对性能有较小影响）

[3]: 进行 LUT6D pack，LUT 合并数量相对较多（对性能影响适中）

**【Planning Before Place】** 开启该开关后可以在布局前进行模块级别的位置规划，可以降低出现布局后同一模块内逻辑分散的问题的几率。

## C、布线模块

**【Enable Fast Router】**：控制是否使用全局布线。

**【Fast Router Mode】**：控制全局布线的模式，分为自动选择模式（Auto），非拆线模式（Mode1），拆线模式（Mode2）。

**【Multi-thread routing thread size】**：控制多线程布线的线程数数量，默认值因器件不同。

**【Reduce CE/RS signal】**：用于控制是否使用 Control Chain。

**【CE/RS signal series limit】** CE/RS 控制信号链最大长度。

**【Enable Global IOCLKGATE】**：控制时钟布线是否自动使用 IOCLKGATE。

**【Slack Prior In Global router】**控制是否使用布线优先的全局布线器。当 Enable Fast Router 勾选时，才能使用。

**【Slack Weight】** 控制时序因素在布线过程中的权重。

**【Pin Cost Increasing Speed】** 控制拥塞端口增加代价速度的速度。

**【Max iterations in congest mode】**该选项用于设置在控制拥塞布线模式下的最大布线迭代次数。

**【Dump Router Congestion Plot】**该选项用于设置是否打印拥塞节点文件，勾选后，运行布线流程时，会在 place\_route 目录新建 CongestionPlot 和 CongestionNetInfo 文件夹，分别记录在详细布线过程中每轮迭代产生的拥塞节点文件和布线冲突的 net 信息。

在 PDS 的 Report Summary -> Flow Summary 内任意位置右键选择“Open Congestion Plot”可打开拥塞图，自动加载 CongestionPlot 中的拥塞节点文件，视图化详细布线每轮迭代拥塞情况。

拥塞图页面功能介绍：

拥塞图左侧显示每轮生成的拥塞节点文件；

右侧视图里根据选中的拥塞节点文件显示 SRB 上的拥塞情况；

右下角 “The max value is: \*” 记录本轮迭代 SRB 上最大的拥塞值；

左下角 “SRB Conflict(max color mapping)” 处可设置最大值，拥塞图中各 SRB 上的拥塞值根据设置的最大值自适应调整，标量显示不同颜色，颜色越深，表明在该处 SRB 上拥塞越大；

支持 SRB 搜索功能，在右上角搜索框内搜索指定 SRB，会在拥塞图上标识出以搜索的 SRB 为中心，上下左右各拓展 4 个 Tile 的方框；

支持 CongestionPlot 文件替换，在页面上方可加载其他工程生成的拥塞节点文件。

【Enable Recovery Analysis in Routing】在 routing 中使能 recovery 时序分析。

【Check Clock Net Routing Path】检测时钟信号是否使用非时钟树路径进行布线，若选择开启，当时钟信号使用非时钟树路径时，对该路径进行告警处理，在 Console 中打印 Critical Warning 信息：C: Route-2036。

#### D、多次运行模块

【Global Random Seed（Placement）】PDS 的布局算法属于全局优化算法，依赖于初始值，不同初始值会得到不同的算法优化效果。该参数主要作用是为布局算法提供随机位置，方便用户获得最佳性能的布局布线结果。

【Iterations（Placement）】执行多种子布局布线的次数，在 Place\_Only 模式下也可以使用，最小为 1。

【Random Seed Step（Placement）】执行多种子布局布线的种子步长。

【Save Best Run（Placement）】执行多种子布局布线的保存最优结果的个数。

【Maximum Number Of Mutil-seed Running In Placement Parallel】并行执行多种子的最大数目。

【Filter Strategies】设置筛选最优种子的过滤条件，最高优先级。

【Sort Strategy】设置筛选最优种子的排序规则，次高优先级。

【Timeout】设置每个种子的最长运行时间，超时自动终止任务；大于 0 时生效，单位为分。

【Generate Bitstream In Multi-seed Mode】设置是否在 PnR 完成后生成位流。

完成设置后，执行【Place & Route】操作。

运行布局布线过程中，多种子保留种子筛选规则：优先以用户设置规则筛选最优种子结果，Filter Strategies 为第一优先级，用户可设置 setup、hold、recovery、removal、wirelength 对应的取值范围，默认无；次优先级 Sort Stratgy，用户可指定排序优先策略，默认排序优先级：setup>hold>recovery>removal>wirelength，若用户指定排序为 hold，除指定排序项 hold 为第一优先级，其余均按照默认优先级筛选种子（即当 hold 对应 slack 值一样时，依次对比 setup、recovery、removal、wirelength 对应的值），以此排序种子结果，最后按照用户设置的【Save Best Run（Placement）】保留筛选后的种子结果。

所有种子结果均汇总在 multiseed\_result.csv 中，其中满足用户设置的过滤条件，表格中

以 UserFilter=1 标记，否则为 0。

说明：Filter Strategies、Sort Strategy 中可设置的四个选项是 setup/hold/recovery/removal 对比 slow 和 fast 的 corner 值取较小值。

multiseed\_result.csv 中保留多种子 setup/hold/recovery/removal 在 slow 和 fast 的 corner 的 8 个结果，以及 Routing Arc Length、Worst Slack(before route)、Wire Length 的值。

此外，在运行过程中还可以点击 Show Multi-Run State 来查看运行状态，并可手动终止相应种子的运行，如下图：

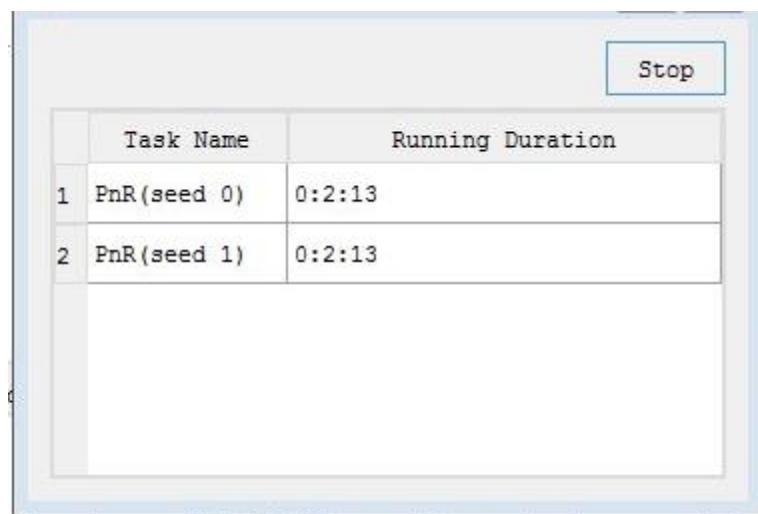


图 3-27 Place & Route Multi-Run State 界面

运行布局布线后界面如下：

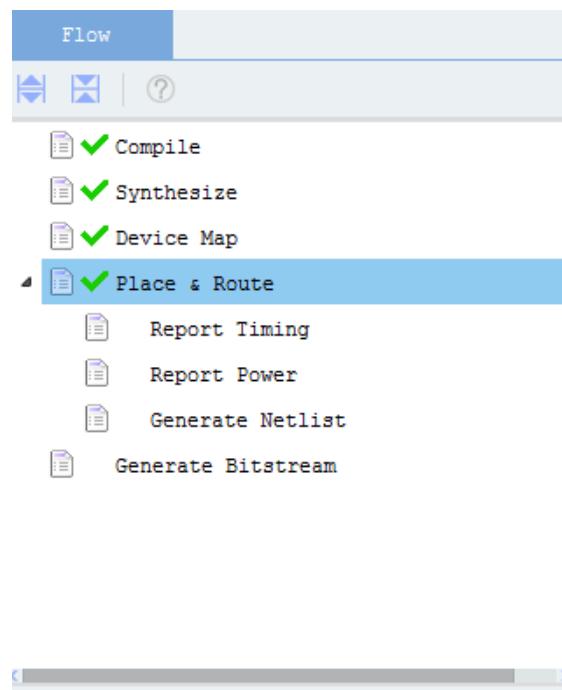


图 3-28 Place & Route 运行后软件界面

用户通过 PNR 右键 Select a Seed To Apply 指定任一种子运行 pnr 之后流程。

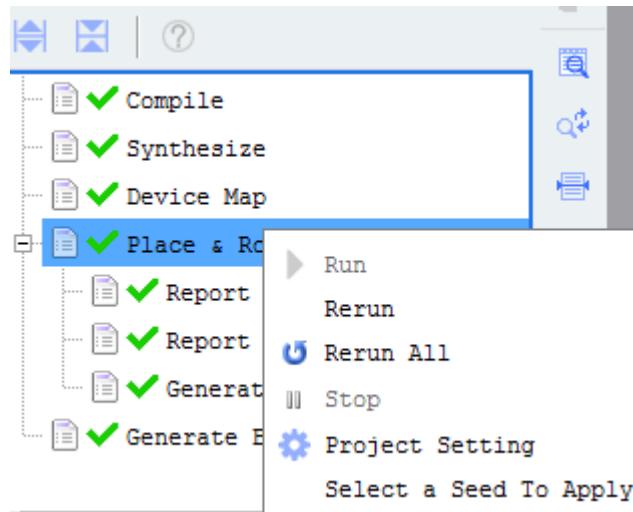


图 3-29 Place & Route 右键界面

选择 Select a Seed To Apply，选择种子指定，点击 apply 后，若已运行 PNR 之后流程显示 out of date，选择 Report timing Rerun，根据指定种子运行流程。

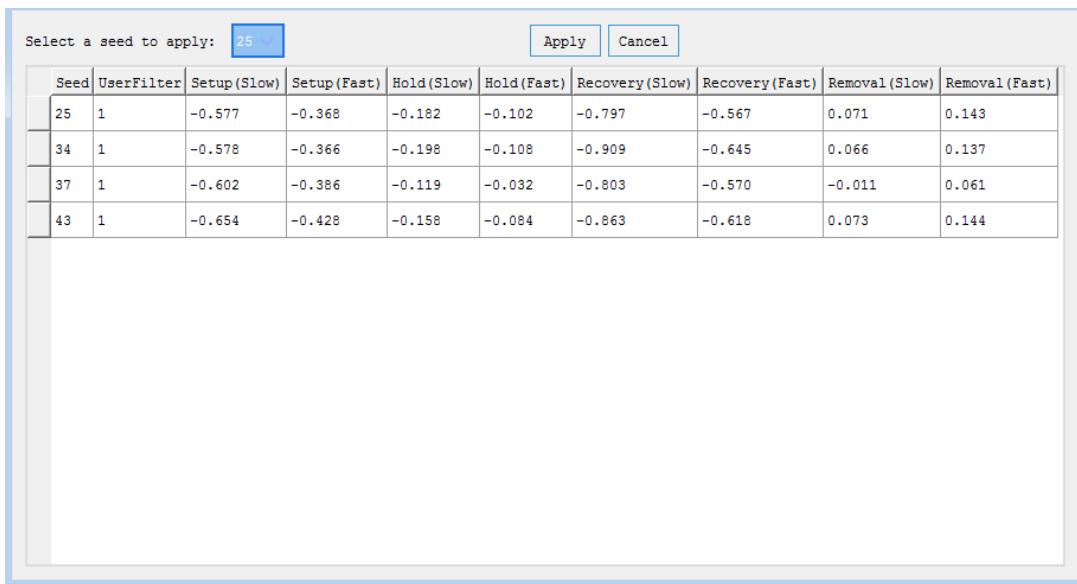


图 3-30 Place & Route Select a Seed To Apply 界面

## 5) 布线自动迭代模块

该模块支持按设置值自动提取不满足 setup slack 的 net，在每轮迭代中生成 route\_constraint\_ite.rcf 文件，对 net 设置优先布线约束指令，每轮迭代的约束指令都累加到 Place\_Route 目录下的 all\_route\_constraint\_ite.rcf 中，下一轮布局布线迭代会重新加载 all\_route\_constraint\_ite.rcf 中的优先布线指令，直到达到设置的迭代最大次数或满足 net 对应的 setup slack 阈值条件才会退出迭代过程。

all\_route\_constraint\_ite.rcf 文件中保留最好的一次迭代结果。

若客户工程原始未添加 rcf 文件，则布线自动迭代后使用的是 PR 目录下迭代累加的“all\_route\_constraint\_ite.rcf”文件，若想使用绕线迭代后的结果，可打开 RCE 加载该文件后重跑 PR 流程；若有原始 user\_rcf 文件，则布线使用的是 user\_rcf+ all\_route\_constraint\_ite.rcf 两个文件中的约束命令合集。若要重新跑 PR 全流程时需要手动汇总两个 rcf 文件结果到 user\_rcf。

**【Enable Route Constraint Iterate】**布线迭代开关，用于决定是否打开或关闭自动迭代，默认不勾选不进行布线迭代，勾选后按相关配置进行布线自动迭代。

**【Route Constraint Iterate Times】**执行布线自动迭代最大次数，当迭代次数达到最大后自动终止迭代。

**【Route Constraint Slack Value(ns)】**指定生成 rcf 的 net 对应的 setup slack 阈值。

**【Route Constraint Max Timing Path】**指定每轮迭代中 timing 报告的最大时序路径条数。

迭代次数和 net 对应的 setup slack 阈值两个参数也是布线迭代退出的条件，满足任意一个条件则会退出布线自动迭代过程。

以下是 PNR 阶段不同系列的各选项支持情况与默认值详情的表格：

其中 PG2L100H 是单线程布线器件，它的 general, router 选项与 Logos 的选项一致

类型	配置项	Titan系列	Logos系列	Compact系列	Logos2系列	Titan2系列
General	Mode	Normal	Normal	Normal	Normal	Normal
	Optimize Multi-Corner Timing	×	×	×	无该选项	无该选项
	Place Only	×	×	×	×	×
	Optimize Hold Timing	×	×	×	无该选项	无该选项
	Max Optimize Hold Timing Iterations	500	500	500	无该选项	无该选项
	Fix Hold Violation Threshold(ps)	0	0	0	无该选项	无该选项
	Optimize Hold Timing In Route	√	√	√	√	√
	Fix Hold Violation Threshold For Clock Path Cross SRB(ps)	200	200	200	200	200
	Fix Hold Violation Threshold For Clock Path On Clock Tree(ps)	100	100	100	100	100
	Max Number Of Hold Violated Paths To Fix	15000	15000	15000	15000	30000
Action Timeout(min)	0	0	0	0	0	0
Placement	Placement Decongestion Mode	1	1	1	1	1

	Placement Initial Method	Quad	Quad	Quad	Quad	Quad
Early Block Placement	Greedy	Greedy	Global	PG2L200H: Global 其他: Greedy	Global	
Auto Dispose Macro Object	√	√	√	PG2L200H: × 其他: √	√	
Dispose DRM Prior	√	√	√	√	√	
Dispose APM Prior	×	√	×	√	√	
Dispose Distribute-RAM Prior	×	×	×	×	×	×
Dispose Carry Chain Prior	×	×	×	×	×	×
Design Cells Duplication	√	√	√	√	√	
RefinePlacement Mode	Swap	Bump	Bump	Swap	Swap	
RefinePlacement Cluster	√	√	√	√	√	
RefinePlacement Iteration	20	20	20	20	20	
Enable Recovery Analysis in GP	×	×	×	×	×	×
Enable Recovery Analysis in DP	×	×	×	×	×	×
Enable Recovery Analysis in RP	×	×	×	×	×	×
Placement Logic Optimization	无该选项	无该选项	无该选项	√	√	
LUT6D Packing Strategy	无该选项	无该选项	无该选项	PG2L200H: 2 其他: 3	3	
Planning Before Place	无该选项	无该选项	无该选项	PG2L200H: √ 其他: ×	√	
Router	Enable Fast Router	√	√	无该选项	无该选项	无该选项
	Fast Router Mode	Auto	无该选项	无该选项	无该选项	无该选项
	Multi-thread routing thread size	无该选项	无该选项	无该选项	PG2L25H: 2 PG2L50H: 3 PG2L200H: 10	7
	Reduce CE/RS signal	√	√	√	√	√
	CE/RS signal series limit	500	500	500	500	500
	Enable Global IOCLKGATE	×	无该选项	无该选项	无该选项	无该选项
	Slack Prior In Global router	√	√	无该选项	无该选项	无该选项
	Slack Weight	90	90	90	90	90
	Pin Cost Increasing Speed	300	300	300	300	300
	Max iterations in congest mode	30	200	20	无该选项	无该选项
	Dump Router Congestion Plot	×	×	×	×	×
	Enable Recovery Analysis in Routinig	√	√	√	√	√
Multi-Run	Check Clock Net Routing Path	√	√	√	√	√
	Global Random Seed (Placement)	0	0	0	0	0
	Iterations (Placement)	1	1	1	1	1

	Random Seed Step (Placement)	1	1	1	1	1
	Save Best Run (Placement)	1	1	1	1	1
	Maximum Number Of Mutil-seed Running In Placement Parallel	1	1	1	1	1
	Filter Strategies	空	空	空	空	空
	Sort Strategy	setup	setup	setup	setup	setup
	Timeout (min)	720	720	720	720	720
	Generate Bitstream In Multi-seed Mode	×	×	×	×	×
Route Iterate	Enable Route Constraint Iterate	×	×	×	×	×
	Route Constraint Iterate Times	10	10	10	10	10
	Route Constraint Slack Value(ns)	0	0	0	0	0
	Route Constraint Max Timing Path	3	3	3	3	3

### 3.4.2 Place & Route 报告

PNR 结束后（包括 Place Only），Report Summary 中的 Place &Route 可查看布局布线的报告，报告如下图所示：

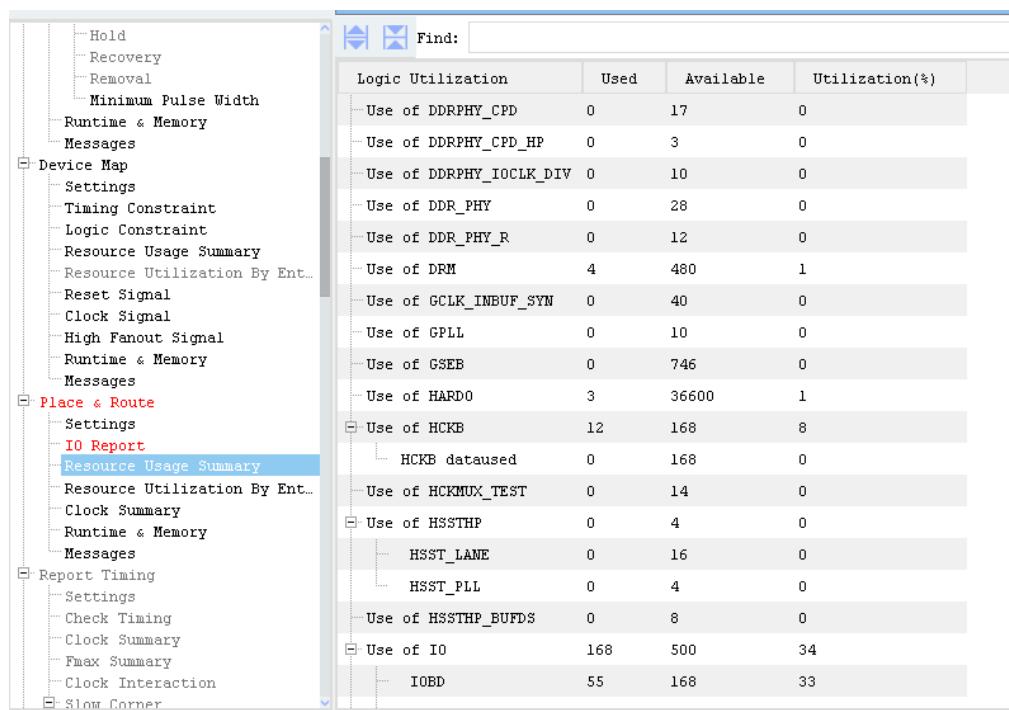
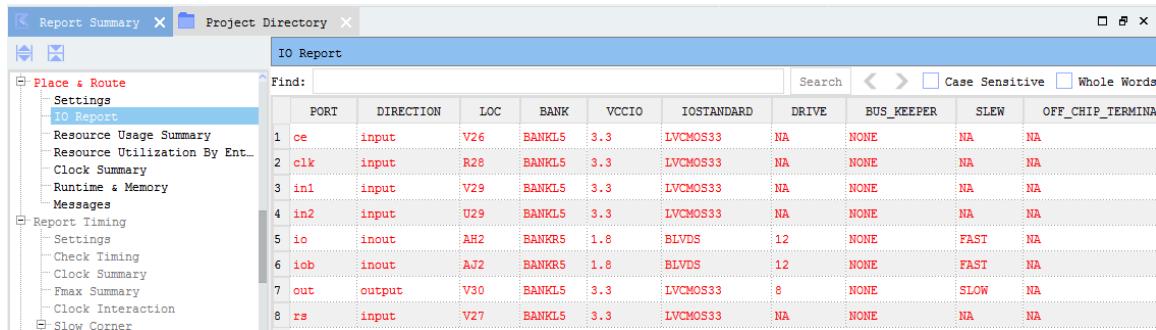


图 3-31 Place & Route Resource Usage Summary 界面

上图显示是 PNR 结束后的工程资源使用情况，对于一些时钟 BUF，在其下方存在一栏 XXXX dataused，此栏显示该时钟 BUF 用于非时钟信号的数量，不同器件中存于此栏的时钟 BUF 类型不同。

## 1) IO Report

在 Place&Route 流程运行完后会生成 IO Report 报告，该报告展示了 IO 的相关信息，报告如下图所示。



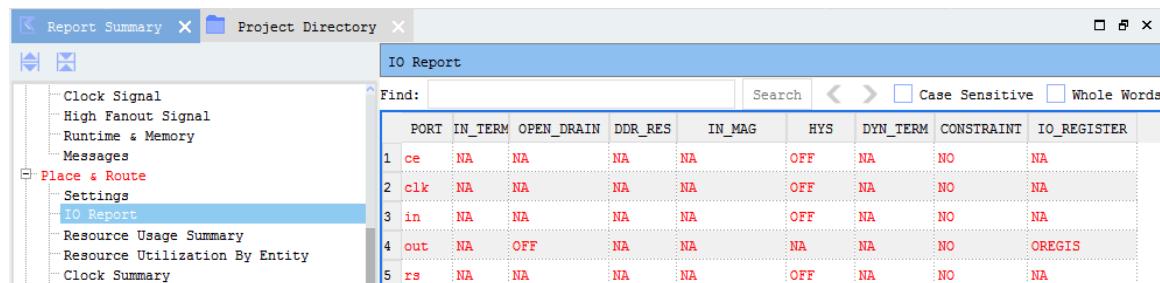
The screenshot shows the 'IO Report' section of the Place & Route report. The table lists the following columns: PORT, DIRECTION, LOC, BANK, VCCIO, IOSTANDARD, DRIVE, BUS\_KEEPER, SLEW, and OFF\_CHIP\_TERMINA. The data rows are:

PORT	DIRECTION	LOC	BANK	VCCIO	IOSTANDARD	DRIVE	BUS_KEEPER	SLEW	OFF_CHIP_TERMINA
1 ce	input	V26	BANKL5	3.3	LVCMS33	NA	NONE	NA	NA
2 clk	input	R28	BANKL5	3.3	LVCMS33	NA	NONE	NA	NA
3 in1	input	V29	BANKL5	3.3	LVCMS33	NA	NONE	NA	NA
4 in2	input	U29	BANKL5	3.3	LVCMS33	NA	NONE	NA	NA
5 io	inout	AH2	BANKR5	1.8	BLVDS	12	NONE	FAST	NA
6 iob	inout	AJ2	BANKR5	1.8	BLVDS	12	NONE	FAST	NA
7 out	output	V30	BANKL5	3.3	LVCMS33	8	NONE	SLOW	NA
8 rs	input	V27	BANKL5	3.3	LVCMS33	NA	NONE	NA	NA

图 3-32 Place & Route IO Report 界面

### a.IO\_REGISTER 列

IO Report 的 IO\_REGISTER 状态显示端口的 IOL 单元是否 pack 触发器 FF，根据触发器 FF pack 到 IOL 的 IDDR、ODDR、TDDR 位置会分别显示为 IREGIS、OREGIS、TREGIS，下图为状态为 OREGIS 的示例。



The screenshot shows the 'IO Report' section of the Place & Route report. The table lists the following columns: PORT, IN\_TERM, OPEN\_DRAIN, DDR\_RES, IN\_MAG, HYS, DYN\_TERM, CONSTRAINT, and IO\_REGISTER. The data rows are:

PORT	IN_TERM	OPEN_DRAIN	DDR_RES	IN_MAG	HYS	DYN_TERM	CONSTRAINT	IO_REGISTER
1 ce	NA	NA	NA	NA	OFF	NA	NO	NA
2 clk	NA	NA	NA	NA	OFF	NA	NO	NA
3 in	NA	NA	NA	NA	OFF	NA	NO	NA
4 out	NA	OFF	NA	NA	NA	NA	NO	OREGIS
5 rs	NA	NA	NA	NA	OFF	NA	NO	NA

图 3-33 IO\_REGISTER 状态 OREGIS

如果 PORT 的 IDDR、ODDR、TDDR 都有 FF，则显示为 I/O/TREGIS，如下图所示。如果无 FF pack 或是非 FF 的单元 pack 到 IOL 的 IDDR/ODDR/TDDR 上，则状态显示为 NA。



The screenshot shows the 'IO Report' section of the Place & Route report. The table lists the following columns: PORT, OPEN\_DRAIN, DDR\_RES, HYS, DYN\_TERM, PREEMP, OUTPUT\_IMPEDANCE, CONSTRAINT, and IO\_REGISTER. The data rows are:

PORT	OPEN_DRAIN	DDR_RES	HYS	DYN_TERM	PREEMP	OUTPUT_IMPEDANCE	CONSTRAINT	IO_REGISTER
1 ce	NA	NA	OFF	NA	NA	NA	NO	NA
2 clk	NA	NA	OFF	NA	NA	NA	NO	NA
3 in1	NA	NA	OFF	NA	NA	NA	NO	NA
4 in2	NA	NA	OFF	NA	NA	NA	NO	NA
5 io	NA	NA	OFF	OFF	NA	NA	NO	I/O/TREGIS
6 iob	NA	NA	OFF	OFF	NA	NA	NO	I/O/TREGIS
7 out	OFF	NA	NA	NA	NA	NA	NO	NA
8 rs	NA	NA	OFF	NA	NA	NA	NO	NA

图 3-34 IO\_REGISTER 状态 I/O/TREGIS

## 2) Clock Summary

在 Place & Route 流程跑完之后会自动生成 Clock Summary 报告，该报告的数据来源是

pnr.adf 文件，用于显示用户设计网表中时钟资源的使用情况，各器件 PnR 的 Clock Summary 报告中包括 Global Clock、Regional Clock、Horizontal Clock 和 PLL Clock 页。根据工程实际是否用到对应时钟元件来决定是否显示对应的显示页，如下图因工程未使用 Horizontal Clock 则无 Horizontal Clock。

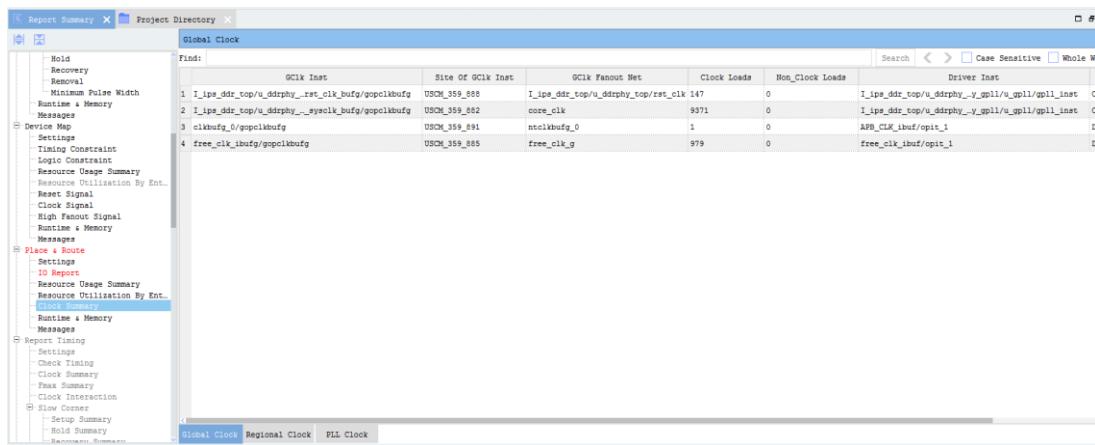
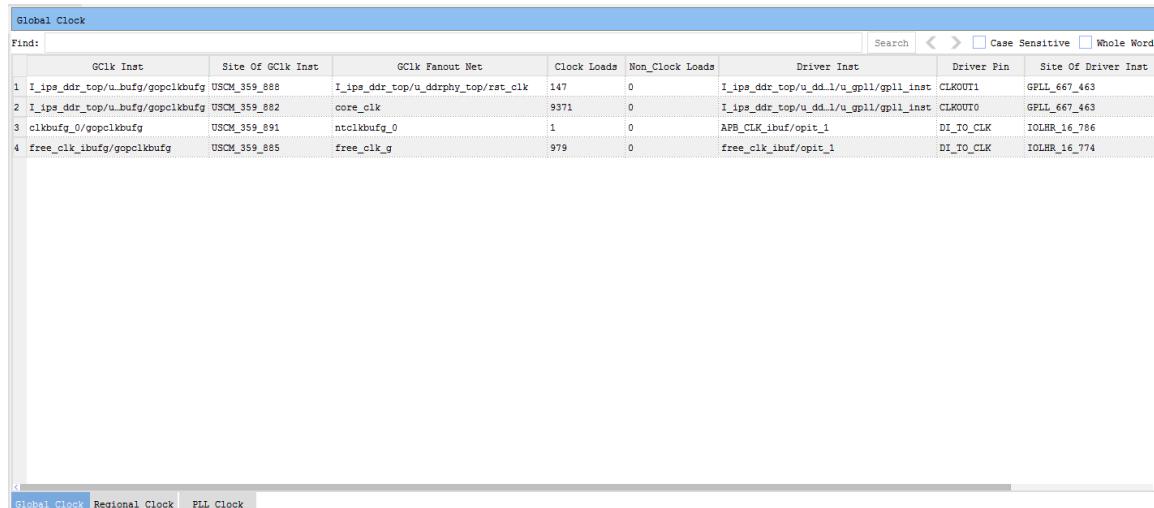


图 3-35 Clock Summary 界面显示

Global Clock 界面显示如下图所示：



GClk Inst	Site Of GClk Inst	GClk Fanout Net	Clock Loads	Non_Clock Loads	Driver Inst	Driver Pin	Site Of Driver Inst
I_ipm_ddr_top/u_bufg/gopclkbuffg	USCM_359_888	I_ipm_ddr_top/u_ddrphy_top/rst_clk	147	0	I_ipm_ddr_top/u_ddr_ddi/u_gpll/gpll_inst	CLKOUT1	GPLL_667_463
I_ipm_ddr_top/u_bufg/gopclkbuffg	USCM_359_882	core_clk	9371	0	I_ipm_ddr_top/u_ddr_ddi/u_gpll/gpll_inst	CLKOUT0	GPLL_667_463
ckbufg_0/gopclkbuffg	USCM_359_891	ntckbufg_0	1	0	APB_CLK_ibuf/opit_1	DI_TO_CLK	IOLHR_16_786
free_clk_ibuf/gopclkbuffg	USCM_359_885	free_clk_g	979	0	free_clk_ibuf/opit_1	DI_TO_CLK	IOLHR_16_774

图 3-36 Global Clock 界面显示

Global Clock 界面用于显示用户设计网表中使用的全局时钟资源的使用情况，主要显示驱动源和时钟负载的信息，变量说明如下：

- 1、GClk Inst： Global Clock Inst 的简称，显示全局时钟的名称；
- 2、Site Of GClk Inst： 显示全局时钟放置的位置信息；
- 3、GClk Fanout Net： 显示全局时钟输出端口所连接的 net 名称；
- 4、Clock Loads： 显示该时钟资源的时钟负载数量；其中连到负载资源时钟端口的成为时

钟负载；

5、Non\_Clock Loads：显示该时钟资源的非时钟负载数量；其中连到负载资源非时钟端口的成为非时钟负载；

6、Driver Inst：显示驱动该时钟资源的时钟源信息；

7、Driver Pin：显示驱动该时钟资源的端口信息；

8、Site Of Driver Inst：显示驱动该时钟资源的时钟源所放置的位置信息；

注：负载数量统计的是负载逻辑单元的数量，如果输出到同一个逻辑单元的不同端口，数量只统计一次；

Regional Clock、Horizontal Clock 界面显示与 Global Clock 界面显示是一样的，在这里不做特殊的说明，可参考上面的介绍。

PLL Clock 界面如下图所示：

PLL Clock									Search	<	>	Case Sensitive	Whole Wo:
Pll Inst	Site Of Pll Inst	Pin /	Net Of Pin	Clock Loads	Non_Clock Loads	Driver(Load) Inst	Driver(Load) Pin	Site Of Driver(Load) Inst					
1 I_ipo_ddr_top/_gpll/gpll_inst	GPLL_667_463	CLKIN1	ref_clk	-	-	refclk_ibuf/opit_2	DI_TO_CLK	IOLHP_664_474					
2 I_ipo_ddr_top/_gpll/gpll_inst	GPLL_667_307	CLKIN1	I_ipo_ddr_cclk [0]	-	-	I_ipo_ddr_r/goprclk	CLKOUT	RCKB_663_456					
3 I_ipo_ddr_top/_ppll/ppll_inst	PPLL_667_1	CLKIN1	I_ipo_ddr_cclk [1]	-	-	I_ipo_ddr_r/goprclk	CLKOUT	RCKB_663_150					
4 T_GTP_PPLL/ppll_inst	PPLL_7_613	CLKIN1	nt_CLKIN1	-	-	CLKIN1_ibuf/opit_1	DI_TO_CLK	IOLHR_16_756					
5 I_ipo_ddr_top/_gpll/gpll_inst	GPLL_667_463	CLKIN2	_GND17	-	-	GND_17	Z	CLMA_651_492					
6 I_ipo_ddr_top/_ppll/ppll_inst	PPLL_667_307	CLKIN2	_GND472	-	-	GND_470	Z	CLMS_645_349					
7 I_ipo_ddr_top/_ppll/ppll_inst	PPLL_667_1	CLKIN2	_GND14	-	-	GND_14	Z	CLMA_651_42					
8 I_ipo_ddr_top/_gpll/gpll_inst	GPLL_667_463	CLKOUT0	I_ipo_ddr_l_cikout0	9352	0	...	...	...					
9 T_GTP_PPLL/ppll_inst	PPLL_7_613	CLKOUT0	nt_CLKOUT0	0	1	BUFRROUTE_0	CLKIN	RCKB_9_764					
10 I_ipo_ddr_top/_gpll/gpll_inst	GPLL_667_463	CLKOUT1	I_ipo_ddr_l_cikout0	147	0	...	...	...					
11 I_ipo_ddr_top/_ppll/ppll_inst	PPLL_667_307	CLKOUTPHY	I_ipo_ddr_ioclk_fb	52	4	...	...	...					
12 I_ipo_ddr_top/_ppll/ppll_inst	PPLL_667_1	CLKOUTPHY	I_ipo_ddr_clk_p [1]	62	2	...	...	...					

图 3-37 PLL Clock 界面显示

PLL Clock 界面用于显示用户设计网表中使用的 PLL 资源的使用情况，主要显示驱动源和时钟负载的信息，变量说明如下：

1、PLL Inst：显示 PLL 资源的名称；

2、Site Of PLL Inst：显示 PLL 资源放置的位置信息；

3、Pin：显示 PLL 资源的输入和输出端口名称；

4、Net Of Pin：显示 PLL 资源的输入和输出端口所连 net 的名称；

5、Clock Loads：显示该时钟资源的时钟负载数量；其中连到负载资源时钟端口的成为时钟负载；

6、Non\_Clock Loads：显示该时钟资源的非时钟负载数量；其中连到负载资源非时钟端口的成为非时钟负载；

7、Driver(Load) Inst: 当 Pin 是输入端口时显示驱动该时钟资源的时钟源信息；当 Pin 是输出端口时并且 Load 数量为 1 (Load 数量 = Clock Loads 数量+ Non\_Clock Loads 数量) 时显示负载资源的信息，Load 数量大于 1 时，显示“...”；

8、Driver(Load) Pin: 当 Pin 是输入端口时显示驱动该时钟资源的端口信息；当 Pin 是输出端口时并且 Load 数量为 1 时显示负载资源的信息，Load 数量大于 1 时，显示“...”；

9、Site Of Driver(Load) Inst: 当 Pin 是输入端口时显示驱动该时钟资源的时钟源所放置的位置信息；当 Pin 是输出端口时并且 Load 数量为 1 时显示负载资源所放置的位置信息，Load 数量大于 1 时，显示“...”；

PLL Clock 报告内容中有一些需要注意事项：对于时钟输入 pin，它是没有 Fanout 值，显示是“-”；PLL 的输入或输出 pin 如果是悬空的，不会被统计到 clock summary 中；

### 3) clock\_utilization.txt

详细的时钟规划信息统计报告统一记录在 place\_route 目录 clock\_utilization.txt 内。

clock\_utilization.txt 中各表格说明如下：

**Clock Regions-Block Scope** 表：该型号芯片被划分为区域块数及各块区域的 Floorplan 坐标范围，与 UCE 或 PCE 工具里面的资源用的 Floorplan 一致，以芯片左下角为起点。

**Clock Regions-Clock Primitives** 表：统计芯片每个区域内可用时钟资源和逻辑资源信息（仅统计 CLM、DRM、APM），如 logos2 系列统计的是 CLK PAD、PLL PAD、RCKB、IOCKB、HCKB、MRCKB、CLMA、CLMS、DRM、APM 等资源信息数量。

其中 PGT180H 的 BRGCLKGATE、部分 IOCLKGATE 跨区域，不统计在内；IOCLKGATE 根据其驱动的 load（不包含 Clock Buffer）所在区域进行归类。IOCLKDIV 按照所对应的 IOCLKGATE 归类。

用户约束信息是布局前的时钟信息。统计是按照全局时钟、区域时钟、IO 时钟网络分类收集和统计，统计时将时钟系统电路按照 Source、Buffer 和 Load 分解为时钟子电路。Clock Source 和 Clock Buffer 不支持 Region 约束，支持物理约束。

因 Global/Regional/IO Constraint 显示界面格式一样，下面以 Global Constraint 举例说明。

**Global Clock Buffer Constraint Details** 表：主要收集 Clock Buffer-loads 的约束信息，各列含义：

1、Source Name: 时钟子电路时钟源的 Instance 的名称；

- 2、Source Pin: 时钟子电路时钟源输出时钟信号的 Pin 的名称;
- 3、Source-Buffer Net: 时钟子电路中时钟源 Source 和时钟核心元件 Buffer 之间连接的 Net 的名称;
- 4、Buffer Input Pin: 时钟子电路中时钟核心元件 Buffer 的 Input Pin 的名称, 是接收时钟信号的 Pin;
- 5、Buffer Name: 时钟子电路中时钟核心元件 Buffer Instance 的名称;
- 6、Buffer Output Pin: 时钟子电路中时钟核心元件 Buffer 的 Output Pin 的名称, 是输出时钟信号的 Pin;
- 7、Buffer-Load Net: 时钟子电路中时钟核心元件 Buffer 的 Output Pin 与 Load 相互连接的 Net 的名称;
- 8、Clock Region Of Buffer Site: 若用户对 Buffer 设置了约束, 则显示约束位置的区域编号, 若未约束则显示 “---” ;
- 9、Buffer Site: 若用户对 Buffer 设置了约束, 则显示具体约束位置, 即资源所在的 Grid Device 名称及坐标;
- 10、IO Load Clock Region: Load 为 IO 类型的区域限制, 受控于驱动的时钟核心元件 Buffer 及用户约束;
- 11、Non-IO Load Clock Region: Load 不为 IO 类型的区域限制, 受控于驱动的时钟核心元件 Buffer 及用户约束;
- 12、Clock Loads: Load 是将 Buffer 传过来信息用作时钟信号, 统计其数量;
- 13、Non-Clock Loads: Load 是将 Buffer 传过来信息用作普通数据信号, 统计其数量。

**Global Clock Source Constraint Details 表:** 主要收集 Clock Source 的约束信息, 各列含义:

- 1、Source Name : 时钟子电路时钟源的 Instance 的名称;
- 2、Source Pin: 时钟子电路时钟源输出时钟信号的 Pin 的名称;
- 3、Source-Load Net: 时钟子电路时钟源输出时钟信号的 Pin 和它的 Load 之间连接的 Net 的名称;
- 4、Clock Region Of Source Site: 若用户对 Source 设置了约束, 则显示约束位置的区域编号, 若未约束则显示 “---” ;

5、Source Site: 若用户对 Source 设置了约束，则显示具体约束位置，即资源所在的 Grid Device 名称及坐标；

6、Clock Buffer Loads: 统计 Source 后面连接 Clock Buffer 数目 (PGT180H 器件会将 USCM 复制 4 份，但 Global Clock Buffer Constraint Details 仅显示原 Clock Buffer，实际布局会根据 load 所在位置保留对应 USCM，多余的则删掉)；

7、Non-Clock Buffer Loads: 统计 Source 后面连接的非 Clock Buffer Instance 的数目。

时钟布局信息为时钟规划阶段的布局信息，非最终布局信息。统计方式跟用户约束信息一致。下面以 Global Clock 举例。

**Device Cell Placement Summary for Global Clock Buffer 表：**主要收集 Clock Buffer 的时钟自动布局的规划信息，各列含义可参照 Global Clock Buffer Constraint Details 表格说明。

其中 IO Load Clock Region、Non-IO Load Clock Region 根据 Buffer 所布局位置及其驱动 load 确定的驱动范围。

**Device Cell Placement Summary for Global Clock Source 表：**主要收集 Clock Source 的时钟自动布局的规划信息，各列含义可参照 Global Clock Source Constraint Details 表格说明。

若时钟规划失败会打印详细规划失败信息，包含失败原因、解决失败问题的方法。

**Placement Failed Information About The Constraint Details Of Buffer 表：**规划失败的时钟核心元件 Buffer 信息，各列含义可参照 Device Cell Placement Summary for Global Clock Buffer 表格说明。

**Placement Failed Information About The Load Of Source Instance 表：**规划失败的时钟源 Instance 信息，各列含义：

1、Driver Pin: 时钟子电路时钟源输出时钟信号的 Pin 的名称；

2、Driver-Load Net: 时钟子电路时钟源输出时钟信号的 Pin 和它的 Load 之间连接的 Net 的名称；

3、Load Instance: 时钟子电路中时钟核心元件 Buffer Instance 的名称；

4、Load Unit: 时钟子电路中时钟核心元件 Buffer Instance 的实现；

5、Load Clock Region: 若用户对 Buffer 设置了约束，则显示约束位置的区域编号，若未约束则显示“---”；

**Placement Failed Information About The Load Of Buffer Instance:** 规划失败的时钟核心元件 Buffer 驱动 Load 信息，各列含义：

1、Driver Pin: 时钟子电路中时钟核心元件 Buffer 的 Output Pin 的名称，是输出时钟信号的 Pin；

2、Driver-Load Net: 时钟子电路中时钟核心元件 Buffer 的 Output Pin 与 Load 相互连接的 Net 的名称；

3、Load Instance: 时钟子电路中时钟核心元件 Buffer Instance 驱动的 load；

4、Load Unit: 时钟子电路中时钟核心元件 Buffer Instance 驱动 load 的实现；

5、Load Clock Region: Load 的区域限制，受控于驱动的时钟核心元件 Buffer 及用户约束。

### 3.5 Report Timing

#### 3.5.1 Report Timing 设置

Report\_Timing\_settings 介绍如下：Report Timing 的选项设置，如下图所示：

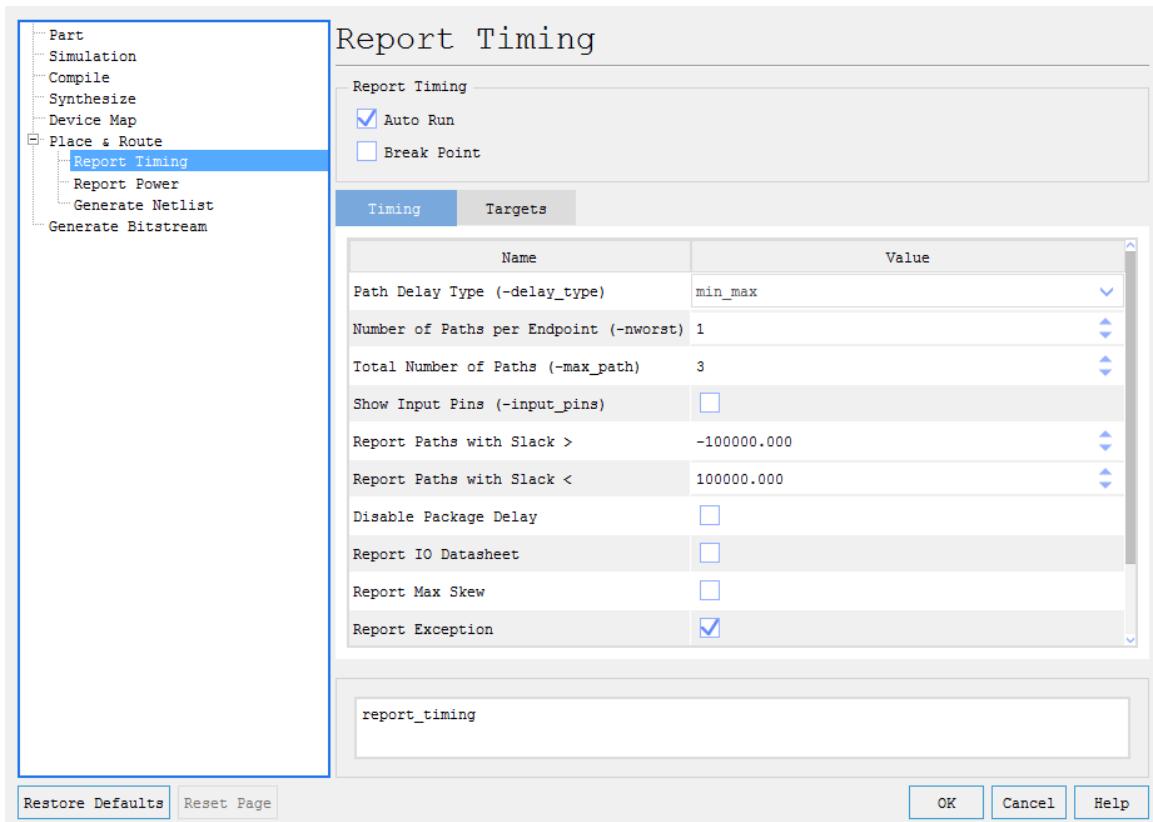


图 3-38 Report Timing 的选项设置

Report 配置选项：

### Timing Analysis

**【Path Delay Type (-delay\_type)】** 设置时序分析的类型，可选项有 "max", "min", "min\_max"。选择 max 将对时序路径只进行 setup 分析，选择 min 将对时序路径只进行 hold 分析，选择 min\_max 则会对时序路径进行 setup 和 hold 分析。默认值为 min\_max。

**【Number of Paths per Endpoint (-nworst)】** 设置每个时序路径终点报告的最大路径数目。默认值为 1。

**【Total Number of Paths (-max\_path)】** 设置时序报告的最大路径数目。时序分析工具按照 slack 对时序路径进行排序，最大路径数目不会超过该项设置数目，默认值为 3。

**【Show Input Pins (-input\_pins)】** 显示输入 Pins。选择该项，则 timing path 中会显示输入端口的信息。不选择该项，则 timing path 中不显示输入端口的信息，默认不选中该选项。

**【Report Paths with Slack >】** 设置报告的 slack 最小范围。slack 小于该项设置的时序路径将不予显示。默认值为 -100000。

**【Report Paths with Slack <】** 设置报告的 slack 最大范围。slack 大于该项设置的时序路径将不予显示。默认值为 100000。

【Report Paths with Logic Levels >】设置报告的时序路径 Logic Levels 最小值。setup/recovery 时序分析中 Logic Levels 小于等于该值的时序路径将不予显示。该设置不影响 hold/removal 分析时序路径。当设置的 logic level > -1 时，时序分析会优先报出 logic level 大的时序路径，而不一定是 slack 差的路径，时序报告也会按照 logic level 的大小排序，此时的时序结果不能作为时序是否收敛的判断标准。（仅在 TA 中可用）。

【Disable Package Delay】设置时序报告是否计算封装（package pin）的延迟。如果选中该选项，时序报告路径中将不会包括封装延迟，默认不选中该选项。

【Report IO Datasheet】设置时序报告是否报告 IO 的时序特性，包括 Input Ports Setup/Hold, Output Ports Clock-to-out, Combinational Delays, Setup/Hold times for input bus, Max/Min delays for output bus。默认不选中该选项。

【Report Max Skew】设置时序报告是否报告用户约束的 set\_max\_skew 命令的时序特性。如果没有选中该选项，那么即使有 set\_max\_skew 这条命令约束，最后的时序报告也不会报告相关内容。默认不选中该选项。

【Report Exception】设置时序报告是否显示设置了 set\_multicycle\_path、set\_max\_delay、set\_min\_delay 和 set\_false\_path 这四种 Exception 约束的路径的独立报告。默认勾选该选项。

【Limit Paths by Clock Group】设置时钟组的名字，时序报告中只会显示已设置时钟组的时序路径。（仅在 TA 中可用）

\*注：执行 Place & Route 且 Placement Iterations 大于 1 时（即：多种子方式运行），系统会在后台执行 Report Timing，并根据生成报告的 slack 值选用最优结果；如果 Path Delay Type (-delay\_type) 选项设置为 min，那么不会输出 slack 相关信息，Place & Route 运行结果未定义。

【Skip DB version check】若旧版工程有运行至 pnr 或 report timing，则新版软件在打开旧版工程，勾选此选项后，不必 rerun all 就可仅运行 report timing 产生时序报告，默认不选中该选项（目前支持该选项的器件仅支持 PG2T390H、PGL25G、PG2L100H、PGT180H）

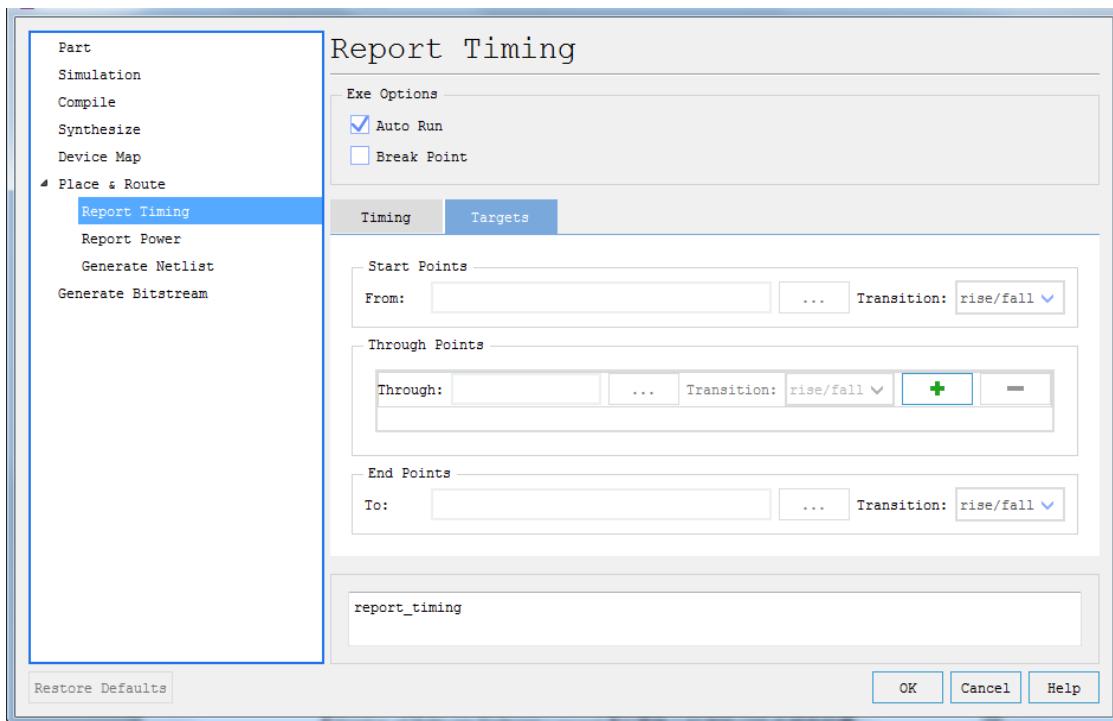


图 3-39 Report Timing 的选项设置

【From】 【Through】 【To】可以指定期序报告中时序路径的 from/through/to 节点。默认值均为空。

### 3.5.2 Report Timing 报告

时序报告有如下项目：Settings, Check Timing, Clock Summary, Fmax Summary, Clock Interaction, Timing Path Report, Max Skew Report, Exception Report, IO Datasheet, Runtime & Memory 和 Messages。其中 Timing Path Report 包括 Slow Corner 和 Fast Corner 两种极端情况下的 Timing Path 报告，这两个 Corner 的 Timing Path 报告都包括了 Setup Summary, Hold Summary, Recovery Summary, Removal Summary, Minimum Pulse Width Summary, Clock Network 和 Worst Case Timing Path，其中 Worst Case Timing Path 又以 Setup, Hold, Recovery, Removal, Minimum Pulse Width 做了区分，Report Timing 对应的目录树如下图所示，

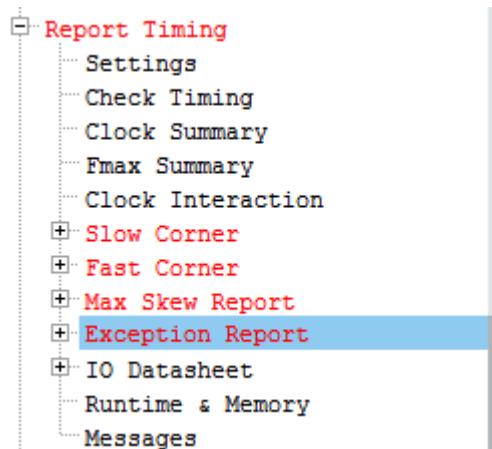


图 3-40 Timing Report 目录

若存在时序要求不满足的情况，目录树上面的节点会对应标红显示，以便于快速定位查看。

### 1. Settings

显示出当前时序分析的设置，具体项目如下图：

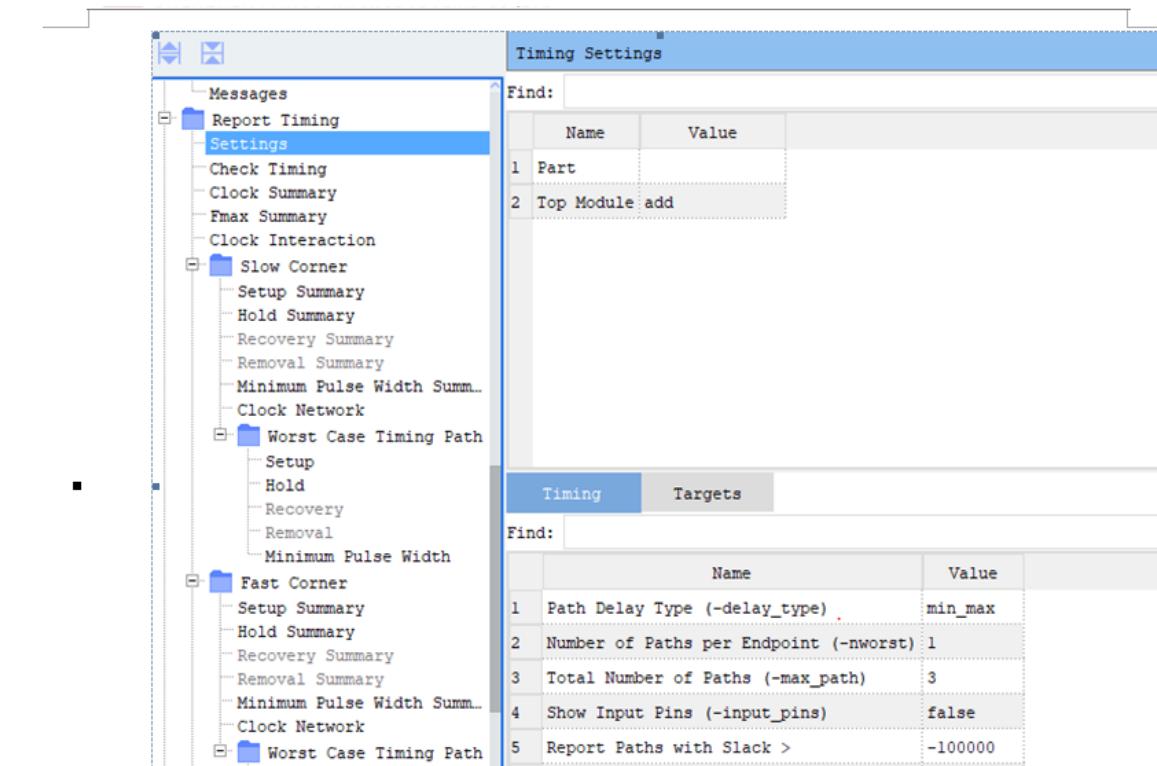


图 3-41 Report Timing 设置

### 2. Check Timing

Check Timing 的作用是用来在时序分析进行之前检查当前设计中时序约束的问题，如图下图所示：

Check Timing Summary

	Check	Number of Issues
1	no_clock	0
2	virtual_clock	1
3	unexpandable_clocks	0
4	no_input_delay	0
5	no_output_delay	0
6	partial_input_delay	0
7	partial_output_delay	0
8	io_min_max_delay_consistency	4
9	input_delay_assigned_to_clock	0
10	loops	0
11	latches	0

virtual\_clock

Issues
No virtual clock was found.

图 3-42 Check Timing Report Summary

Check timing 的主要检查条目和介绍如下：

1) no\_clock

该项 check 主要用来检查时序器件的 clock 端口是否有时钟驱动（clock propagate），如果时序器件时钟端口没有时钟数据，时序分析就无法对该时序器件进行 setup、hold 等时序分析；

2) virtual\_clock

该项 check 用来检查 design 中是否创建虚拟时钟，或者设置了虚拟时钟但是并没有被使用。

3) unexpandable\_clocks

该选项 check 异步时钟之间是否可以在 1000 个周期之内找到公共周期。

4) no\_input\_delay

该项检查输入端口没有设置 set\_input\_delay。

#### 5) no\_output\_delay

该查输出端口没有设置 set\_output\_delay。

#### 6) partial\_input\_delay、partial\_output\_delay

该项检查输入端口 delay 约束存在部分的设置, 某个输入端口仅存在 set\_input\_delay -max 或者 set\_input\_delay -min。给端口部分设置 delay 约束会导致时序分析工具无法对该 path 进行全面的时序分析。如果对端口设置的约束 delay 不指定-min 或者-max, 时序分析工具会假定端口具有相同的 min 和 max delay。

#### 7) partial\_output\_delay

该项检查输出端口 delay 约束存在部分的设置, 某个输出端口仅存在 set\_output\_delay -max 或者 set\_output\_delay -min。给端口部分设置 delay 约束会导致时序分析工具无法对该 path 进行全面的时序分析。如果对端口设置的约束 delay 不指定-min 或者-max, 时序分析工具会假定端口具有相同的 min 和 max delay。

#### 8) io\_min\_max\_delay\_consistency

该项检查 IO delay 的约束设置一致性, 对 set\_input\_delay 和 set\_output\_delay 的设置进行检查, 如果 IO delay 的 min delay 大于等于 max delay, 报告 warning。对于没有指定 min、max 的 IO delay 约束, 软件默认 min、max delay 相等, 也需要进行报告。

#### 9) input\_delay\_assigned\_to\_clock

该选项检查是否把 Input delay 设置到时钟端口上, 在这种情况下, input delay 会被忽略。

#### 10) Loops

检查 design 里是否有组合环路, 组合环路可能没有被正确分析。

#### 11) latches

检查 design 里是有 latch。

### 3. Clock Summary

显示时钟的基本信息。

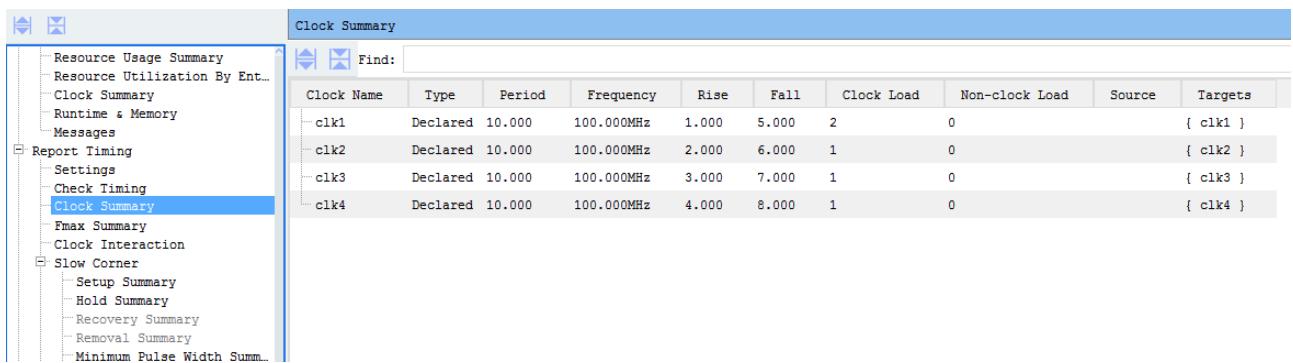


图 3-43 Clock Summary

#### 4. Fmax Summary

显示时钟的最大工作频率，时钟的 Fmax 是根据同时钟域中 slack 最小的那条 timing path 来计算的，跨时钟域的 timing path 不会影响到 Fmax 结果。由于 Fmax 的计算不会考虑跨时钟域的 timing path，因此该结果只作为时钟性能的参考，不能作为时序收敛的评判依据。

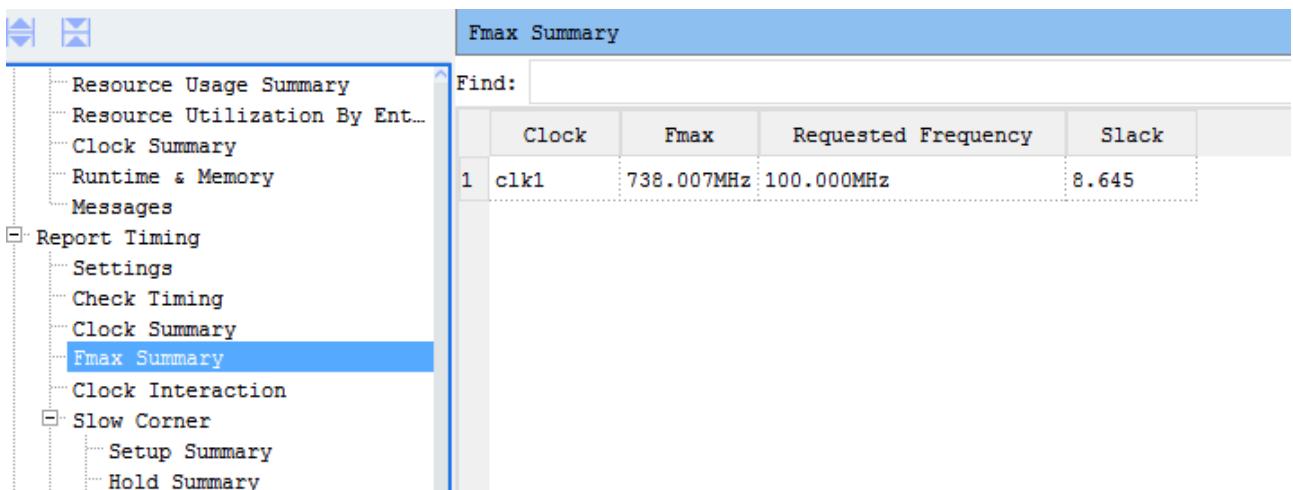


图 3-44 Fmax Summary

若 clock 的最大工作频率不满足用户约束，即  $F_{max} < \text{Requested Frequency}$ ，则该 clock 所在行会标红显示。

#### 5. Clock Interaction Report

Clock Interaction Report 用于报告 design 中时钟之间的关系和时序约束情况，如下图所示：

Clock Interaction						
Find:				Search	<	>
	Launch Clock	Capture Clock	Common Primary Clock	Inter-Clock Constrains	WNS(ns)	TNS(ns)
1	CLKA	CLKA	YES	Timed	7.210	0.000
2	CLKA	CLKB	NO	Timed(unsafe)	7.758	0.000
3	CLKA	CLKC	NO	Partial False Path(unsafe)	6.046	0.000
4	CLKB	CLKC	NO	False Path		
5	CLKA	CLKD	NO	Asynchronous Groups		
6	CLKC	CLKD	NO	Asynchronous Groups		
7	CLKD	CLKD	YES	Partial False Path	-7.528	-29.121
8	REF_CLK(p1l_inst0/I_GTP_PLL/p1l/CLKOUT3_Inferred	CLKD	NO	Asynchronous Groups		
9	CLKA	REF_CLK(p1l_inst0/I_GTP_PLL/p1l/CLKOUT3_Inferred	NO	Phase Error(unsafe)	-0.263	-0.523

图 3-45 Clock Interaction Report

报告中的主要内容如下：

**Launch Clock:** 表示发射时钟。

**Capture Clock:** 表示接收时钟。

**Common Primary Clock:** 表示发射时钟和接收时钟的主时钟是否一致。

**Inter-Clock Constrains:** 表示发射时钟和就收时钟之间的关系，有如下几种关系：

- **Timed:** 表示发射和接收时钟是同步时钟，两者之间的路径被安全约束。
- **Timed(unsafe):** 表示发射和接收时钟是异步时钟，两者之间的路径没有约束。
- **False Path:** 表示发射和接收时钟之间的所有路径都通过命令 `set_false_path` 定义为了伪路径。
- **Partial False Path:** 表示发射和接收时钟是同步时钟，两者之间有部分路径被用户使用命令 `set_false_path` 定义为了伪路径。
- **Partial False Path(unsafe):** 表示发射和接收时钟是异步时钟，两者之间有部分路径被用户使用命令 `set_false_path` 定义为了伪路径。
- **Asynchronous Groups:** 表示发射和接收时钟之间的所有路径都通过命令 `set_clock_groups` 定义为了伪路径。
- **Phase Error(unsafe):** 表示发射和接收时钟中至少有一个时钟是 generated clock，且该 generated clock 和另一个 clock 之间的相位关系无法确定，说明这两个时钟之间的时序路径是不安全的。
- **Max Delay Datapath Only:** 表示发射和接收时钟之间的所有路径都通过 `set_max_delay -datapath_only` 约束。

**WNS:** 表示发射和接收时钟之间所有 setup 和 recovery path 的最差 slack。

**TNS:** 表示发射和接收时钟之间所有 setup 和 recovery path 所有小于 0 的 slack 值的和。

**Failing End Point(TNS):** 表示发射和接收时钟之间所有 setup 和 recovery path 中 slack 小

于 0 的 end point 个数。

Total End Point(TNS): 表示发射和接收时钟之间所有 setup 和 recovery path 中 end point 的个数。

WHS: 表示发射和接收时钟之间所有 hold 和 removal path 的最差 slack。

THS: 表示发射和接收时钟之间所有 hold 和 removal path 所有小于 0 的 slack 值的和。

Failing End Point(THS): 表示发射和接收时钟之间所有 hold 和 removal path 中 slack 小于 0 的 end point 个数。

Total End Point(THS): 表示发射和接收时钟之间所有 hold 和 removal path 中 end point 的个数。

## 6. Clock Network Report

Clock Network Report 用于报告用户 design 中的时钟路径和路径延时，位于 Slow Corner 和 Fast Corner 目录下方，如下图所示：

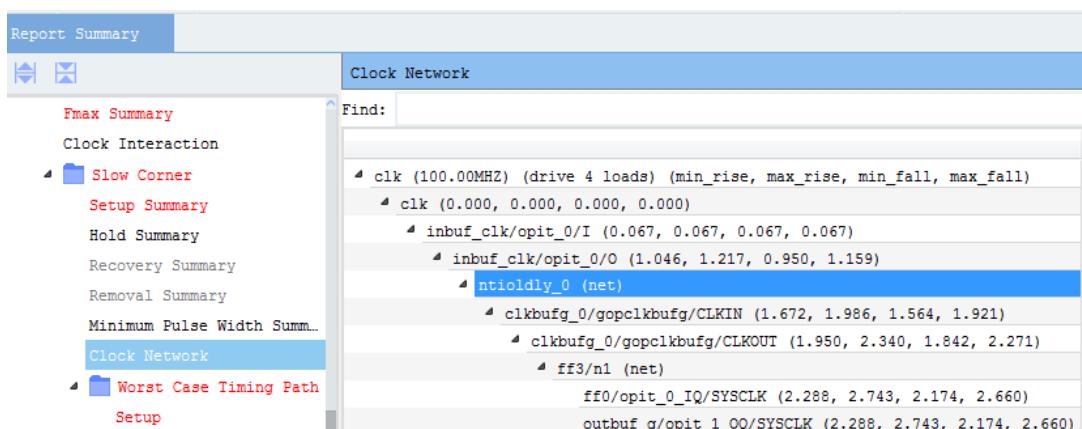
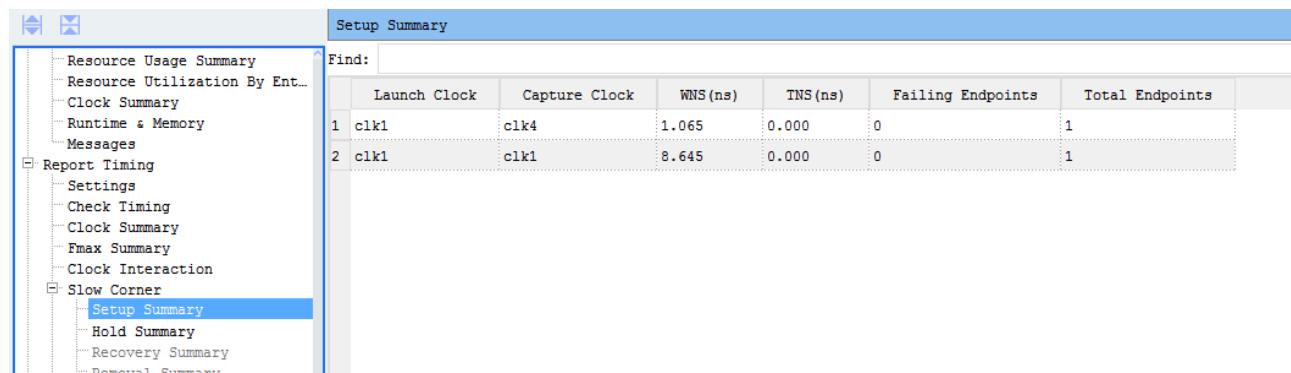


图 3-46 Clock Network Report

## 7. Timing Path Report

### ➤ Setup Summary

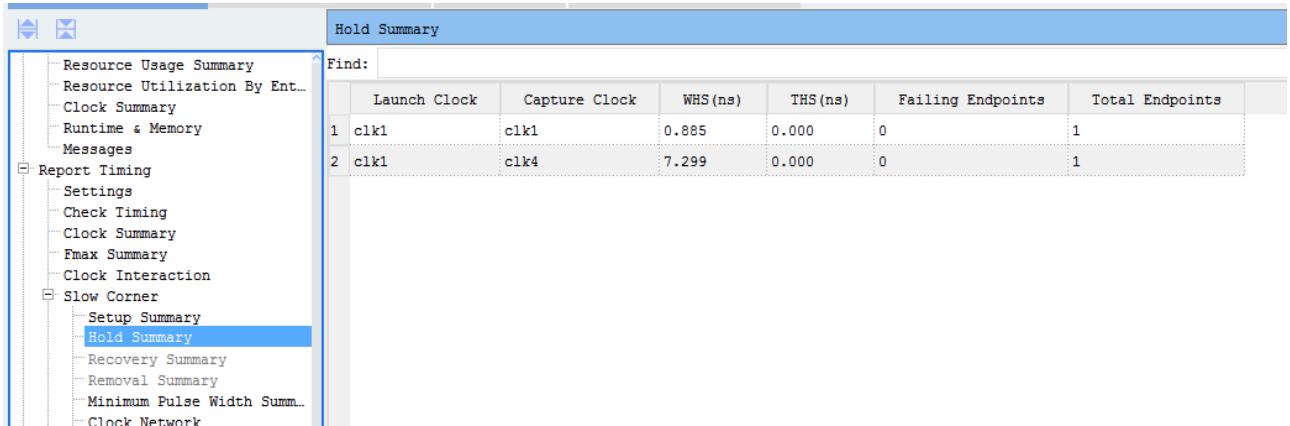


	Launch Clock	Capture Clock	WNS(ns)	TNS(ns)	Failing Endpoints	Total Endpoints
1	clk1	clk4	1.065	0.000	0	1
2	clk1	clk1	8.645	0.000	0	1

图 3-47 Setup Summary

上图中 Launch Clock 为定义的发射时钟的名字, Capture Clock 为定义的接收时钟的名字; Slack 是指这两个 clock 所在的 setup path group 中 slack 的最小值; Total Negative Slack(TNS) 表示这两个 clock 所在的 setup path group 中所有 slack 小于 0 的和; Failing End Point 表示这两个 clock 驱动的 End Point 有多少个存在时序不收敛; Total End Point 表示这两个 clock 驱动了多少个 End Point。

#### ➤ Hold Summary

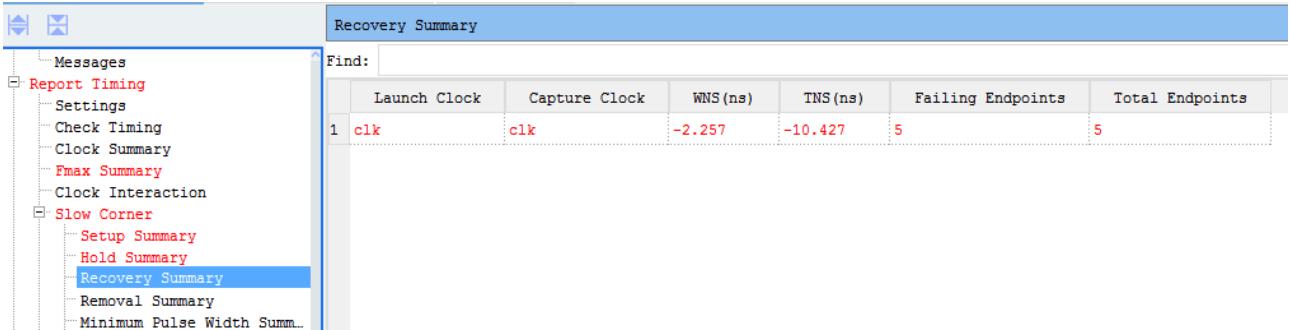


The screenshot shows the Hold Summary report interface. The left sidebar has a tree view with 'Report Timing' expanded, showing 'Hold Summary' selected. The main area is titled 'Hold Summary' with a table:

	Launch Clock	Capture Clock	WHS(ns)	THS(ns)	Failing Endpoints	Total Endpoints
1	clk1	clk1	0.885	0.000	0	1
2	clk1	clk4	7.299	0.000	0	1

图 3-48 Hold Summary

#### ➤ Recovery summary

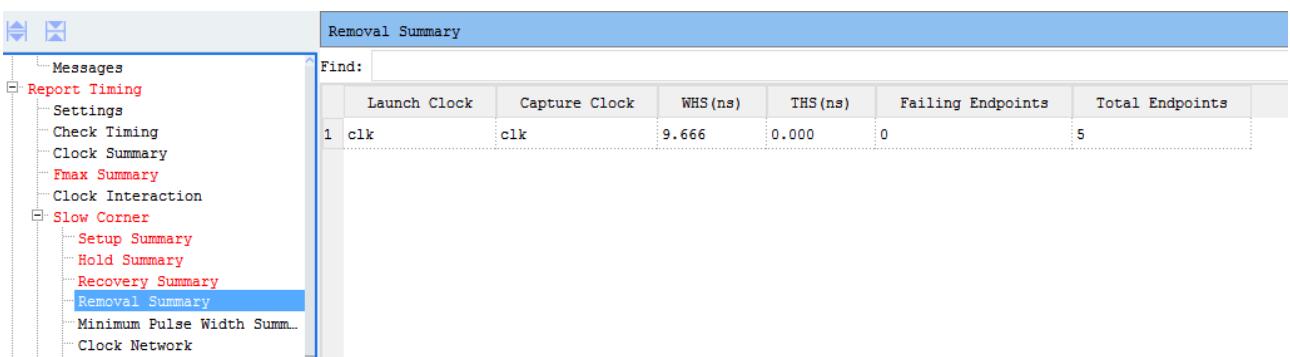


The screenshot shows the Recovery Summary report interface. The left sidebar has a tree view with 'Report Timing' expanded, showing 'Recovery Summary' selected. The main area is titled 'Recovery Summary' with a table:

	Launch Clock	Capture Clock	WNS(ns)	TNS(ns)	Failing Endpoints	Total Endpoints
1	clk	clk	-2.257	-10.427	5	5

图 3-49 Recovery Summary

#### ➤ Removal summary

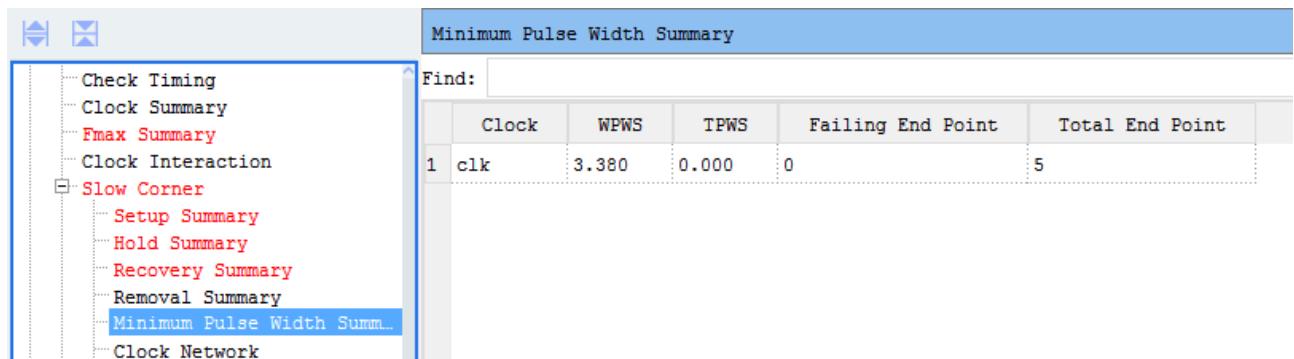


The screenshot shows the Removal Summary report interface. The left sidebar has a tree view with 'Report Timing' expanded, showing 'Removal Summary' selected. The main area is titled 'Removal Summary' with a table:

	Launch Clock	Capture Clock	WHS(ns)	THS(ns)	Failing Endpoints	Total Endpoints
1	clk	clk	9.666	0.000	0	5

图 3-50 Removal Summary

➤ Minimum Pulse Width Summary

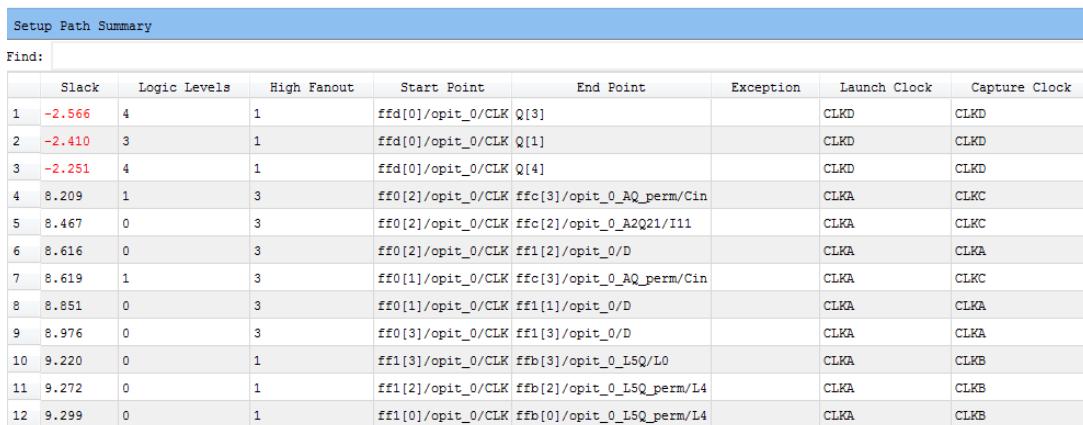


Clock	WPWS	TPWS	Failing End Point	Total End Point
1 clk	3.380	0.000	0	5

图 3-51 Minimum Pulse Width Summary

➤ Worst Case Timing path

其中 Worst Case timing path 分成了 Setup、Hold、Recovery、Removal、Minimum Pulse Width 几种类型，报告中分别列出了对应 path 的 summary 信息和详细信息。其中，Setup、Hold、 Recovery、Removal 的报告类似，下面以 Setup 和 Minimum Pulse Width 的 Path 为例进行说明，具体见下图：



Slack	Logic Levels	High Fanout	Start Point	End Point	Exception	Launch Clock	Capture Clock
-2.566	4	1	ffd[0]/opit_0/CLK	Q[3]	CLKD	CLKD	
-2.410	3	1	ffd[0]/opit_0/CLK	Q[1]	CLKD	CLKD	
-2.251	4	1	ffd[0]/opit_0/CLK	Q[4]	CLKD	CLKD	
8.209	1	3	ff0[2]/opit_0/CLK	ffc[3]/opit_0_AQ_perm/Cin	CLKA	CLKC	
8.467	0	3	ff0[2]/opit_0/CLK	ffc[2]/opit_0_A2Q21/I11	CLKA	CLKC	
8.616	0	3	ff0[2]/opit_0/CLK	ff1[2]/opit_0/D	CLKA	CLKA	
8.619	1	3	ff0[1]/opit_0/CLK	ffc[3]/opit_0_AQ_perm/Cin	CLKA	CLKC	
8.851	0	3	ff0[1]/opit_0/CLK	ff1[1]/opit_0/D	CLKA	CLKA	
8.976	0	3	ff0[3]/opit_0/CLK	ff1[3]/opit_0/D	CLKA	CLKA	
9.220	0	1	ff1[3]/opit_0/CLK	ffb[3]/opit_0_LSQ/L0	CLKA	CLKB	
9.272	0	1	ff1[2]/opit_0/CLK	ffb[2]/opit_0_LSQ_perm/L4	CLKA	CLKB	
9.299	0	1	ff1[0]/opit_0/CLK	ffb[0]/opit_0_LSQ_perm/L4	CLKA	CLKB	

图 3-52 Setup Path Summary Report

上图中 Setup Path Summary 从左至右依次有如下信息：Slack、Logic Levels、High Fanout、Start Point、End Point、Exception、Launch Clock、Capture Clock、Clock Edges、Clock Skew、Launch Clock Delay、Capture Clock Delay、Clock Pessimism Removal、Requirement、Data Delay、Logic Delay、Route Delay。在 Path Summary 中 slack 小于 0 的行会标红，且默认情况下是按 slack 从小到大的顺序排列显示的。在 Path Summary 的下方为所选 path 的详细信息，包括该条 path 是否满足时序要求，path 的 Data arrival time 和 required time。

Setup Path Summary

	Slack	Logic Levels	High Fanout	Start Point	End Point	Exception
1	-8.142	2	4	tmp[0]/opit_0_inv_L5Q_perm/CLK	out[0]	MaxDelay Path 2.000ns c
2	7.151	1	2	tmp[2]/opit_0_inv_A2Q21/CLK	tmp[7]/opit_0_inv_AQ/Cin	c
3	7.238	1	2	tmp[2]/opit_0_inv_A2Q21/CLK	tmp[6]/opit_0_inv_A2Q21/Cin	c
4	7.360	0	2	tmp[2]/opit_0_inv_A2Q21/CLK	tmp[4]/opit_0_inv_A2Q21/Cin	c
5	7.938	0	2	tmp[2]/opit_0_inv_A2Q21/CLK	tmp[2]/opit_0_inv_A2Q21/I12	c
6	7.991	0	3	tmp[2]/opit_0_inv_A2Q21/CLK	tmp[2]/opit_0_inv_A2Q21/I01	c

Path #1: setup slack is -8.142(VIOLATED)

Data arrival time is 12.592

Find:

	Location	Delay Type	Incr	Path	Trans	Logical Resource
12	CLMA_147_144/Q2	tco	0.374	9.009	f	tmp[0]/opit_0_inv_L5Q_perm/Q
13		net (fanout=4)	0.418	9.507		nt_out[0]
14	IOL_154_146/D0					outobuf[0]/opit_1_IIN
15	IOL_154_146/DQ	td	0.451	9.958	f	outobuf
16		net (fanout=1)	0.000	9.958		outobuf
17	IOBS_156_145/D0					outobuf
18	IOBS_156_145/PAD	td	2.518	12.476	f	outobuf
19		net (fanout=1)	0.116	12.592		out[0]
20	AB15				f	out[0] (

Right-click context menu for row 19 (AB15):  
 Undo Sort  
 Copy  
 Select All  
 Locate In File Editor  
 Locate In Device View  
 Export  
 Settings

Data required time is 4.450

Find:

	Location	Delay Type	Incr	Path	Trans	Logical Resource
1	max_delay		2.000	2.000		
2	clock source latency		6.000	8.000		
3	clock pessimism		0.000	8.000		

图 3-53 Detail timing Path Report

上图中的 Start Point、End Point 以及详细 path 中的 logical resource，可以右键映射跳转到相应的 RTL 源文件和 DE 中查看。

Minimum Pulse Width Path Summary

	Slack	Actual Width	Require Width	Clock	Type	Location	Pin
1	3.380	4.000	0.620	clk	High Pulse Width	CLMA_147_144/CLK tmp[0]/opit_0_inv_L5Q_perm/CLK	
2	3.380	4.000	0.620	clk	High Pulse Width	CLMS_147_145/CLK tmp[2]/opit_0_inv_A2Q21/CLK	
3	3.380	4.000	0.620	clk	High Pulse Width	CLMS_147_145/CLK tmp[4]/opit_0_inv_A2Q21/CLK	
4	3.380	4.000	0.620	clk	High Pulse Width	CLMS_147_151/CLK tmp[6]/opit_0_inv_A2Q21/CLK	
5	3.380	4.000	0.620	clk	High Pulse Width	CLMS_147_151/CLK tmp[7]/opit_0_inv_AQ/CLK	
6	5.380	6.000	0.620	clk	Low Pulse Width	CLMA_147_144/CLK tmp[0]/opit_0_inv_L5Q_perm/CLK	
7	5.380	6.000	0.620	clk	Low Pulse Width	CLMS_147_145/CLK tmp[2]/opit_0_inv_A2Q21/CLK	
8	5.380	6.000	0.620	clk	Low Pulse Width	CLMS_147_145/CLK tmp[4]/opit_0_inv_A2Q21/CLK	
9	5.380	6.000	0.620	clk	Low Pulse Width	CLMS_147_151/CLK tmp[6]/opit_0_inv_A2Q21/CLK	
10	5.380	6.000	0.620	clk	Low Pulse Width	CLMS_147_151/CLK tmp[7]/opit_0_inv_AQ/CLK	

图 3-54 Minimum Pulse Width Path Report

Minimum Pulse Width Path Report 和 Setup Path Report 不同，所有信息都已经在 summary 中列出，不需要详细信息列表，这些信息包括 Slack、Actual Width、Required Width、Clock、Type、Location 和 Pin。

另外，在 Flow Summary 里，也可以看到时序是否违例的信息，若时序都满足，显示"All Constraints Met"; 若存在时序不满足，则显示具体的不满足项（如下图中标红部分）。点击可链接到详细报告。如下图所示：

Timing Constraints: All Constraints Met

图 3-55 时序是否违例显示

## 8. IO Datasheet

IO Datasheet 主要反映 IO 的时序特性，包括：

### ➤ Input Ports Setup/Hold

Input Ports Setup/Hold									
Find: Search < > Case Sensitive Whole Words									
Reference Clock	Input Port	IO Reg Type	Delay Type	Setup(ns) to Clk (Edge)	Setup Process Corner	Hold(ns) to Clk (Edge)	Hold Process Corner	Internal Clock	
1 CLKIN	A[0]			0.526 (r)	SLOW	1.717 (r)	SLOW		
2 CLKIN	A[0]			0.794 (r)	FAST	0.701 (r)	FAST		
3 CLKIN	A[1]			0.558 (r)	SLOW	1.759 (r)	SLOW		
4 CLKIN	A[1]			0.847 (r)	FAST	0.742 (r)	FAST		

CLKIN -> A[0]									
Setup(0.526) = arr_t(5.949) - launch_clk(0.000) - input_delay(2.000) - req_t(8.423) + cap_clk(5.000) Hold(1.717) = req_t(4.409) - cap_clk(0.000) - arr_t(3.692) + launch_clk(0.000) + input_delay(1.000)									
Find: Search < > Case Sensitive Whole Words									
Location	Delay Type	Incr	Path	Trans	Logical R	Location	Delay Type	Incr	Path
1 Data Path						1 Data Path			
2 Clock CLKIN (rising edge)		0.000	0.000	r		2 Clock CLKIN (rising edge)		0.000	0.000
3 Input external delay		2.000	2.000	f		3 Input external delay		1.000	1.000
4 F2		0.000	2.000	f	A[0] (port)	4 F2		0.000	1.000
5 net (fanout=1)	0.085	2.085			A[0]	5 net (fanout=1)	0.085	1.085	
6 IOBS_0_112/PAD	A_ibuf[0]/opit_0/I (go					6 IOBS_0_112/PAD	A_ibuf[0]/opit_0/I (g		
7 IOBS_0_112/DIN	td	1.367	3.452	f	A_ibuf[0]/opit_0/O (go	7 IOBS_0_112/DIN	A_ibuf[0]/opit_0/O (g		
8	net (fanout=1)	0.000	3.452		A_ibuf[0]/ntD	8	net (fanout=1)	0.000	1.998
9 IOL_7_113/DI					A_ibuf[0]/opit_1/IN (g	9 IOL_7_113/DI	A_ibuf[0]/opit_1/IN (		
10 IOL_7_113/RX_DATA_DD	td	0.127	3.579	f	A_ibuf[0]/opit_1/OUT (	10 IOL_7_113/RX_DATA_DD	A_ibuf[0]/opit_1/OUT (		
11 net (fanout=3)	1.985	5.564			nt_A[0]	11 net (fanout=3)	1.612	3.692	nt_A[0]

图 3-56 Input Ports Setup/Hold

如果 Input Port 最终连接到一个时序单元，Input Ports Setup/Hold 将分 clock 分 corner 报该 Input Port 的 Setup/Hold 要求，该要求是考虑时序单元的 Setup/Hold 以及 Port 和 Clock 端口连接到时序单元的 Delay 反算回来。

### ➤ Output Ports Clock-to-out

Output Ports Clock-to-out									
Find: Search < > Case Sensitive Whole Words									
Reference Clock	Output Port	IO Reg Type	Delay Type	Max Clk (Edge) to port(ns)	Max Process Corner	Min Clk (Edge) to port(ns)	Min Process Corner	Internal Clock	
1 CLKIN	O2[0]			9.758 (r)	SLOW	7.670 (r)	SLOW		
2 CLKIN	O2[0]			6.637 (r)	FAST	5.319 (r)	FAST		
3 CLKIN	O2[1]			10.054 (r)	SLOW	7.930 (r)	SLOW		
4 CLKIN	O2[1]			6.873 (r)	FAST	5.499 (r)	FAST		
5 CLKIN	O2[2]			9.848 (r)	SLOW	7.750 (r)	SLOW		

CLKIN -> O2[0]									
Max Clk (r) to port(9.758) = TCO(9.708) + uncertainty(0.050) Min Clk (r) to port(7.670) = TCO(7.720) - uncertainty(0.050)									
Find: Search < > Case Sensitive Whole Words									
Location	Delay Type	Incr	Path	Trans	Logical R	Location	Delay Type	Incr	Path
1 Clock CLKIN (rising edge)		0.000	0.000	r		1 Clock CLKIN (rising edge)		0.000	0.000
2 H4		0.000	0.000	r	CLKIN (port)	2 H4		0.000	0.000
3	net (fanout=1)	0.042	0.042		CLKIN	3	net (fanout=1)	0.042	0.042
4 IOBS_0_136/PAD					CLKIN_ibuf/opit_0/I (	4 IOBS_0_136/PAD	CLKIN_ibuf/opit_0/I (		
5 IOBS_0_136/DIN	td	1.254	1.296	r	CLKIN_ibuf/opit_0/O (	5 IOBS_0_136/DIN	CLKIN_ibuf/opit_0/O (		
6	net (fanout=1)	0.000	1.296	r	CLKIN_ibuf/ntD	6	net (fanout=1)	0.000	0.955
7 IOL_7_137/DI					CLKIN_ibuf/opit_1/IN (	7 IOL_7_137/DI	CLKIN_ibuf/opit_1/IN (		
8 IOL_7_137/INCK	td	0.076	1.372	r	CLKIN_ibuf/opit_1/INCH	8 IOL_7_137/INCK	CLKIN_ibuf/opit_1/INCH		
9	net (fanout=1)	1.032	2.404		_NO	9	net (fanout=1)	0.878	1.881
10 USCM_56_112/CLK_A					clkbufg_0/gopclkbufg/	10 USCM_56_112/CLK_A	clkbufg_0/gopclkbufg/		

图 3-57 Output ports Clock-to-out

如果 Output Port 是从时序单元连接出去, Output ports Clock-to-out 将分 clock 报告从连接时序单元的 Clock Port 到 Output Port 的 Delay。

➤ Combination Delays

Combinational Delays					
From Port	To Port	Max Delay(ns)	Max Process Corner	Min Delay(ns)	Min Process Corner
1 A[0]	P_O[0]	5.295	SLOW	3.983	SLOW
2 A[0]	P_O[0]	4.052	FAST	3.146	FAST
3 A[1]	P_O[1]	5.350	SLOW	4.046	SLOW

Combinational delays from A[0] -> P_O[0]					
Max: 5.295(ns)			Min: 3.983(ns)		
Location	Delay Type	Incr	Path	Trans	Logical Reso
1 Data Path					
2 Clock CLKIN (rising edge)		0.000	0.000	r	
3 Input external delay		2.000	2.000	f	
4 F2		0.000	2.000	f	A[0] (port)
5 net (fanout=1)	0.085	2.085			A[0]
6 IOBS_0_112/PAD			A_ibuf[0]/opit_0/I (g		
7 IOBS_0_112/DIN	td	1.367	3.452	f	A_ibuf[0]/opit_0/O (g
8 net (fanout=1)	0.000	3.452	A_ibuf[0]/ntD		
9 IOL_7_113/DI			A_ibuf[0]/opit_1/IN (g		
10 IOL_7_113/RX_DATA_DD	td	0.127	3.579	f	IOL_7_113/DI
11 net (fanout=3)	0.466	4.045	nt_A[0]		
12 IOL_7_118/TX_DATA[7]			P_O_obuf[0]/opit_1/IN		
13 IOL_7_118/TX_DATA[0]			P_O_obuf[0]/opit_1/OUT		

图 3-58 Combination Delays

如果 Output Port 是从 Input Port 连接过来, 则 Combination Delays 将报告从 Input Port 到 Output Port 的 Delay。

➤ Setup/Hold times for input bus

Setup/Hold times for input bus Clocked by: CLKIN							
Source	Setup(ns)	Setup Process Corner	Hold(ns)	Hold Process Corner	Setup Slack(ns)	Hold Slack(ns)	Source Offset to Center(ns)
1 A[0]	0.526	SLOW	1.717	SLOW	2.474	-0.717	1.596
2 A[0]	0.794	FAST	0.701	FAST	2.206	0.299	0.954
3 A[1]	0.558	SLOW	1.759	SLOW			

CLKIN -> A[0]					
Setup(0.526) = arr_t(5.949) - launch_cik(0.000) - input_delay(2.000) - req_t(0.423) + cap_cik(5.000)				Hold(1.717) = req_t(4.409) - cap_cik(0.000) - arr_t(3.692) + launch_clk(0.000) + input_delay(1.000)	
Location	Delay Type	Incr	Path	Trans	Logical Reso
1 Data Path					
2 Clock CLKIN (rising edge)		0.000	0.000	r	
3 Input external delay		2.000	2.000	f	
4 F2		0.000	2.000	f	A[0] (port)
5 net (fanout=1)	0.085	2.085	A[0]		
6 IOBS_0_112/PAD			A_ibuf[0]/opit_0/I (g		
7 IOBS_0_112/DIN	td	1.367	3.452	f	A_ibuf[0]/opit_0/O (g
8 net (fanout=1)	0.000	3.452	A_ibuf[0]/ntD		
9 IOL_7_113/DI			A_ibuf[0]/opit_1/IN (g		

图 3-59 Setup/Hold times for input bus

Setup/Hold times for input bus 是专门针对 Input Buses 报告 Setup/Hold 要求, 以及计算 Buses 中最差的情形, 并报告当前设置过 Input Delay 等时序约束的端口的 Slack 以及如果时钟要位于数据的中心需要偏移的 Delay (即 Source Offset to Center)。

➤ Max/Min delays for output bus

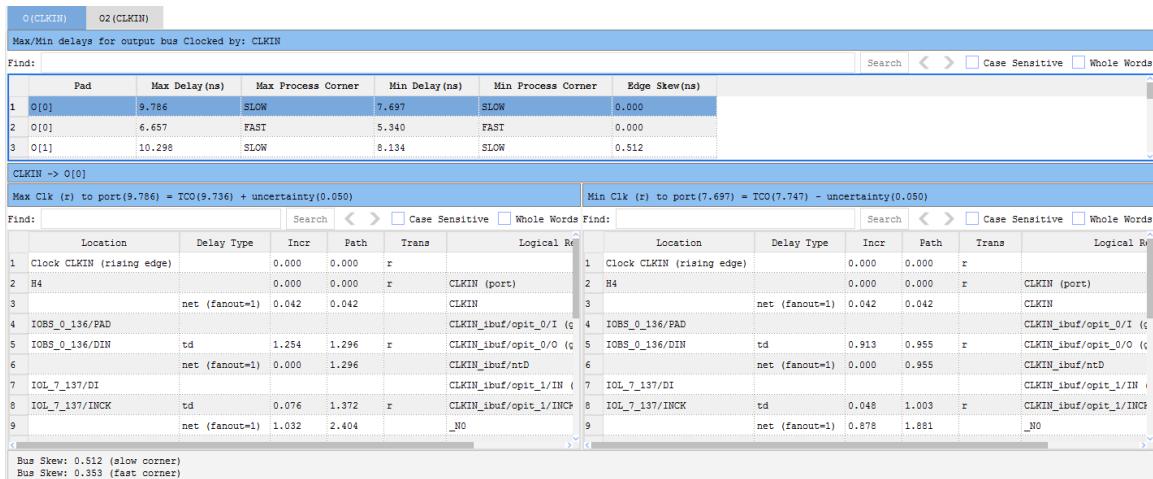


图 3-60 Max/Min delays for output bus

Max/Min delays for output bus 是专门针对 Output Buses 报告 Output Port 的 Delay 要求，并计算最差的 delay，以及选取端口 0 做参考端口计算其他端口和该端口的 Edge skew。

## 9. Max Skew Report

Max Skew Report 主要反映了 set\_max\_skew 约束的一组时序 path 是否满足约束的 skew 值。Max Skew 报告的上半部分窗口显示了相关 path 的一些 summary 信息，选中某条 path 后，会在报告的下半部分显示 path 的详细信息。如果 path 的 slack 值小于 0，即如果不满足，则会标红提醒。

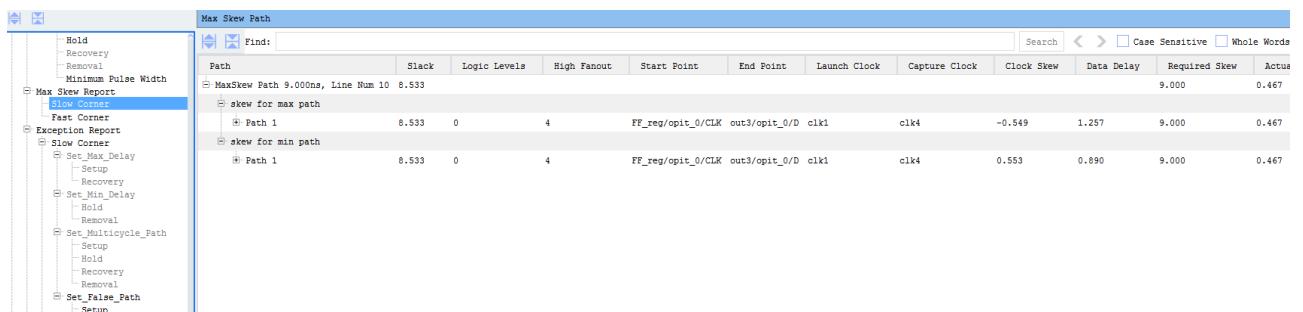


图 3-61 Max Skew Report

## 10. Exception Report

Exception Report 会对设置了 set\_multicycle\_path、set\_max\_delay、set\_min\_delay 和 set\_false\_path 这四种 Exception 约束的路径进行一个独立的报告，包括 Slow Corner 和 Fast Corner 两种极端情况下的 Timing Path 报告，这两个 Corner 的 Timing Path 报告都包括了 Set\_Max\_Delay、Set\_Min\_Delay、Set\_Multicycle\_Path 和 Set\_False\_Path，其中 Set\_Max\_Delay 以 Setup, Recovery 做了区分，Set\_Min\_Delay 以 Hold, Removal 做了区分，Set\_Multicycle\_Path 和 Set\_False\_Path 以 Setup, Hold, Recovery, Removal 做了区分。报告中分别列出了对应 path 的 summary 信息和详细信息。其中，Setup、Hold、Recovery、Removal 的报告类似，下面以

Setup 为例进行说明，具体见下图：

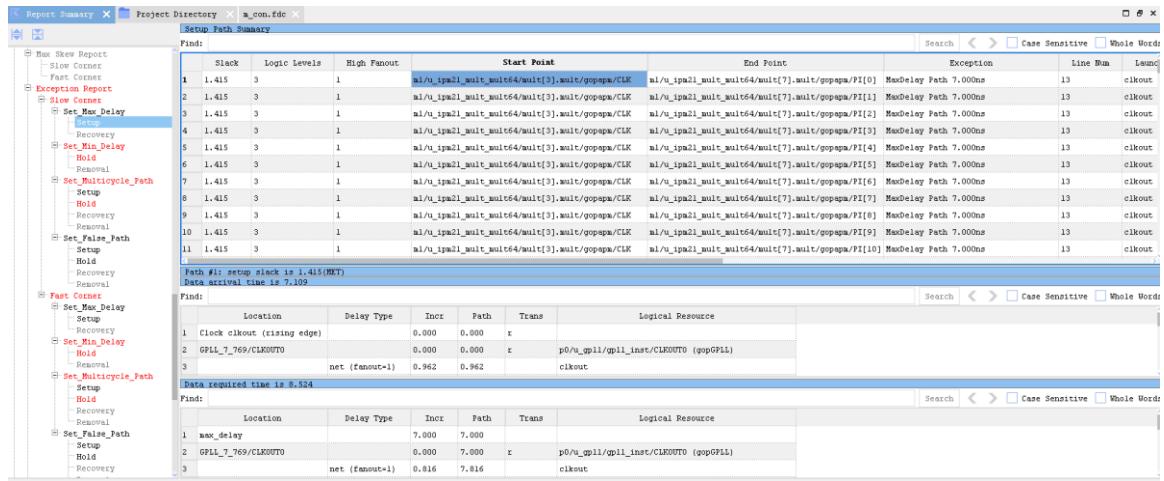


图 3-62 Exception Report

上图中 Setup Path Summary 中显示的信息与 Worst Case Timing Path 中的类似，均会显示 Slack、Logic Levels 等信息，不同之处在于会多显示一列“Line Num”信息，显示的信息为设置在路径上的 Exception 约束在约束文件中的行数。

## 3.6 Report power

### 3.6.1 Report power 设置

Report\_Power\_settings 介绍如下 Report Power 的选项设置如下图所示：

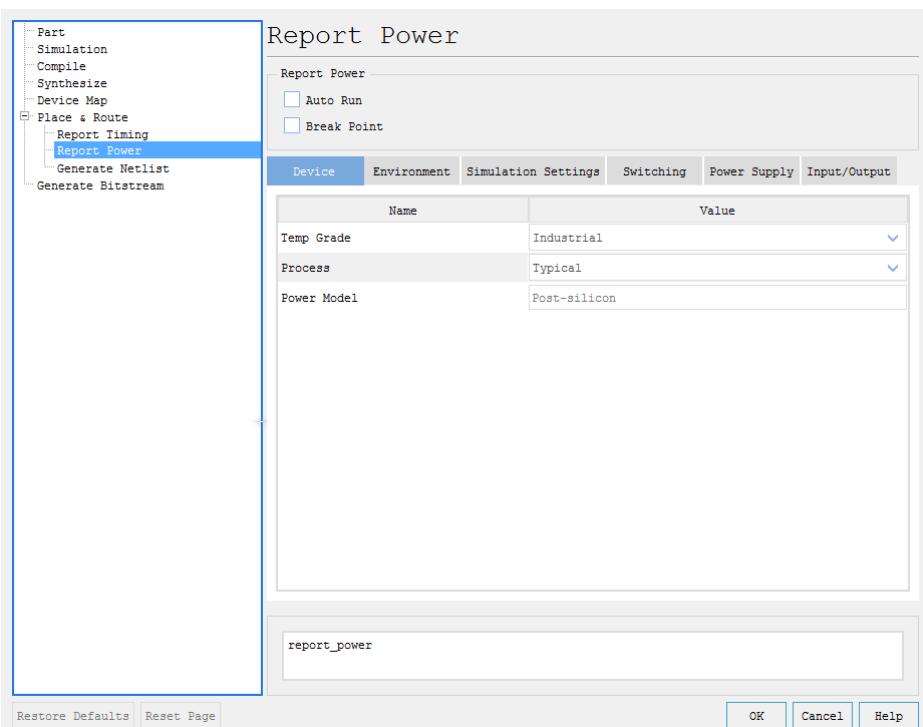


图 3-63 Report power 的选项设置

该窗口中有两个复选框和六个 Tab 页面：Device、Environment、Simulation Settings、Switching、Power Supply 和 Input/Output。

(1) Device 页面中，有两项需要用户选择的器件相关参数，更改的选项将会出现在下框中，分别如下：

Temp Grade: 温度等级，包括商业级 Commercial 和工业级 Industrial，主要影响 Environment 中结温和环境温度的设置范围。默认为 Industrial。

Process: 工艺偏差，包括标准 Typical 和最差 Worst，影响最终功耗评估结果。默认为 Typical。

Power Model: 功耗模型版本，显示当前器件的所处的功耗模型版本阶段，包括 Pre-silicon、Post-silicon 和 Production。只读，不可选。

(2) Environment 页面中，有 9 项需要用户勾选或者指定的环境参数配置，分别为：

Custom Junction Temperature: 是否用户指定节点温度，勾选后，除了 Junction Temperature 可以设置外，其余均不可修改。不勾选则为默认值 25.000。默认为不勾选。

Ambient Temperature: 环境温度，由 Device 中的 Temp Grade 的设置而定，具体为：Commercial (0~85)，Industrial (-40~100)。默认为 25.000。

Junction Temperature: 默认值为 25.000，同样由 Device 中的 Temp Grade 的设置而定，范围同环境温度。

Custom Effective Theta JA : 是否用户指定热系数，勾选后，用户可修改默认热系数。默认为不勾选。

Effective Theta JA: 芯片产生的热量扩散到外界的等效热阻抗，与器件的工艺、封装、散热等有关。Airflow(LFM)为 Still，Heat-Sink 为 None，Board Thermal Model 为 Normal 时，各器件的 Effective Theta JA 默认值如下：

compact 系列：

PGC1KG-MBG256 , 30.3 ; PGC1KG-FBG256 , 29.3 ; PGC1KG-LPG144 , 31.0 ;  
PGC1KG-LPG100, 31.73; PGC1KL-UWG36, 27.4; PGC2KG-MBG256, 30.3; PGC2KG-FBG256,  
29.3 ; PGC2KG-LPG144 , 31.0 ; PGC2KG-LPG100 , 31.73 ; PGC2KL-UWG49 , 27.4 ;  
PGC2KL-SSBG256 , 27.4 ; PGC4KD-MBG256 , 25.7 ; PGC4KD-LPG144 , 30.1 ;  
PGC4KD-MBG324 , 24.9 ; PGC4KD-MBG332 , 24.2 ; PGC4KD-FBG256 , 25.5 ;  
PGC4KL-SSBG256, 21.6; PGC4KL-UWG81, 21.6; PGC4KLS-SBG81, 21.6; PGC7KD-FBG484,

19.7 ; PGC7KD-MBG400 , 22 ; PGC7KD-MBG332 , 23.2 ; PGC7KD-MBG256 , 21.8 ;  
PGC7KD-LPG144 , 28.9 ; PGC10KD-MBG400 , 21.0 ; PGC10KD-MBG484 , 19.6 ;  
PGC10KD-MBG256, 20.8

PGL 系列:

PGL12G-FBG256 , 30.7 ; PGL12G-LPG144 , 29.7 ; PGL22G-FBG256 , 20.7 ;  
PGL22G-MBG324, 21.2; PGL22GS-LPG176, 21.8; PGL25G-FBG256, 19.8; PGL25G-MBG324,  
18.1 ; PGL25G-FBG484 , 17.8 ; PGL50G-FBG484 , 16.7 ; PGL50G-MBG324 , 16.7 ;  
PGL50G-MBG484, 16.7; PGL50H-FBG484, 16.7; PGL50H-MBG324, 16.7; PGL50H-MBG484,  
16.7; PGL100H-FBG900, 12.4

PG2L 系列:

PG2L25H-MBG325 , 21.2 ; PG2L25H-SBG238 , 21.2 ; PG2L100H-FBG676 , 12.5 ;  
PG2L100H-FBG484, 15.2; PG2L100H-MBG324, 17

PGT 系列:

PGT30G-FFBG484, 10.59; PGT180H-FFBG1152, 7.06; PGT180H-FFBG1140, 7.06;  
PGT180H-FFBG676, 7.58

PG2T 系列:

PG2T390H-FFBG676, 9.9; PG2T390H-FFBG900, 9.7

Airflow(LFM): 气流条件, 有 Still, Low, Medium 和 High 四个选项。默认为 Medium。

Heat-Sink: 散热情况, 有 None, Custom, Low, Medium 和 High 五个选项。默认为 Medium。

Effective Theta SA: 散热热阻系数, 当 Heat-Sink 选为 Custom 时可修改, 其余不可修改。  
默认为空。

Board Thermal Model : 板子散热情况, 有 JEDC, Best, Normal, Worst 四个选项。默认  
为 Normal。

(3) Simulation Settings 仿真波形设置页面:

Specify Input VCD File: 用户指定的后仿真文件, 该文件记录了仿真波形, 通过该文件能  
够获取信号的翻转情况。默认为空。

(4) Switching : 信号相关参数配置界面有 12 项需要用户设置的参数值, 分别为:

Output Load (pF):

IO 外部负载等电容, 取值为[0~50], 默认为 5.000;

Default Clock Freq (MHz): 默认时钟频率[0~500], 默认为 100.000;

Default Toggle Rate: 默认翻转率[0~1], 默认为 0.125;

Default Static Probability: 默认静态高电平概率[0~1], 默认为 0.125;

Default I/O Toggle Rate: I/O 的默认翻转率[0~1], 默认为 0.125;

Default I/O Enable Probability: I/O 的默认使能概率[0~1], 默认为 0.500; Default BRAM

Toggle Rate: DRM 的默认翻转率[0~1], 默认为 0.125;

Default BRAM Enable Probability: DRM 的默认使能概率[0~1], 默认为 1.000;

Default APM Toggle Rate: APM 的默认翻转率[0~1], 默认为 0.125;

Default APM Enable Probability: APM 的默认使能概率[0~1], 默认为 1.000; Default HSST

Toggle Rate: HSST 的默认翻转率[0~1], 默认为 0.125;

Default HSST Enable Probability: HSST 的默认使能概率[0~1], 默认为 0.500。

(5) Power Supply: 主要电源电压设定界面包括 7 项电压值, 分别为:

Core VCC (compact 系列电压范围为[2.25-3.64], PGL、PGT 系列电压范围为[0.99-1.21], PG2L、PG2T 系列电压范围为[0.95-1.05]), PGC 系列默认为 3.300, PGL、PGT 系列默认为 1.100, PG2L、PG2T 系列默认为 1.000;

Bank VCCIO3.3 电压[2.97-3.63], 默认为 3.300;

Bank VCCIO2.5 电压[2.25-2.75], 默认为 2.500;

Bank VCCIO1.8 电压[1.62-1.98], 默认为 1.800;

Bank VCCIO1.5 电压[1.35-1.65], 默认为 1.500;

Bank VCCIO1.2 电压[1.08 - 1.32], 默认为 1.200;

辅助电源 VCCAUX(PGC 系列不支持, PGL、PGT 系列电压范围为[2.97-3.63], PG2L、PG2T 系列电压范围为[1.71-1.89]), PGL、PGT 系列默认为 3.300, PG2L、PG2T 系列默认为 1.800。

(6) Input/Output: 输入输出设置界面主要选择导入配置文件及输出功耗文件路径。包含三项内容:

Output power report file: 功耗报告输出文件, 默认为空;

Output Fast settings file: 工程的功耗报告设置文件, 默认为空;

Output PPF file: 输出到 PPP 的功耗评估文件, 默认为空;

### 3.6.2 Report Power 报告

双击或右键->Run 运行 Report Power 会显示相应的报告。如下图所示：

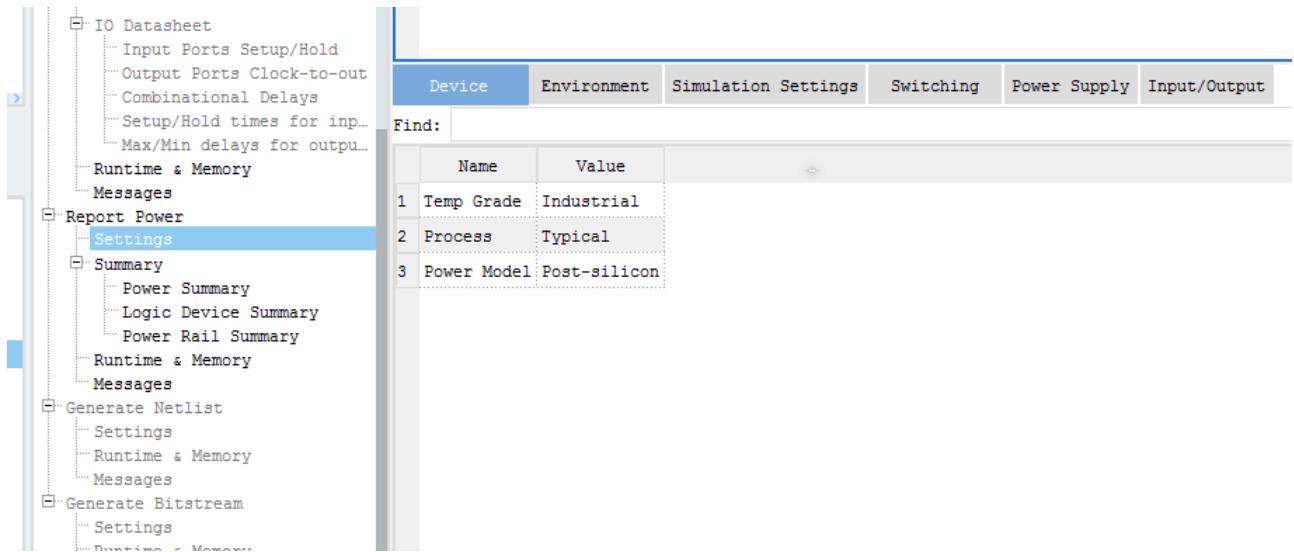


图 3-64 Report power 的报告

在 Reports Summary 中可以看到功耗报告的设置信息和功耗信息，设置信息包括：

#### 4) Settings

Settings 中的信息主要是 Design 的相关信息，如顶层 module 名，计算功耗的信号数据的 VCD 文件等。

#### 5) Device Settings

Device Settings 为器件的信息，主要包括器件类型、系列、封装以及温、速度等级和工艺偏差。

#### 6) Environment

Environment 设置为功耗报告的环境参数，包括温度、结温、热阻等。

#### 7) Power Rail

Power Rail 设定了在计算功耗中芯片中各个电源的电压值，如 vcc、vccio 等。

#### 8) Default Activity Rate

Default Activity Rate 设定了功耗计算过程中各个资源的信号的传输速率、翻转情况以及端口使能概率等的默认值。

功耗信息包括：

#### 1) Power Summary

Power Summary 中报告了该 design 在当前设置下的总体功耗和结温情况，主要包括总的

片上功耗、静态功耗和芯片结温等。

### 2) Logic Device Summary

Logic Device Summary 按照 design 中资源使用情况分类进行报告各项主要资源的动态功耗和占总功耗的百分比。

### 3) Power Rail Summary

Power Rail Summary 中按照芯片中各个电源的功耗报告功耗，主要包括电源名称、电压值（单位 V）、电流值（单位 A）、外部电流值（单位 A）和片上功耗（单位 W）。

## 3.7 Generate Netlist

Generate\_Netlist\_settings 运行 Generate Netlist 产生布局布线后仿真网表文件 sim.v 和时序反标文件 sdf。

后仿库目录为\$InstallDir\arch\vendor\pango\verilog\bsim，如：

C:\pango\PDS\_2021.3\arch\vendor\pango\verilog\bsim。

Generate Netlist 的选项设置如下图所示：

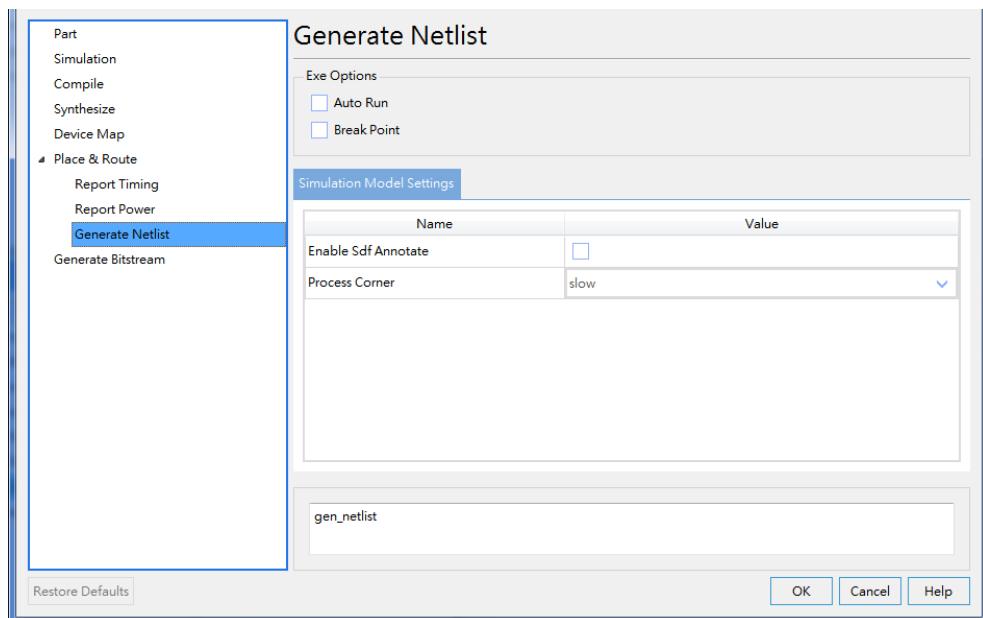


图 3-65 Generate Netlist 的选项设置

### Simulation Model settings

【Enable Sdf Annotate】选择是否在仿真网表（sim.v）中产生 sdf\_annotation 任务声明。

【Process Corner】指定 sdf delay 文件对应的 process corner（slow 或 fast）。

### 3.8 Design Editor

执行完 Place & Route 后可以通过 Tools 中的 Design Editor 打开 Design Editor。Design Editor 用法请查阅《Design\_Editor\_User\_Guide》。

### 3.9 Generate Bitstream

Generate\_Bitstream\_settings 生成二进制位流文件，是 Pango Design Suite 设计的最后一步。

Generate Bitstream 的选项设置 如下图所示：

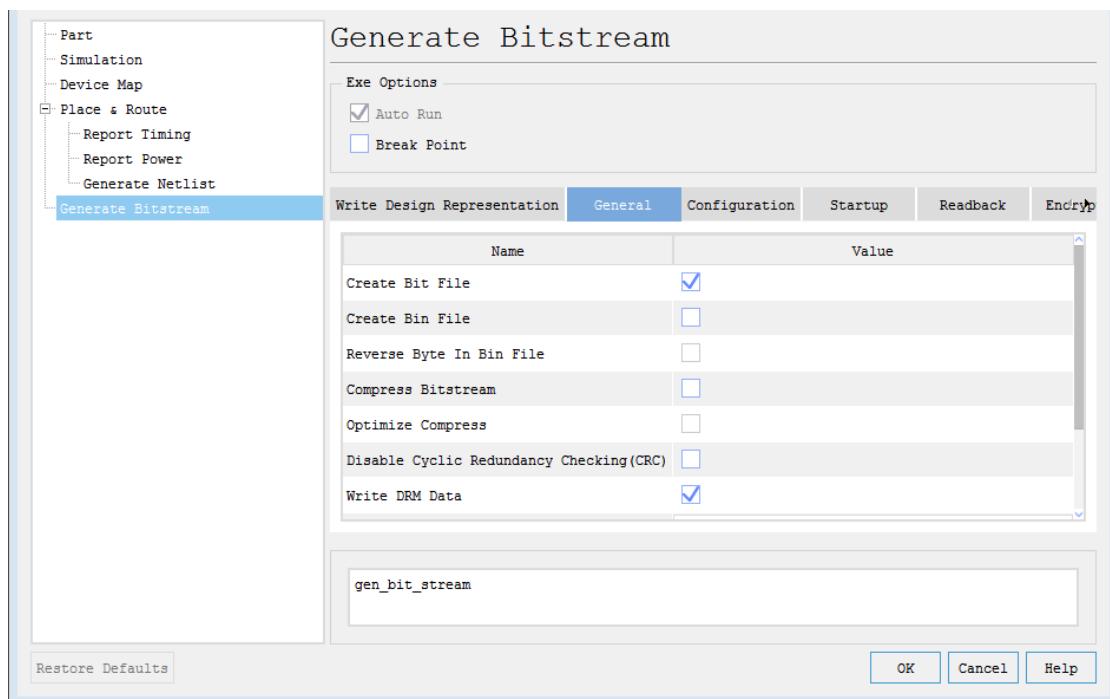


图 3-66 Generate Bitstream 的选项设置

Generate Bitstream 配置选项：位流配置选项内容与具体的芯片相关，上图所示为 PGL50H 的位流配置选项。

#### General

【Create Bit File】选择是否生成比特文件，选择 Yes 生成 sbit 文件，选择 No 则不生成，  
默认值为 Yes

【Create Bin File】选择是否生成无信息头的比特文件，Create bit file 配置为 Yes 的前提  
下，勾选则生成无信息头的 bin 文件，不选择则不生成，默认为不勾选

【Reverse Bit In a Byte】选择是否将 bin 文件中每个 Byte 的高位和低位反序，勾选后将  
bit7...bit0 反序为 bit0...bit7， 默认不勾选。

【Reverse Byte In a Word】选择是否将 bin 文件中每个 Word(一个 word = 4 个 byte)的高位

和低位反序，勾选后将 byte3byte2...byte0 反序为 byte0...byte2byte3， 默认不勾选

【Enable Write Feature Value】选择是否在 bin 文件中增加特征控制数据， 默认不勾选。

【Compress Bitstream】选择是否压缩位流文件，在 Create bit file 配置为 Yes 的前提下，选择 Yes 生成压缩的 sbit 文件，选择 No 生成未压缩的 sbit 文件， 默认值为 No

【Optimize Compress】选择是否优化压缩位流，优化压缩的含义为当帧数据为 0 时，不进行写入压缩位流文件中，当用户选择压缩位流时，优化压缩默认值为跟器件相关。

【Disable Cyclic Redundancy Checking (CRC)】选择是否忽略 CRC 验证。默认为进行 CRC 验证

【Write DRM Data】选择是否写入 DRM 数据， 默认值为 Yes。

【Unused IO Status】选择未使用 IO 的状态，不同系列的默认值不同，PGT180H、PGT200H 的默认值为 PULLUP，Compact、Logos 系列默认值为 PULLDOWN，Logos2、Titan2 系列默认值为 UNUSED。

【Show Time Info】选择是否显示位流头部信息中的时间和日期， 默认勾选，该选项不影响用户设计功能。

【Show Manufacturer Info】选择是否显示位流头部信息中的厂商， 默认勾选，该选项不影响用户设计功能。

## Configuration

【Usercode】给用户标识寄存器赋值，输入 8 位十六进制的 user code，该值会被写入到位流文件中， 默认值为 0xFFFFFFFF

【Set Configuration Bank Voltage(SCBV)】设置配置 bank 的电压， 默认为 None（该选项只在 PG2L100H、PG2T390H、PG2L25、PG2L50 和 PG2L200H 中支持），具体使用规则可以参考如下文档：

《UG040012\_Logos2 单板硬件设计指南 V1.3》中的表 1

《UG050012\_Titan2 单板硬件设计指南 V1.0》中的表 1

**注：当 Bank L4 和 Bank L5 都使用时，此 SCBV 设置无效。**

【Master Configuration Clock Frequency】主模式时钟 clk 频率选择， Titan 系列默认值为 12.5M， Logos 系列默认值为 3.125M， Compact 系列的默认值为 2.08M， Logos2 和 Titan2 系列默认值为 2.99M。

【System Clock Frequency】系统时钟 CLK 频率选择， PGL 系列默认值为 100M， Titan

系列默认值为 50M。（该选项只在 PGL 以及 Titan 系列中支持）

【Enable OSC Shut Off】控制是否允许用户逻辑关断 OSC， 默认值不允许

【Enable Watchdog In User Mode】选择是否在用户模式下使能看门狗， 默认为不使能。

【Enable Watchdog In Configuration Mode】选择是否在配置模式下使能看门狗， 默认为不使能。

【Load Watchdog】设置看门狗的超时时间， 默认为 3FFF\_FFFF

【Enable Version Select(VS) Pin】使能版本控制管脚， 默认不使能

【Enable Version Fallback】选择是否使能版本回退， 默认为使能

【Version Fallback Control】选择回退的位流版本， 可以选择“黄金位流”或者“Last Used bit stream”， 默认为黄金位流

【Enable RS Pin】选择是否使能 RS【1:0】pin 输出， 默认不使能。

【RS Pin Value】设置 RS Pin Value， 范围 0~3， 默认值为 0

【Set Fallback Retry Times】设置版本回退尝试次数， 默认为 1

【Set Bitstream Retry Times】设置位流尝试次数， 默认为 0

【RST\_N pin】选择 RST 管脚的上下拉还是悬空， 默认为上拉

【TDI Pin】选择 TDI 管脚的上下拉还是悬空， 默认为上拉

【TMS Pin】选择 TMS 管脚的上下拉还是悬空， 默认为上拉

【TCK Pin】选择 TCK 管脚的上下拉还是悬空， 默认为上拉

【TDO Pin】选择 TDO 管脚的上下拉还是悬空， 默认为上拉

【INIT\_FLAG\_N pin】选择 INIT\_FLAG\_N 管脚的上拉还是悬空， 默认为上拉

【CFG\_DONE Pin】选择 CFG\_DONE 管脚的上下拉还是悬空， 默认为上拉

【CFG\_CLK Pin】选择 CFG\_CLK 管脚的上拉还是悬空， 默认为上拉

【MODE\_2 Pin】选择 MODE\_2 管脚的上下拉还是悬空， 默认为上拉

【MODE\_1 Pin】选择 MODE\_1 管脚的上下拉还是悬空， 默认为上拉

【MODE\_0 Pin】选择 MODE\_0 管脚的上下拉还是悬空， 默认为上拉

【Enable External Master Configuration Clock】选择是否使能外部主配置时钟， 默认为不使能

【BPI Read Cycle】选择 BPI 读取周期， 默认为 1

【BPI Page Size】选择 BPI 的页面大小， 默认为 1 Byte/Half Word

【BPI Mode】选择 BPI 的模式， 默认为 asynchronous

【Enable Done Synchronization】选择是否 Done 信号同步， 默认为不同步

【SRAM Retention】开启 SRAM retention 测试， 默认不勾选

【Over Temperature Shut Off】过温关断使能， 默认不勾选

## Startup

【Wake Up Time Slot Length】唤醒周期长度选择，即 T1, T2, T3 之间的时间间隔，默认值为 1，指一个唤醒时钟周期

【Done Time Slot】done 信号拉高时刻选择， 默认值为 T1

【Gwen Time Slot】gwen 信号拉高时刻选择， 默认值为 T3

【Grsn Time Slot】grsn 信号拉高时刻选择， 默认值为 T4

【Gouten Time Slot】gouten 信号拉高时刻选择， 默认值为 T2

【Enable Wait DCI Match】唤醒等待 DCI 匹配， 默认值为不等待

【Enable Wait PLL Lock】唤醒时等待 PLL， 默认值为不等待

【Wake Up Clock】选择唤醒时钟， 默认为主配置时钟

【Enable USER Clock Wake Up】是否使用用户时钟进行唤醒， 默认不勾选

## Readback

【Create Mask File】选择是否生成 mask 文件，用来决定回读文件与原始文件的比较规则。选择 Yes 生成 mask 文件，选择 No 则不生成， 默认值为 Yes

【Persist Slave Parallel Pins】控制用并行模式配置完成后，外部并行端口是否保留用于回读， 默认不保留。（Titan 系列器件选项）

【Persist Ipal】控制配置完成后，内部并行端口是否保留用于回读， 默认保留(只针对 PGT30G 设置有效)

【Disable Readback CRC】选择是否在回读时使能 CRC 验证， 默认使能回读 CRC 验证

【Bitstream Security】控制重配及回读配置存储器功能， 默认值为允许重配， 允许回读

【Select Persist Pin】下拉框可选择 Persist Slave Parallel Pins 或 Persist Slave SPI Pins 或 None， 默认为 None。

【Master Configuration Clock Frequency In Read Back】回读时钟选择， 默认值为 2.08M

【Enable CRC Clock In Read Back】使能回读 CRC 时钟， 默认值为不勾选

**【Enable Mask Variable Memory In Read Back】** 选择回读时是否 mask RAM 区域， 默认为 mask RAM 区域。

**【Persist Pin】** 选择保留复用端口用于配置接口或释放给用户使用， 默认为释放端口。

### Encryption

**【Encrypt Bitstream】** 勾选否加密位流文件，在 Create bit file 配置为 Yes 的前提下，勾选则生成加密的 sbit 文件及 nky 文件（包含加密所需的 128 比特的初始 CBC 值和 256 比特的密钥），不勾选则生成未加密的 sbit 文件，默认值为不勾选。用户可以编辑 CBC 和密钥字符串或选择 nky 文件，若用户不指定，则随机生成 nky 文件

**【Select Key Type】** 配置是否使用内部 KEY， 默认值为 Key。选择 Temporary Key 时，每次下载位流前需先下载对应的 nky 文件，选择 Key，则先将密钥烧录到芯片的 efuse 模块（该操作仅执行一次），然后直接下载位流即可

**【Starting CBC Value】** 用户手动输入初始的 CBC 字符串，字符为 16 进制，输入长度为最多 32 个字符，默认为空

**【Starting GCM Value】** 用户手动输入初始的 GCM 字符串，字符为 16 进制，输入长度为最多 24 个字符，默认为空

**【Input Encryption Key File】** 选择 nky 文件，如果在此选择了密钥文件，则软件选择以密钥文件进行加密，而不管用户在选项中是否输入了密钥字符串，默认为空。

**【Key String】** 用户手动输入密钥字符串，字符为 16 进制字符，输入长度为最多 64 个字符，默认为空

**【Number of Encryption Block】** 分块加密数据的块数，字符为 10 进制字符，输入范围为 0~max\_number (max\_number 在不同的器件中取值不同， PG2L100H: 58; PG2L200H: 136; PG2L50H: 32; PG2L25H: 18; PG2T390H: 201)，默认值为 0

**【Enable Obfuscate】** 使能混淆加密，与加密功能配合使用， 默认不使能

**【Enable Self Authentication】** 使能自认证功能，对加密数据进行自认证， 默认不使能

**【Enable DPA】** 使能 DPA 保护功能，与加密功能配合使用， 默认不使能

**【DPA Protect Region】** 设置 DPA 保护区域， 默认为 ALL

**【Data Width】** 加密数据宽度设置，配置加密数据时使用的数据位宽， 默认值为 X1

### Authentication

**【Authenticate Bitstream】** 选择是否生成认证位流文件， 默认为不生成

**【Input RSAPrivate Key File】**选择 RSA-2048 认证的私钥文件（pem 文件），若用户选择认证数据流而不指定私钥文件，则随机生成私钥文件，默认为空

### Logos2 加密注意事项：

- a) 不支持压缩数据流的 DPA 保护（压缩勾选时， Enable DPA 不能进行勾选）
- b) 压缩数据流和 DPA 保护数据流不支持分块加密（压缩数据流或 DPA 保护数据流时，

Number of Encryption Blocks 必须为 0）

- c) Enable Self Authentication 与 Enable DPA 不能同时勾选
- d) Enable DPA 勾选， Authenticate Bitstream, Encrypt Bitstream 必须勾选
- e) Enable Self Authentication 勾选时，不支持分块加密。
- f) Number of Encryption Blocks 不为 0 时， Encrypt Bitstream 必须勾选

### SEU File

**【Soft Error Insertion】**用于 SEU 验证的软插错文件生成配置，可选择插错位置为 Random, Unused 区域。

### Feature Control

**【Feature Value】** 显示所选特征控制位后对应的特征控制值（十六进制）。

目前仅支持 PGC 系列，如图 3-67 所示为特征控制位的选项界面。

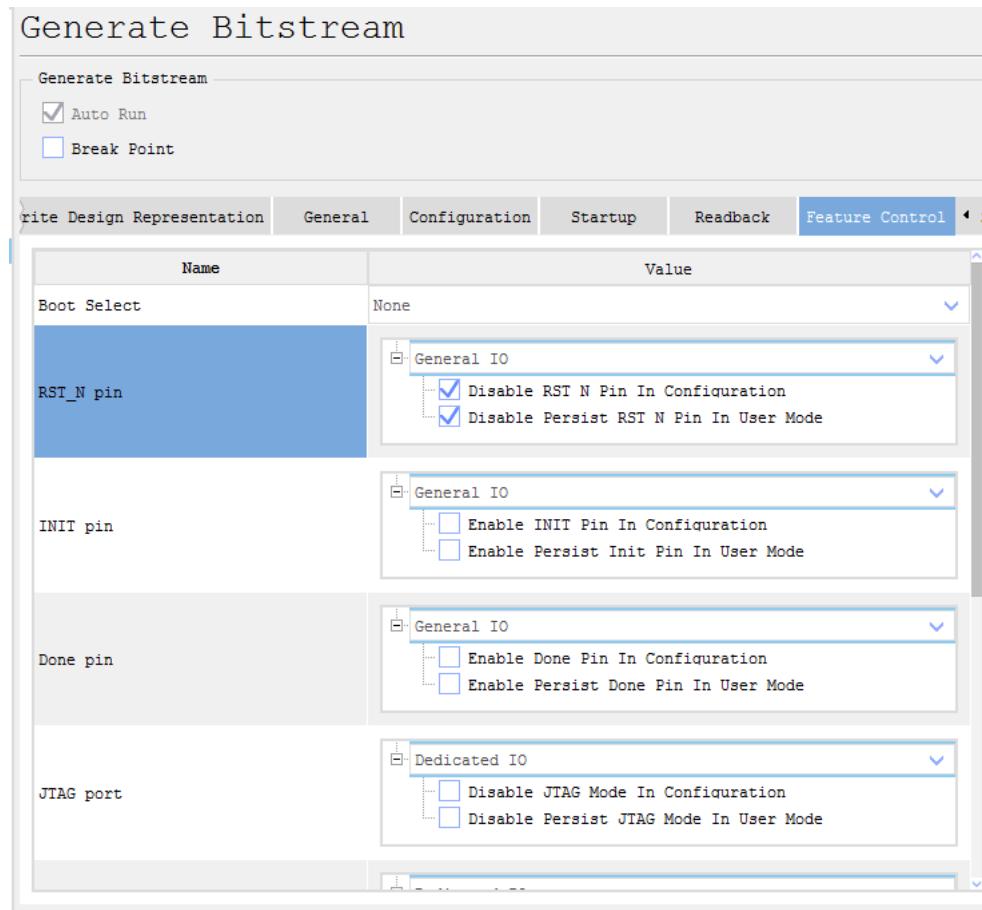


图 3-67 特征控制位的选项界面

以下表来展示 compact 系列特征控制位的功能的支持情况：

Name	Status	Value	描述
Boot Select	\	Master Auto Mode(from embed flash)\ (默认值) Master SPI Mode(from outer SPI flash)\None	选择启动模式，选择主自加载的时候才可设置主自加载双启动；选择不为 None 的时候才可设置普通双启动
RST_N pin	General IO (默认值)	勾选 Disable RST_N Pin In Configuration 和 Disable persist RST_N Pin in User Mode	在用户模式和配置模式下都禁用 RST_N 管脚
	Dedicated IO	不勾选 Disable RST_N Pin In Configuration 和 Disable persist RST_N Pin in User Mode	在用户模式和配置模式下都使能 RST_N 管脚
	Other Usage	任意选项组合都可设为 Other Usage	根据实际选项禁用或使能 INIT 管脚
INIT pin	General IO (默认值)	不勾选 Enable INIT Pin In Configuration 和 Enable persist INIT Pin in User Mode	在用户模式和配置模式下都禁用 INIT 管脚
	Dedicated IO	勾选 Enable INIT Pin In Configuration 和 Enable persist INIT Pin in User Mode	在用户模式和配置模式下都使能 INIT 管脚
	Other Usage	任意选项组合都可设为 Other Usage	根据实际选项禁用或使能 INIT 管脚

Done pin	General IO (默认值)	不勾选 Enable Done Pin In Configuration 和 Enable persist Done Pin in User Mode	在用户模式和配置模式下都禁用 Done 管脚
	Dedicated IO	勾选 Enable Done Pin In Configuration 和 Enable persist Done Pin in User Mode	在用户模式和配置模式下都使能 Done 管脚
	Other Usage	任意选项组合都可设为 Other Usage	根据实际选项禁用或使能 Done 管脚
Jtag port	General IO	勾选 Disable Jtag Mode In Configuration 和 Disable persist Jtag Mode in User Mode	在用户模式和配置模式下都禁用 Jtag 接口
	Dedicated IO (默认值)	不勾选 Disable Jtag Mode In Configuration 和 Disable persist Jtag Mode in User Mode	在用户模式和配置模式下都使能 Jtag 接口
	Other Usage	任意选项组合都可设为 Other Usage	根据实际选项禁用或使能 Jtag 接口
Slave SPI port	General IO	勾选 Disable Slave SPI Mode In Configuration 和 Disable persist Slave SPI Mode in User Mode	在用户模式和配置模式下都禁用从 SPI 接口
	Dedicated IO (默认值)	不勾选 Disable Slave SPI Mode In Configuration 和 Disable persist Slave SPI Mode in User Mode	在用户模式和配置模式下都使能从 SPI 接口
	Other Usage	任意选项组合都可设为 Other Usage	根据实际选项禁用或使能从 SPI 接口
Slave I2C port	General IO	勾选 Disable Slave I2C Mode In Configuration 和 Disable persist Slave I2C Mode in User Mode	在用户模式和配置模式下都禁用从 I2C 接口
	Dedicated IO (默认值)	不勾选 Disable Slave I2C Mode In Configuration 和 Disable persist Slave I2C Mode in User Mode	在用户模式和配置模式下都使能从 I2C 接口
	Other Usage	任意选项组合都可设为 Other Usage	根据实际选项禁用或使能 I2C 接口
I2C Address	\	默认值为 0	I2C 地址
Dual Boot Mode	\	Disable (默认值) / Dual boot	是否选择双启动模式
Advance	\	Disable Slave Parallel Mode In Configuration(默认勾选)	在配置模式下是否禁用从并接口
		Enable Fast Master SPI Mode (默认不勾选)	是否使能主 SPI 接口
		Disable Persist Slave Parallel Mode In User Mode (默认勾选)	在用户模式下是否禁用从并接口
		Enable Master SPI Mode In User Mode (默认不勾选)	在用户模式下是否使能主 SPI 接口
		Enable Slowly Read Mode (默认不勾选)	是否使能慢读模式
Feature	\	默认为 2090	特征控制值

表 3-1 PGC 系列支持情况

不同器件支持的功能不一样，以下表来展示除 Feature Control 之外的功能支持情况：

类型	配置项	PGT180H	PGL系列	PGC系列	Logos2、 Titan2系列
General	Create Bit File	√	√	√	√
	Create Bin File	√	√	√	√
	Reverse Bit In a Byte	√	√	√	√
	Reverse Byte In a Word	√	√	√	√
	Enable Write Feature Value	×	×	√	×
	Compress Bitstream	√	√	√	√
	Optimize Compress	×	√	√	√
	Disable CRC	×	√	√	√
	Write DRM Data	×	√	√	√
	Unused IO Status	√	√	√	√
Configuration	Usercode	√	√	√	√
	Set Configuration Bank Voltage(SCBV)	×	×	×	√
	Master Configuration Clock Frequency	√	√	√	√
	System Clock Frequency	√	√	×	×
	Enable OSC Shut Off	√	√	√	√
	Enable Watchdog In Configuration Mode	×	√	√	√
	Load Watchdog	×	√	√	√
	Enable Done Synchronization	×	√	√	√
	Enable Version Select(VS) Pin	×	√	×	√
	Enable RS Pin	×	×	×	√
	RS Pin Value	×	×	×	√
	Enable Version Fallback	×	×	×	√
	Version Fallback Control	×	√	×	√
	Set Fallback Retry Times	×	√	×	×
	Set Bitstream Retry Times	×	√	×	×
	RST_N pin	×	√	×	√
	TDI Pin	×	√	×	√
	TMS Pin	×	√	×	√
	TCK Pin	×	√	×	√
	TDO Pin	×	√	×	√
	INIT_FLAG_N pin	×	×	×	√

CFG_DONE pin	×	×	×	√
CFG_CLK pin	×	×	×	√
MODE_2 pin	×	×	×	√
MODE_1 pin	×	×	×	√
MODE_0 pin	×	×	×	√
Enable External Master Configuration Clock	×	√	×	√
BPI Read Cycle	×	√	×	√
BPI Page Size	×	√	×	√
BPI Mode	×	√	×	√
Enable Wake Up Fabric Only	×	×	×	×
Control cmi_cram_fsm_reset	×	×	×	×
Control apb_fsm_reset Of APB1	×	×	×	×
Control cmi_drm_fsm_reset	×	×	×	×
Control apb_fsm_reset Of APB2	×	×	×	×
Enable Deskew CLK	×	×	×	×
If Screen deskew_over signal	×	×	×	×
Time Difference Between Fabric Glogen and HPIO Glogen	×	×	×	×
Generate hpio_rst_n	×	×	×	×
Pull up HPIO Glogen And Fabric Glogen	×	×	×	×
Full Load Line Of FIFO	×	×	×	×
Set Frequency of clk_refresh	×	×	×	×
Set Timeout Limit Of deskew_timer	×	×	×	×
Set Timeout Limit Of hpio_timer1	×	×	×	×
Set Timeout Limit Of hpio_timer0	×	×	×	×
CCS clk_refresh Switch	×	×	×	×
DIO BKCL VCCIO	×	×	×	×
DIO_1 Port Status	×	×	×	×

	DIO_2 Port Status	×	×	×	×
	DIO_3 Port Status	×	×	×	×
	DIO_4 Port Status	×	×	×	×
	DIO_5 Port Status	×	×	×	×
	DIO_6 Port Status	×	×	×	×
	DIO_7 Port Status	×	×	×	×
	DIO_8 Port Status	×	×	×	×
	DIO_9 Port Status	×	×	×	×
	DIO_10 Port Status	×	×	×	×
	DIO_11 Port Status	×	×	×	×
	DIO_12 Port Status	×	×	×	×
	Enable Internal SPI_1	×	×	×	×
	Internal SPI Data Width	×	×	×	×
	Enable Internal SPI_0	×	×	×	×
	Enable User Master Clock	×	×	×	×
	Enable User JTAG	×	×	×	×
	Enable User Control gwen	×	×	×	×
	Enable User Control grs_n	×	×	×	×
	Enable User Control gouten	×	×	×	×
	SRAM Retention	×	×	√	×
Startup	Wake Up Time Slot Length	√	√	×	√
	Done Time Slot	√	√	×	√
	Gwen Time Slot	√	√	×	√
	Grsn Time Slot	√	√	×	√
	Gouten Time Slot	√	√	×	√
	Enable Wait PLL Lock	√	√	√	√
	Enable Wait DCI Match	×	√	×	×
	Wake Up Clock	×	√	√	×
	Enable Wait Deskew	×	×	×	×
	Enable Wait HPIO Initialization	×	×	×	×
ReadBack	Enable USER Clock Wake Up	×	×	×	√
	Create Mask File	√	√	√	√
	Persist Slave Parallel Pins	√	√	×	×
	Bitstream Security	√	√	√	√
	Disable Readback CRC	×	√	√	√
	Persist Slave SPI Pin	×	√	×	×
	Enable CRC Clock In Read Back	×	×	√	×

	Master Configuration				
	Clock Frequency In Read Back	×	×	√	×
	Persist Ipal	×	×	×	×
Encryption	Enable Mask Variable Memory In Read Back	×	√	×	√
	Encrypt Bitstream	√	√	×	√
	Select Key Type				√
	Starting CBC Value				×
	Input Encryption Key File				√
	Key String				√
	Strarting GCM Value	×	×	×	√
	Encrypt Mode	×	×	×	×
	Starting INV Value	×	×	×	×
	Number of Encryption Blocks	×	×	×	√
Authentication	Enable Obfuscate	×	×	×	√
	Enable Self Authentication	×	×	×	√
	Enable DPA	×	×	×	√
	Data Width	×	×	×	√
	Authenticate Bitstream	×	×	×	√
	Authenticate Mode	×	×	×	×
	Input Private Key File	×	×	×	×
	Input RSAPrivate Key File	×	×	×	√

表 3-2 其他功能支持情况

说明：上述表格中 PGC 系列包括 PGC1KG, PGC1KL, PGC2KG, PGC2KL, PGC4KD, PGC4KL, PGC7KD, PGC10KD。

运行 Generate Bitstream 生成位流文件，可以看到界面如下：

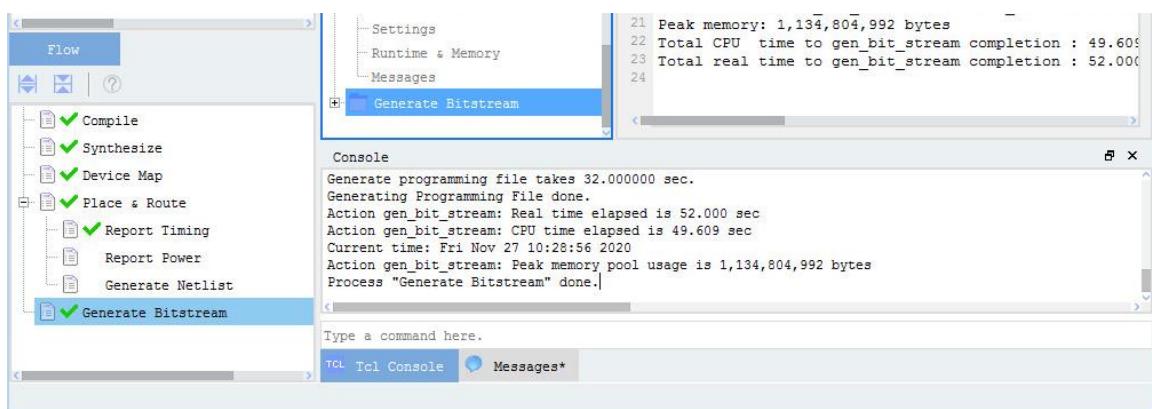


图 3-68 运行 Generate Bitstream 后软件界面

生成.sbit 位流文件完毕，查看工程文件夹可以看到生成的位流文件如下图所示：

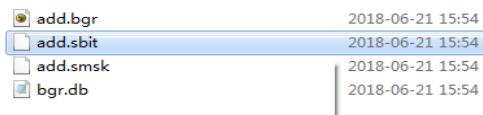


图 3-69 生成的位流文件

### 3.10 命令行操作

上述各个操作都支持直接用命令行进行操作，可以逐条执行命令，也可以将命令写到 tcl 脚本里，进行批处理操作，用法：

```
source <批处理 tcl 文件>;
```

在执行了一次后，可以进行 clean -all 命令，将之前的结果全部删掉，再次执行 source 命令；

如果命令中存在-option 并且运行成功，会改变工程的配置。

关于命令行操作的详细介绍，可以参阅第五章关于命令行运行的说明。

## 4 相关插件

在开发流程 `synthesize`、`device map`、`place & route`、`report timing`、`report power` 或 `generate bitstream` 在运行时，禁止打开各插件（此时，各插件的工具栏按钮不可用）；但已经打开的插件仍然可以正常使用。各插件可以同时打开，但相同的插件只能打开一次（ip 配置界面除外）。

### 4.1 ADS

综合工具 ADS 的具体用法可以查阅《ADS\_Synthesis\_User\_Guide》和《ADS\_Language\_Support\_Reference\_Manual》。

### 4.2 Synplify\_Pro

综合工具 Synplify\_Pro 的具体用法可以查阅安装路径下 `syn/doc` 文件夹中《fpga\_attribute\_reference》、《fpga\_command\_reference》、《fpga\_hdl\_reference》、《fpga\_reference》、《fpga\_user\_guide》。

### 4.3 User Constraint Editor

UCE 具体用法请查阅《User\_Constraint\_Editor\_User\_Guide》。

### 4.4 IP Compiler

IP Compiler 具体用法请查阅《IP\_Compiler\_User\_Guide》。

### 4.5 Physical Constraint Editor

PCE 具体用法请查阅《Physical\_Constraint\_Editor\_User\_Guide》。

### 4.6 Design Editor

Design Editor 具体用法请查《Design\_Editor\_User\_Guide》。

### 4.7 Fabric Inserter

Fabric Inserter 具体用法请查《Fabric\_Inserter\_User\_Guide》。

## 4.8 Fabric Debugger

Fabric debugger 具体用法请查阅《Fabric\_Debugger\_User\_Guide》。

## 4.9 Fabric Configuration

Fabric Configuration 具体用法请查阅《Fabric\_Configuration\_User\_Guide》。

## 4.10 Fabric JtagServer

Fabric JtagServer 具体用法请查阅《Fabric\_JtagServer\_User\_Guide》。

## 4.11 Pango Power Calculator

Pango Power Calculator 具体用法请查阅《Pango\_Power\_Calculator\_User\_Guide》。

## 4.12 Pango Power Planner

Pango Power Planner 具体用法请查阅《Pango\_Power\_Planner\_User\_Guide》。

## 4.13 Timing Analyzer

Timing Analyzer 具体用法请查阅《Timing\_Analyzer\_User\_Guide》。

## 4.14 Route Constraint Editor

Route Constraint Editor 具体用法请查阅《Route\_Constraint\_Editor\_User\_Guide》。

## 4.15 Pango SSN Estimator

Pango SSN Estimator 具体用法请参阅《Pango\_SSN\_Estimator\_User\_Guide》。

## 4.16 Pango SSN Analyzer

Pango SSN Analyzer 具体用法请参阅《Pango\_SSN\_Analyzer\_User\_Guide》。

## 4.17 Simulation

Simulation 具体用法请查阅《Simulation\_User\_Guide》。

## 4.18 View RTL Schematic

如果综合工具是 ADS，View RTL Schematic 的具体用法可以查阅

《ADS\_Synthesis\_User\_Guide》。

如果综合工具是 Synplify Pro, View RTL Schematic 的具体用法可以查阅安装路径下 syn/doc 文件夹中的《fpga\_reference》。

#### 4.19 View Technology Schematic

如果综合工具是 ADS, View Technology Schematic 的具体用法可以查阅《ADS\_Synthesis\_User\_Guide》。

如果综合工具是 Synplify Pro, View Technology Schematic 的具体用法可以查阅安装路径下 syn/doc 文件夹中的《fpga\_reference》。

## 5 命令行运行

PDS 软件支持三种命令行运行方式，我们以 Win7 上的 pds.exe 为例，下面分别介绍：

### 5.1 工程运行方式

pds 的-project 参数可以以直接打开一个工程的方式来运行 PDS 软件，操作方式是在 DOS 界面键入：

```
C:\Users\pango>C:\pango\PDS_2021.3\bin\pds.exe -project
```

```
C:\Users\pango\Desktop\project\project.pds
```

按 Enter 键后，PDS 软件主界面会打开，并且工程 project.pds 已经被打开，如下图所示：



图 5-1 pds.Exe 打开 pds

### 5.2 TCL 脚本运行方式

主流程中每一步都有对应的命令，这些命令可以直接写在一个 tcl 脚本中，在 pds 界面如下的 Console 窗口，可以 source 一个 tcl 文件执行。

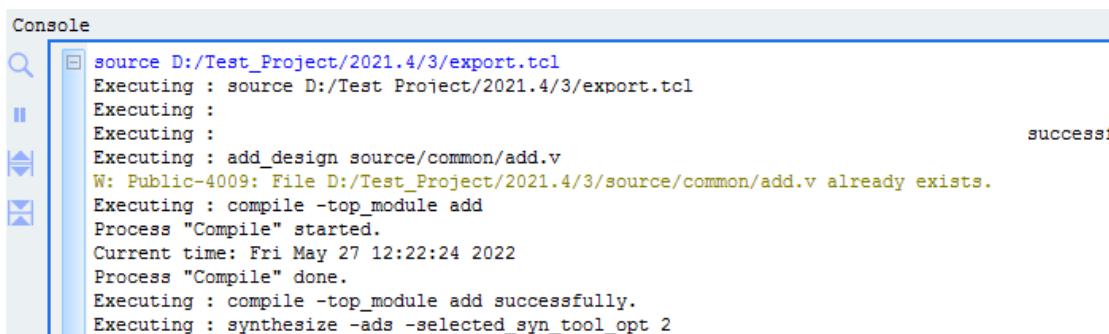


图 5-2 脚本运行 tcl 文件

示例脚本如下：

```
add_design "E:/add.v"
```

```
add_design "E:/assign001.v"
```

```
set_arch -family Logos -device PGL50H -speedgrade -6 -package FBG484
```

```
synthesize -ads -selected_syn_tool_opt 2 -top_module add
```

dev\_map  
pn  
report\_timing  
gen\_bit\_stream

### 5.3 Shell 运行方式

一、 pds 的-file 参数可以以直接执行 tcl 脚本文件的方式来运行 PDS 软件，操作方式是在 DOS 界面键入：

```
C:\Users\pango> C:\pango\PDS_2021.3\bin\pds.exe -file F:\work\project\pds\Adder\Adder.tcl  
-work_dir C:\work_dir -project_name test
```

其中， work\_dir 和 project\_name 是可选项。 Work\_dir 指定工作目录， project\_name 指定工程文件名。规则如下：

1. 指定 file 时，如果不指定 work\_dir 和 project\_name，则工程存放在 CMD 窗口目录，且 pds 工程文件名为 tcl 文件同名。
2. 指定 file 时，如果指定 work\_dir，但不指定 project\_name，则工程存放在 work\_dir 选项值目录，且 pds 工程文件名为 tcl 文件同名。
3. 指定 file 时，如果不指定 work\_dir，但指定 project\_name，则工程存放在 tcl 文件同级目录，且 pds 工程文件名为 project\_name 选项值。
4. 指定 file 时，如果指定 work\_dir 和 project\_name，则工程存放在 work\_dir 选项值目录，且 pds 工程文件名为 project\_name 选项值。

二、 pds 的-project 参数可以以打开.pds 工程文件的方式来运行 PDS 软件，操作方式是在 DOS 界面键入：

```
C:\Users\pango> C:\pango\PDS_2021.3\bin\pds.exe -project  
F:\work\project\pds\Adder\project.pds -run gen_bit_stream
```

其中， run 是可选项，用来指定运行到的流程步骤。规则如下：

-run compile	运行到 Compile
-run synthesize	运行到 Synthesize
-run dev_map	运行到 Device Map
-run pnr	运行到 Place & Route

- run report\_timing 运行到 Report Timing
- run report\_power 运行到 Report Power
- run gen\_netlist 运行到 Generate Netlist
- run gen\_bit\_stream 运行到 Generate Bit Stream

## 5.4 PDS 支持的命令

语法: command [OPTION]… [OPTION VALUE]…

例:

```
add_design "F:/ipcore/rom/inst.idf"  
add_constraint -logic -sdc pgdemo.sdc  
compile -include_path {C:/Program Files C:/Program Files (x86)}  
synthesize -ads  
gen_bit_stream -create_bin -startup_sel TCK
```

注: 在 tcl 标准语法中, “\”是转义字符, 您可能需要将 windows 文件夹路径中的 “\” 更改为 “/” 后再运行

**【generate\_constraint】**:用于将 pnr 之后的 DB 结果以约束的形式导出, 可以指定导出的格式以及导出的约束资源类型, 如:

```
generate_constraint -file {file_name}
```

以 pcf 的形式生成所有的约束信息, 包括 IO, group, prim 资源。

```
generate_constraint -file {file_name} -device {APM/PLL/DRM}
```

以 pcf 的形式生成指定类型资源的约束信息

```
generate_constraint -file {file_name} -logic
```

以 fdc 的形式生成 IO、device 的约束信息

```
generate_constraint -file {file_name} -slack -1.0
```

读取 pnr 之后的 DB 跑时序分析, 以 txt 的形式生成 slack 值大于等于 -0.1ns 的 instance 的位置约束信息

```
generate_constraint -file {file_name} -less_mode -slack -1.0
```

读取 pnr 之后的 DB 跑时序分析，以 txt 的形式生成 slack 值小于 -0.1ns 的 instance 的位置约束信息

```
generate_constraint -file {file_name} -group
```

以 fdc 的形式生成 group instance 的约束信息

```
generate_constraint -file {file_name} -prim
```

以 fdc 的形式生成 prim instance 的约束信息

(注：生成的约束文件中均带有 no\_check 选项，即生成的 IO 命令不会进行 IO 标准检查（也就是说此时默认命令中的设置都是正确的，命令中的配置均会正常作用于 pnr），生成的 instance 相关命令不会对 instance 是否存在进行检查，不存在的 instance 忽略）

如生成的 fdc 中：

```
define_attribute {clk} {PAP_IO_CHECK_IO} {FALSE}
```

即除了 clk 的 PAP\_IO\_LOC 之外，其他 IO 标准不会检查，直接传递下去

生成的 pcf 中：

```
def_port {clk} LOC=A1 IOSTANDARD=LVCMS33 VCCIO=3.3 CEHCK_IO=FALSE
```

即关于 clk 的除了 LOC 之外，其他相关标准不会检查，直接传递到 pnr

```
def_inst_site {inst} device_name -no_check
```

即命令中不会检查 inst 的名合法性，如果该 inst 存在，该命令就正常起作用，否则就会将该命令忽略

**【export\_fdc】**:用于将 dev\_map 之后的 pcf 结果反标转换成 fdc 形式约束导出，可以指定导出的路径及文件命令

```
export_fdc -file file_name.fdc
```

此命令可支持-no\_log，缺省默认打印转换失败提示信息，若要关闭此打印信息 export\_fdc

-file file\_name.fdc -no\_log true

## 6 时序约束命令

### 6.1 对象获取命令

对象获取命令主要是用来辅助时序约束和时序报告命令，约束命令和报告命令中的 object 必须通过对象获取命令获取。获取的对象来自于 design 网表，因此用户约束也是以网表为约束源，在 design 转化为 RTL 后，软件会显示 RTL 级的图形显示，用户可以根据图形显示的名字直接写 sdc 命令。

#### 6.1.1 all\_clocks

此命令返回所有用户定义的 clock(使用 create\_clock 或者 create\_generated\_clock 产生的)。

格式如下：

```
all_clocks
```

#### 6.1.2 all\_inputs

返回所有满足条件的 input 或者 inout port，如果未设置任何参数，则表示返回 design 中所有的 input 或者 inout port。

```
all_inputs [-level_sensitive | -edge_triggered] [-clock clock_name]
```

-level\_sensitive：表示返回使用 set\_input\_delay 约束过的，并且为电平触发的 input，见 set\_input\_delay。

-edge\_triggered：表示返回使用 set\_input\_delay 约束过的，并且为边沿触发的 input，见 set\_input\_delay。

注意：格式中用到了或符号，表示上面两个参数只能选其中一个，或者都不选，都不选的情况下表示不考虑输入信号的触发方式。

-clock clock\_name：这个返回所有使用 set\_input\_delay 约束过的，并且关联的时钟名字为 clock\_name 的 input，可不选。

#### 6.1.3 all\_outputs

返回所有满足条件的 output 或者 inout port，如果未设置任何参数，则表示返回 design 中所有的 output 或者 inout port。

```
all_outputs [-level_sensitive | -edge_triggered] [-clock clock_name]
```

-level\_sensitive：表示返回使用 set\_output\_delay 约束过的，并且为电平触发的 output，

见 set\_output\_delay。

-edge\_triggered : 表示返回使用 set\_output\_delay 约束过，并且为边沿触发的 input，见 set\_output\_delay。

注意：格式中用到了或符号，表示上面两个参数只能选其中一个，或者都不选，都不选的情况下表示不考虑输出信号的触发方式。

-clock clock\_name : 这个返回所有使用 set\_output\_delay 约束过的，并且关联的时钟名字为 clock\_name 的 output，可不选。

#### 6.1.4 all\_registers

all\_registers [-no\_hierarchy] [-clock clock\_name | -rise\_clock clock\_name | -fall\_clock clock\_name] [-cells | -data\_pins | -clock\_pins | -async\_pins | -output\_pins ] [-level\_sensitive | -edge\_triggered]

-clock clock\_name : 表示 ck 输入端的时钟为 clock\_name 的 register。

-rise\_clock clock\_name : 表示 ck 输入端的时钟为 clock\_name 并且触发边沿为上升沿的 register。

-fall\_clock clock\_name : 表示 ck 输入端的时钟为 clock\_name 并且触发边沿为下降沿的 register。

注意：格式定义中用了两个或符号，如下所示：

[-clock clock\_name | -rise\_clock clock\_name | -fall\_clock clock\_name]

表示只能使用-clock clock\_name、-rise\_clock clock\_name、-fall\_clock clock\_name 中的任意一个。

-cells : 返回带 register 功能的 instance。

-data\_pins : 返回一个 data pin 的集合，集合的大小取决于 design 本身及命令中的其他参数。

-clock\_pins : 返回一个 data pin 的集合，集合的大小取决于 design 本身及命令中的其他参数。

-async\_pins : 返回异步的置位或者复位 pins。

注意：格式中[-cells | -data\_pins | -clock\_pins | -async\_pins | -output\_pins ]中间用或符号隔开，表示这四个选项只能选择其中的一个。

-edge\_triggered : 表示边沿触发的寄存器。

-level\_sensitive : 表示电平触发的锁存器。

注意：格式中[-level\_sensitive | -edge\_triggered]中间有一个或符号，表示这两个选项只能选择其中一个。

-no\_hierarchy: 仅考虑顶层的 register。

#### 6.1.5 get\_cells

cell 在的 design 中对应的是网表中的 instance，如 and2 等，design 有多种类型，如 RTL 级网表，综合后网表以及布局布线后的网表等，用户只需要在 RTL 级网表上做时序约束，其他网表的时序约束由软件内部完成。

get\_cells [-hierarchical] [-hsc separator] [-regexp] [-nocase] <patterns>

-hierarchical : 匹配完整层次名，例如 A\*C 可以匹配到完整层次名为 A/B/C 的对象。

-hsc separator : hierarchical 分隔符。如果该 cell 是 hierarchical，那么在书写名字的时候应该加上 module 的名字，并且用分隔符隔开，例如，假设分隔符为 ‘\_’ module A 中有 cell 名字为 unst，A 被例化为 a，那么 cell 的名字应该写为 a\_unst。

-regexp : 表示支持正则表达式而不是简单通配符。

-nocase : 表示不关心 pattern 的大小写。

patterns : instance 的名字。

#### 6.1.6 get\_clocks

返回当前用户定义过的（通过 create\_clock 和 create\_generated\_clock）所有满足条件的 clocks。

get\_clocks [-regexp] [-nocase] [-include\_generated\_clocks] <patterns>

-regexp : 表示支持正则表达式而不是简单通配符。

-nocase : 表示不关心 pattern 的大小写。

-include\_generated\_clocks : 返回的时钟里面包括匹配到的 patterns 时钟所驱动的 generated clocks。

patterns : clock 的名字。

#### 6.1.7 get\_nets

返回 design 中所有满足条件的 net。

get\_nets [-hierarchical] [-hsc separator] [-regexp] <patterns>

-hierarchical : 匹配完整层次名，例如 A\*C 可以匹配到完整层次名为 A/B/C 的对象。

-hsc separator : hierarchical 分隔符。如果该 net 是 hierarchical，那么在书写名字的时候应该加上 module 的名字，并且用分隔符隔开，例如，假设分隔符为‘\_’module A 中有 net 名字为 reset，那么 net 的名字应该写为 A\_reset。

-regexp : 表示支持正则表达式而不是简单通配符。

-nocase : 表示不关心 pattern 的大小写。

patterns : net 的名字。

#### 6.1.8 get\_pins

返回 design 中所有满足条件的 pin。

get\_pins [-hierarchical] [-hsc separator] [-regexp] [-nocase] <patterns>

-hierarchical : 匹配完整层次名，例如 A\*C 可以匹配到完整层次名为 A/B/C 的对象。

-hsc separator : hierarchical 分隔符。如果该 pin 是 hierarchical，那么在书写名字的时候应该加上 module 的名字，并且用分隔符隔开，例如，假设分隔符为‘\_’module A 中有 pin 名字为 reset，那么 pin 的名字应该写为 A\_reset

-regexp : 表示支持正则表达式而不是简单通配符。

-nocase : 表示不关心 pattern 的大小写。

patterns : pin 的名字。

#### 6.1.9 get\_ports

返回设计中所有满足条件的 port。

get\_ports [-regexp] [-nocase] <patterns>

-regexp : 表示支持正则表达式而不是简单通配符。

-nocase : 表示不关心 pattern 的大小写。

patterns : port 的名字。

## 6.2 时钟约束命令

### 6.2.1 create\_clock

命令格式如下：

create\_clock

-period <period\_value>

- [ -name <clock\_name> ]
- [ -waveform <edge\_list> ]
- [ -add ]
- [ <source\_objects> ]
- [ -disable ]

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参数列表：

-period : 定义创建时钟的周期，为浮点数，单位是 ns。

[ -name ] : 创建的时钟的名字，如果未定义，则一定要指定 source\_objects，同时名字和 source\_objects 中第一个参数的名字相同；如果用户未定义 source objects，则用户必须定义 -name，此时创建的 clck 是一个虚拟时钟。

[ -waveform ] : 定义时钟上升沿和下降沿的位置，其中上升沿在前，下降沿在后，成对存在。格式如下：

-waveform {time\_rise time\_fall time\_rise time\_fall ...}

[ -add ] : 表示可以在同一个引脚上定义多个 clock，如果之前该 source\_object 上面已经定义了一个时钟，并且用户未加-add 参数，即用户并未说明该 pin 或者 port 上面需要定义多个时钟，那么后面的定义的 clock 就会将前面的覆盖掉，同时软件将会给出警告。

[ <source\_objects> ] : 创建的时钟的源，可以是 pin 或 port（注意，必须通过 get\_pins、get\_ports 得到）。

[ -disable ] : 使本条约束不生效。

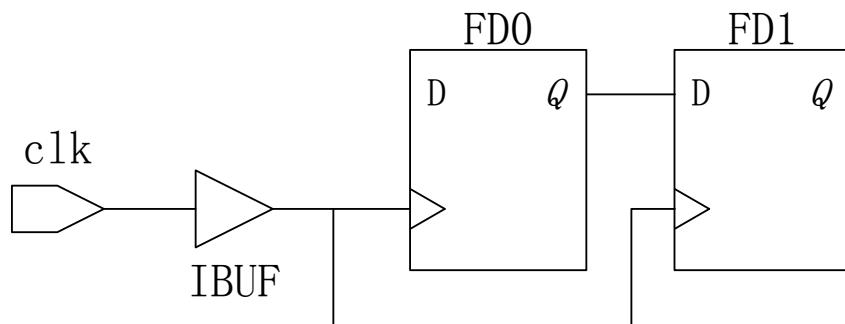


图 6-1 时钟创建实例

clk 信号控制 FD0 和 FD1 触发器的发射和捕获时间，为了确保从 FD0/C 端到 FD0/D 端的

时序路径建立和保持时间检查准确，需要在输入 port clk 处创建一个时钟，例如：

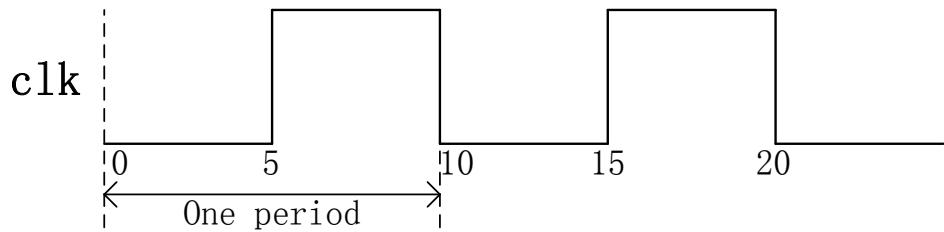


图 6-2 时钟创建波形图

约束命令如下：create\_clock -name clk -period 10 -waveform {5 10} [get\_ports clk]

### 6.2.2 create\_generated\_clock

```
create_generated_clock
[-name <clock_name>]
-source <master_pin>
[-master_clock <clock>]
[-edges <edge_list>] [-edge_shift <shift_list>]
[-multiply_by <factor>] [-divide_by <factor>]
[-duty_cycle <percent>]
[-invert]
[-add]
<source_objects>
[-disable]
```

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参数列表：

**[-name]** : 创建的时钟的名字。

**-source** : 指定 source clock 的 pin 或者 port，驱动该 generated clock。

**[-master\_clock]** : 指定 source clock，在 master pin 上面有多个 source clock，而 generated clock 只来自于其中一个 clock 时使用；当 master pin 上面只有一个 source clock 的时候，可以不指定该参数。

[`-edges`]：表示从 source clock 引用的边沿的列表，每一个 edge 不能小于 1 且不能小于上个 edge，和 `create_clock` 中 waveform 的格式一致；不同的是最后还要加上第二个周期开始的位置，PDS 要求 `edge_list` 中参数个数一定为 3 个，里面不仅包含了第一个周期的上升沿和下降沿，还包含第二个周期的上升沿，软件能从中计算出 generated clock 的周期；如下图所示：

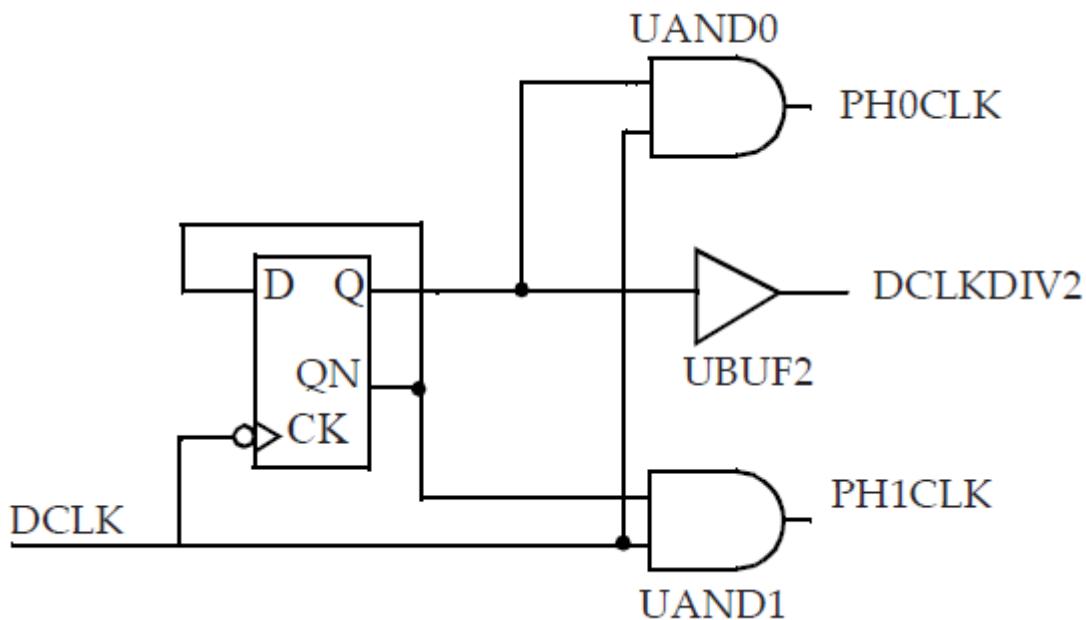


图 6-3 generated clock 周期计算

对于 DCLKDIV2，其波形如下：

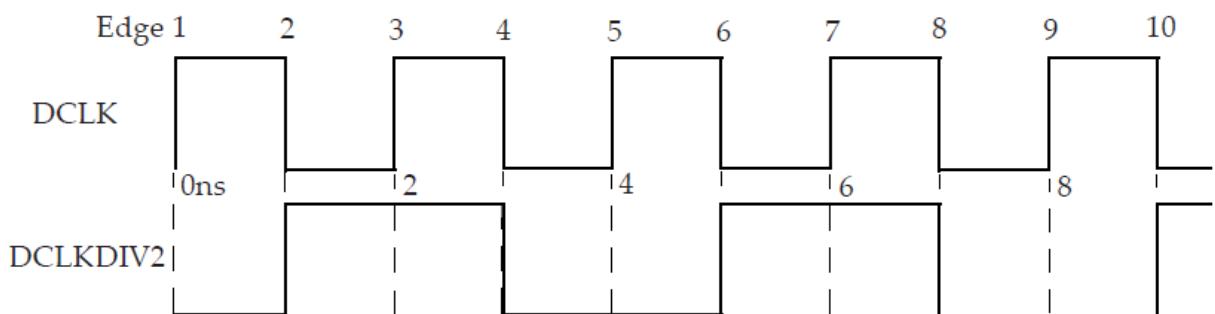


图 6-4 DCLKDIV2 波形

对应的约束如下：

```
create_generated_clock -name DCLKDIV2 -edges {2 4 6} -source DCLK [get_pins
UBUF2/Z]
```

其中 2 表示 source clock 的第二个 edge，“4”表示 source 的第四个 edge……依次类推；在 generated clock 里，“2”对应第一个上升沿位置，“4”表示第一个下降沿的位置，“6”表示下一

个上升沿的位置。

**[-edge\_shift]** : 指定了一系列浮点数, 来表示每条边的偏移量, 单位是 ns, 必须和-edges一起使用; 如果 generated clock 需要实现相移功能, 那么就需要用到“-edge\_shift”, “-edge\_shift”里面的列表元素个数必须和“-edges”里面的个数一致。

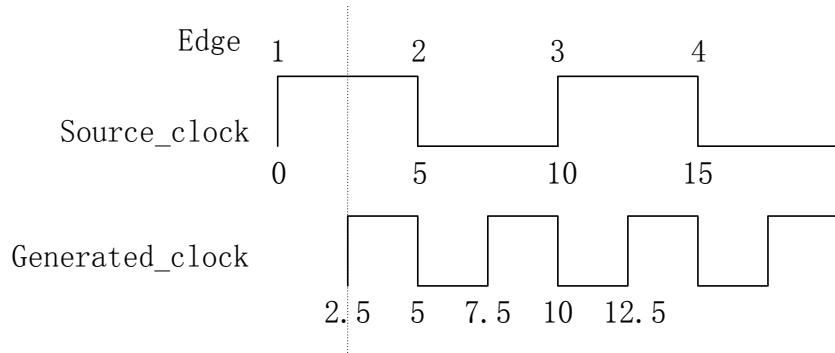


图 6-5 相移波形示例

上图中的例子中, generated clock 针对 source clock 实现了倍频加 180 度相移的功能, 可写成如下格式:

```
create_generated_clock -name generated_clock -source master_pin
-edges {1 2 3} -edge_shift {2.5 0 -2.5} [get_pins PLL/OUT0]
```

此命令中, 由-edges 参数可知, generated clock 的第一个上升沿、第一个下和第二个上升沿分别对应 source clock 的 edge1、edge2 和 edge3, , 由-edge\_shift 参数可知, generated clock 的第一个上升相对于 source clock 的 edge1 沿向右偏移了 2.5 个时间单位, 第一个下降沿相比 source clock 的 edge2 没有偏移, 第二个上升沿相对于 source clock 的 edge3 向左偏移了 2.5 个时间单位。

**[-multiply\_by]** : 倍频系数, 倍频系数是一个大于0的整数, 将时钟周期缩小指定倍数, 倍频不止引起周期的变化, 波形也将等比例缩小。

**[-divide\_by]** : 分频系数, 分频系数是一个大于0的整数, 将时钟周期扩大指定倍数。分频不止引起周期的变化, 还会导致波形的变化。在PDS里, 当分频系数为2以及2的指数倍时会将占空比设置为50%, 其他系数值则保持原占空比不变。

-divide\_by 和-multiply\_by 可以同时使用, 可以-divide\_by 在前, 也可以-multiply\_by 在前; 分频倍频同时使用可以做到非整数倍的频率变换, 即小数分频, 在 UCE 界面就可以通过设置 multiply\_devide\_by 来实现小数分频, 如下:

```
create_generated_clock -name MIIDIV2 -source MICK -mutiply_by 2 -devide_by 3
```

## [get\_pins UMIIDIV/Q]

**[-duty\_cycle]** : 单位是 percent (%) , 描述占空比, 即高电平占整个周期的比例, 占空比只能在使用-multiply\_by 的时候使用;

**[-invert]** : 表示相对于 source clock, generated clock 的 clock 信号被取反。如下图所示:

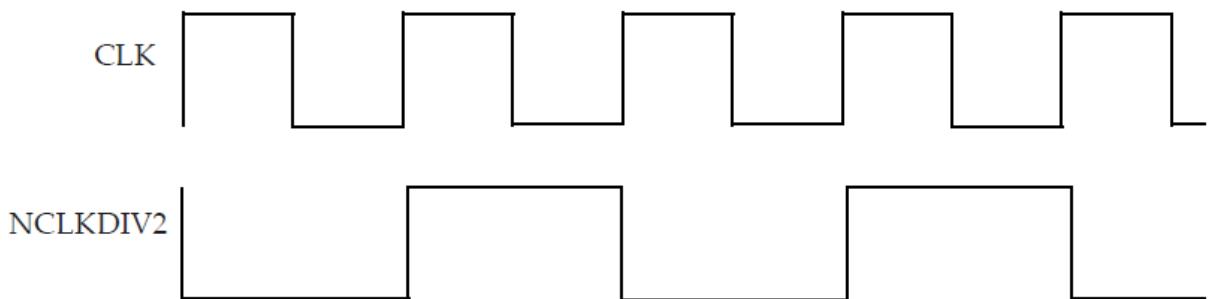


图 6-6 -invert 实例

对应约束命令如下:

```
create_generated_clock -name NCLKDIV2 -divide_by 2 -invert -source CLK [get_pins  
UINVQ/Z]
```

**[-add]** : 如果之前已经为这个 pin 定义了一个相同类型的 clock, 那么加-add 选项将会在之前的基础上加入一个新的 clock, 如果用户没有加这个选项, 那么新创建的 clock 将会将原来的 clock 替换。如果在同一个 source object 上面定义了多个类型完全相同的 clock, 这多个 clock 之间是物理独立关系。

**<source\_objects>** : 被创建 generated clock 的 pin, port 或者是 net。

**[-disable]** : 使本条约束不生效。

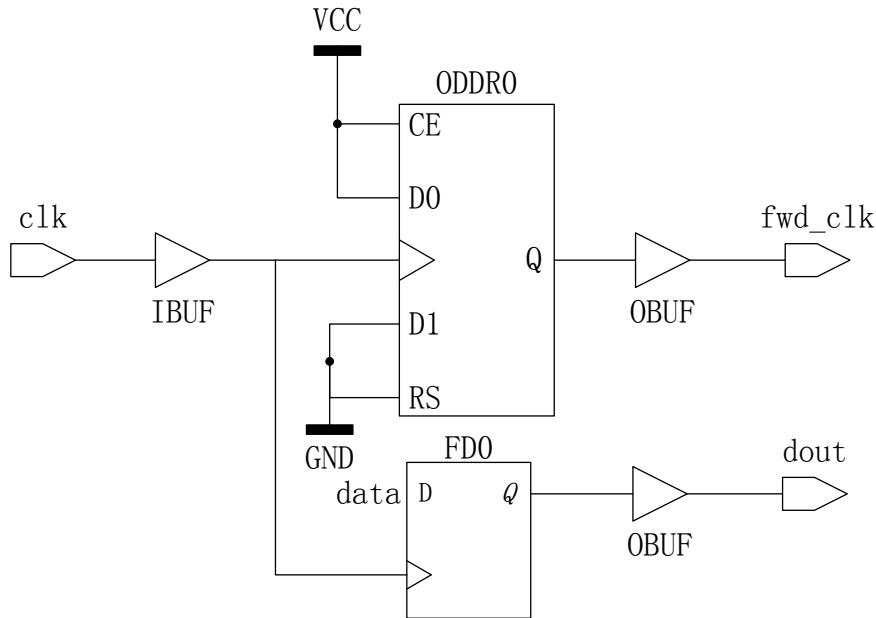


图 6-7 随路时钟示例

随路时钟是定义在输出 port 处，复制了 FPGA 内部时钟信号的生成时钟，随路时钟通常作为源时钟同步接口下其他输出 port 的参考时钟。设置输出 port 的 output delay 时可以指定到随路时钟上以检查是否满足源同步接口时序要求。创建如图所示的随路时钟，约束如下：

```
create_generated_clock -name fwd_clk -divide_by 1 -source [get_pins ODDR0/C] [get_ports
fwd_clk]
```

注：

1. `-edges/-edge_shift` 不能和 `-multiply_by`、`-divide_by`、`-duty_cycle` 同时使用。
2. 对于用户在 PLL 上创建的 `create_generated_clock` 约束，如果 PLL 配置了相移，那么约束命令就需要通过 `-edges/-edge_shift` 选项将 PLL 的相移信息体现在 generated clock 的波形上；软件自动在 PLL 上 infer 的时钟也通过上述方法体现 PLL 的相移延时。另外，用户需要注意时序分析路径上的时钟边沿对，有时可能需要通过 `set_multicycle_path` 约束进行调整。
3. `create_generated_clock` 命令支持 `rename` 功能，即用户只需要指定 `create_generated_clock` 中的 `-name`, `source_objects`, 软件会自动 infer 出该时钟的周期和波形，同样也可以指定 `-source` | `-master_clock` 来筛选 source 时钟，该功能只支持 PLL 和 CLKDIV 的时钟输出端口。如果用户想要使用此功能，则不能指定如下选项：`[-edges edge_list]` `[-edge_shift shift_list]` `[-divide_by factor]` `[-multiply_by factor]` `[-duty_cycle percent]` `[-invert]`。用户可使用该功能重命名软件 infer

的 generated clock，并基于指定的名字进行后续的时序约束。命令示例如下：

```
create_generated_clock -name pll_clk [get_pins {pll.clkout1}]
```

### 6.2.3 set\_clock\_latency

set\_clock\_latency 只支持 source latency，不支持 network latency，约束中必须定义 -source 参数。Source latency 主要用于定义时钟生成点到时钟定义点之间的延时信息。即如下图所示：

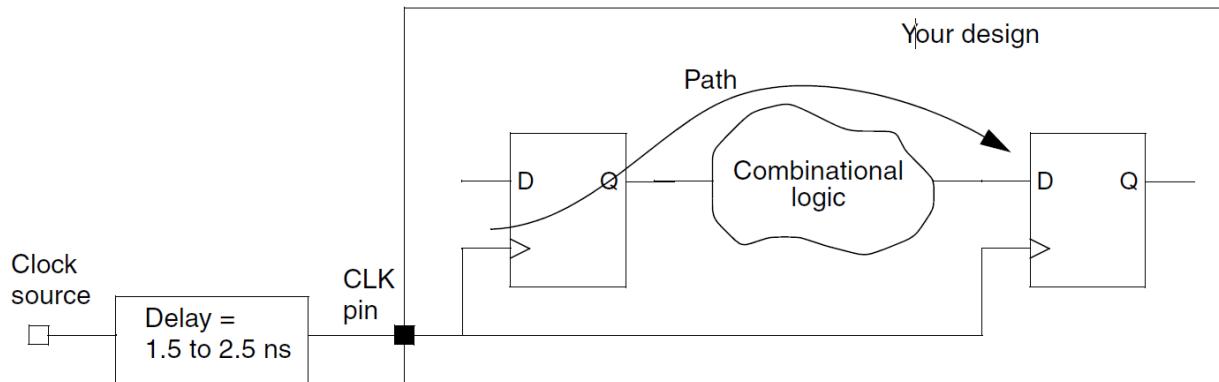


图 6-8 Source latency 示意图

其中，CLK pin 是的 FPGA 输入时钟的定义点，Clock source 是该时钟的生成点，source latency 即是定义这两点之间的延时信息，具体命令及参数介绍如下：

`set_clock_latency`

`[-clock <clock_list>]`

`[-rise] [-fall]`

`[-min] [-max]`

`[-early] [-late]`

`--source`

`<object_list>`

`<latency_value>`

`[-disable]`

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参数列表：

`[-clock]`：当 object\_list 的 pin/port 上定义了多个时钟，该选项用于筛选用户指定的时钟，使用时需要遵循先声明后使用的原则，即：clock 需要先通过 `create_generated_clock` 或 `create_clock` 创建。

`[-rise]`：表示该 latency 只应用在触发时钟边沿为上升沿的情况。

[**-fall**]：表示该 latency 只应用在触发时钟边沿为下降沿的情况。

[**-min**]：如果用在 bc\_wc 的环境下，-min 表示 fast corner，-max 表示 slow corner；不指定-min 和-max，默认 fast corner 和 slow corner 命令都会生效，但-min 和-max 不能同时指定。**-early** 表示该 latency 是 clock 的左偏移量，**-late** 表示该 latency 是 clock 的右偏移量；不指定-early 和-late，默认 clock 的左右偏移量都会生效，但-early 和-late 不能同时指定。

[**-max**]：见 min 的解释。

[**-early**]：见 min 的解释。

[**-late**]：见 min 的解释。

**-source**：表示该 latency 是 source latency，约束中必须指定。

**object\_list**：clock definition point，即时钟定义点。可以通过 get\_pins、get\_ports 的方式获取对应 pinport 上定义的时钟，也可以直接将 latency 定义在 clock 上面。

<latency\_value>：延时值，为浮点数，单位是 ns。

[**-disable**]：使本条约束不生效。

命令示例如下：

#在 MAIN\_CLK 上的上升沿 source latency 时钟延时是 1.8ns:

set\_clock\_latency 1.8 -rise [get\_clocks MAIN\_CLK] -source

#在所有 clock 上的下降沿 source latency 时钟延时是 2.1ns

set\_clock\_latency 2.1 -fall [all\_clocks] -source

#### 6.2.4 set\_clock\_uncertainty

set\_clock\_uncertainty 的作用见 set\_clock\_latency 处解释。

set\_clock\_uncertainty

[<uncertainty\_value>]

[**-setup**] [**-hold**]

[<object\_list> |

-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>]

[-disable]

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参数列表，| 表示或符号，同级的多个选项只能使用一个：

-from : 表示 launch clock，不考虑 clock 的极性，使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

-rise\_from : 表示只考虑 launch clock 为上升沿的情况，使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

-fall\_from : 表示只考虑 launch clock 为下降沿的情况，使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

-to : 表示 capture clock，并不考虑 clock 的极性，使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

-rise\_to : 表示只考虑 capture clock 为上升沿的情况，使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

-fall\_to : 表示只考虑 capture clock 为下降沿的情况，使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

-setup : 表示 uncertainty value 只用在 setup 分析中。

-hold : 表示 uncertainty value 只用在 hold 分析中。

uncertainty : uncertainty 值，为浮点数，单位是 ns。

object\_list : 如果触发器时钟端的 pin 或者 port 上面定义了多个时钟，而用户只想为其中的一个时钟定义 uncertainty，那么 object 可以是用户定义的一个 clock，如果用户想为触发器是终端的 pin 上面的所有 clock 定义同一个 uncertainty，那个 object 可以是一个 pin 或者 port。

注意：所有的 object 必须通过 get\_clocks、get\_pins、get\_ports 三个命令得到，使用 get\_clock 时需要遵循先声明后使用的原则，即：该 clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

[-disable] : 使本条约束不生效。

注意：上面[object\_list], [-from from\_clock | -rise\_from rise\_from\_clock | -fall\_from fall\_from\_clock]以及[-to to\_clock | -rise\_to rise\_to\_clock | -fall\_to fall\_to\_clock ]之间有一个或符号，这表示用户定义此命令时只能选用两种格式中的一种，要么选用 object\_list，要么选

用 -from -to 的形式来定义。

时钟周期抖动及时序检查中的额外留边可以使用 set\_clock\_uncertainty 来设置，指定 setup, hold 来改变有效时钟周期，如下图所示：

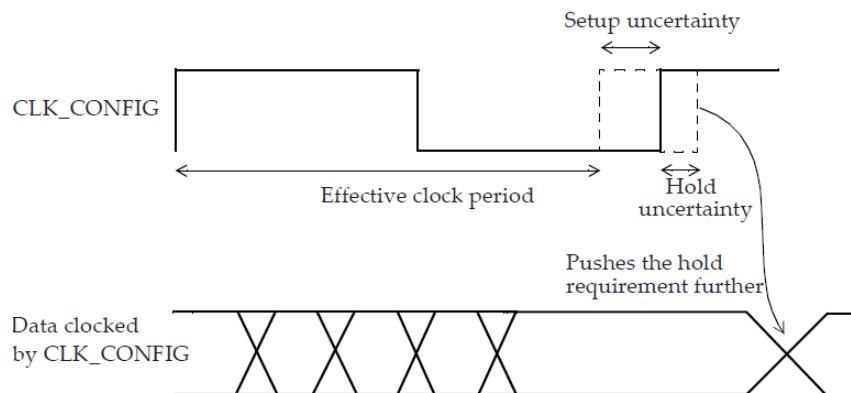


图 6-9 set\_clock\_uncertainty 示意图

命令如下：

```
set_clock_uncertainty -setup 0.2 [get_clocks CLK_CONFIG]
set_clock_uncertainty -hold 0.05 [get_clocks CLK_CONFIG]
```

注：PDS 软件会通过命令 set\_clock\_uncertainty 默认给约束的时钟添加一定的 jitter，比如 50ps 的 system jitter 和 150ps 的 pll jitter。

### 6.2.5 set\_clock\_group

使用该命令时，option group 中的 clock 需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建，然后才能在 set\_clock\_groups 中使用，命令格式如下：

```
set_clock_groups
[-name group_name]
[-asynchronous | -logically_exclusive | -physically_exclusive]
-group <clock_list>*
[-disable]
```

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参数列表，\*表示该参数可以重复，| 表示或符号，同级的多个选项只能使用一个：

**[-name]:** clock group 的名字。

**[-logically\_exclusive] :** 表示 clock group 内的各个 clock 是逻辑独立的。所谓逻辑独立，

即在物理上不同 clock 与 clock pin 之间都存在时钟通路，但逻辑上通路不会同时打开，典型如 mux，例子如下：

例子 1：Logically exclusive with no interaction

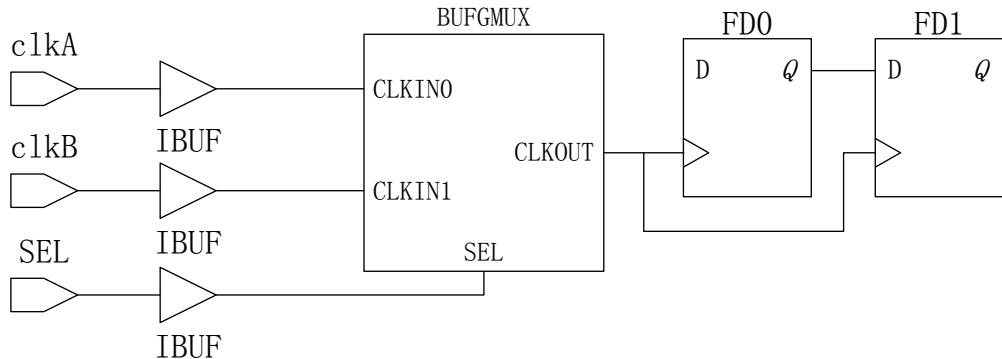


图 6-10 Logically exclusive with no interaction 示例

本例中，clkA 和 clkB 之间不存在任何逻辑路径，也不由任何 BUFGMUX 驱动的组合逻辑单元共享。因此，我们可以直接在 clkA 和 clkB 之间直接设置 exclusive clock group，这样并没有遗漏任何真正的时序路径免于分析。约束命令为：set\_clock\_groups -logically\_exclusive -group clkA -group clkB

例子 2：Logically exclusive with interaction

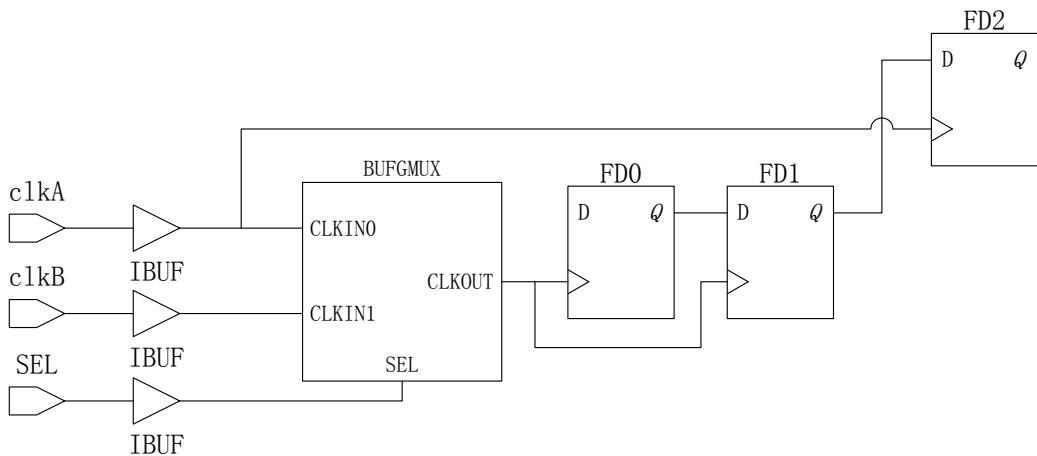


图 6-11 Logically exclusive with interaction 示例

本例中，clkA 和 clkB 输入到一个输出时钟驱动了 FD0 和 FD1 两个触发器的 BUFGMUX。同时 clkA 也驱动了和 FD1 之间有时序路径的触发器 FD2。因此 clkA 和 BUFGMUX 的输出时钟之间有影响。对于这种情况不建议直接在 clkA 和 clkB 之间直接设置时钟组，因为这样设置会忽略 BUFGMUX 选择 clkB 时 FD1 和 FD2 之间的时序分析。所以可以先对 BUFGMUX 的输出时钟设置两个 generated clock，再对这两个生成时钟设置时钟组。

约束命令如下：

```
create_generated_clock -name clkAmux -divide_by 1 \
-source [get_pins BUFGMUX/I0] [get_pins BUFGMUX/O]
create_generated_clock -name clkBmux -divide_by 1 \
-source [get_pins BUFGMUX/I1] [get_pins BUFGMUX/O] \
-add -master_clock clkB
set_clock_groups -logically_exclusive -group clkAmux -group clkBmux
```

-physically\_exclusive : 表示 clock group 内的各个 clock 是物理独立的，所谓物理独立，即各个 clock 之间完全不存在任何的实质连接关系，只是不同分析状况下的不同情况，典型的如在同一个 clock pin 上面定义多个时钟，在实际中，这些时钟显然是不可能同时存在的，例子如下：

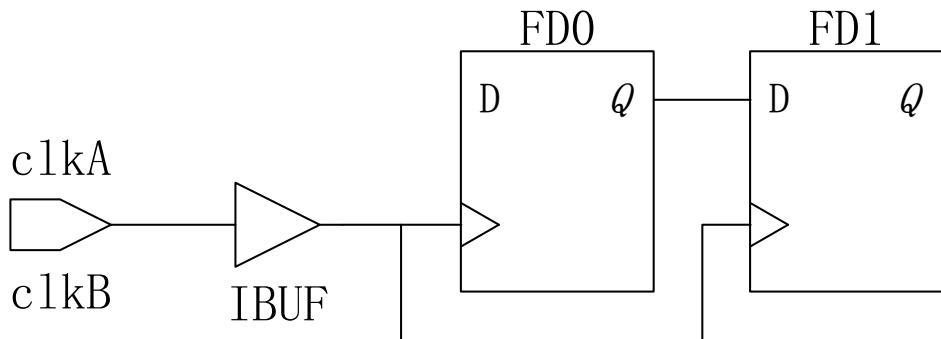


图 6-12 Physically exclusive 示例

本例中，两个主时钟 clkA 和 clkB 被定义在同一个 input port clk 上。

```
create_clock -period 10.000 -name clk_A -waveform {0.000 5.000} \
[get_ports clk]
create_clock -period 20.000 -name clk_B -waveform {0.000 10.000} \
-add [get_ports clk]
```

因为 clkA 和 clkB 不可能同时生效，需要设置 physically exclusive group 来防止在这两个时钟之间进行分析。

```
set_clock_groups -physically_exclusive -group clkA -group clkB
```

对于因为设计逻辑可能引起的时钟扇出，可以在设置约束时使用 -include\_generated\_clock 包含主时钟以及它的生成时钟。

```
set_clock_groups -physically_exclusive \
```

```
-group [get_clocks -include_generated_clock clkA] \
```

```
-group [get_clocks -include_generated_clock clkB]
```

-asynchronous : 表示 clock group 之间的各个 clock 是异步的。软件不会对异步电路进行分析，例子如下：

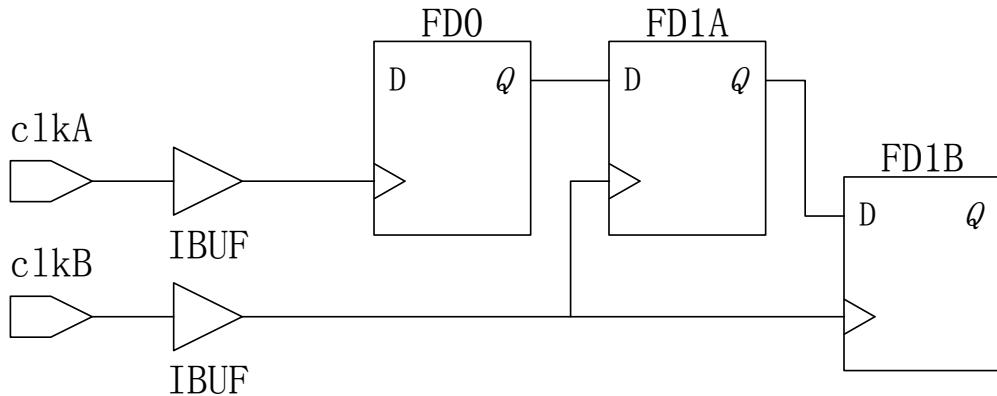


图 6-13 Asynchronous 示例

本例中，clk0 和 clk1 从不同的 port 输入器件，无法知道他们之间的相位关系。因此他们是异步的。默认情况下，从 FD0 到 FD1A 的路径被时序分析工具认为是常规的同步路径，除非添加了时序例外。

因为 clk0 和 clk1 是异步的，不能准确计算 FD1A 触发器 D 端的建立和保持时间裕度，这意味着 FD1A 触发器可能处于亚稳态。在上图例子中，FD1A 和 FD1B 组成双寄存器同步器，可以显著减少亚稳态传播的几率。

而在此例中，只有一条 clk0 到 clk1 的路径而没有 clk1 到 clk0 的路径，因此下面两条约束作用相同。

```
set_clock_groups -asynchronous -group clk0 -group clk1
```

```
set_false_path -from [get_clocks clk0] -to [get_clocks clk1]
```

注：-logically\_exclusive -physically\_exclusive -asynchronous 这三个选项在时序分析过程中是等价的，都表示了时钟之间的异步关系。

-group clock\_list : 注意这个参数的格式后面带了一个“\*”，“\*”表示该参数可以重复，即该命令可以指定多个 group，不同 group 间的两两时钟是异步关系；每个 group 里面可以指定 1 个或多个时钟，同一个 group 内的时钟是同步关系，即会进行时序分析；当只指定 1 个 group 时，那么这个 group 内的时钟之间是同步关系，和不包含在这个 group 内的其他时钟都是异步关系，并且这种异步关系也包括了由他们派生的 generated clock，例子如下：

Example 1: `set_clock_groups -name grp0 -asynchronous -group [get_clocks {CLK}]`, 表示时钟 CLK 和其他时钟（包括其派生时钟）都是异步关系。如果需要时钟 CLK 和它的派生时钟之间进行时序分析，那么需要将派生时钟通过 `create_generated_clock` 约束写出来，然后把他们约束到同一个 group 里面，如下：`set_clock_groups -name grp0 -asynchronous -group {CLK CLK_DRV}`。

Example 2: `set_clock_groups - logically_exclusive -name grp1 -group {CLK33} -group {CLK50}`, 表示且只表示时钟 CLK33 与时钟 CLK50 是异步关系。

Example 3: `set_clock_groups - physically_exclusive -group {CLK1 CLK3} -group {CLK2 CLK4}`, 表示同一个 group 内的时钟 CLK1 CLK3 是同步关系, CLK2 和 CLK4 也是同步关系，会进行时序分析，但两个 group 之间的两两时钟都是异步关系。

Example 4 : `set_clock_groups -name grp0 -asynchronous -group [get_clocks {CLK} -include_generated_clocks]`, 表示时钟 CLK 和由它派生的 generated 时钟之间是同步关系，与其他时钟都是异步的。

## 6.2.6 set\_external\_delay

`set_external_delay` 命令用于设置一个 Output port 和一个 Input port 之间的外反馈路径延时，这个延时被用于 PLL 的外反馈路径延时计算。可以指定最大和最小值，默认情况下这个延时同时用于最大和最小延时分析。命令格式如下：

`set_external_delay`

- from <Output port>

- to <Input port>

[ -min ][ -max ]

[ -add ]

<delay\_value>

[ -disable ]

命令中各参数含义说明如下，方括号表示该参数为可选值，尖括号表示该参数为必选值：

-from: PLL 芯片外外反馈下的 Output port name。

-to: PLL 芯片外外反馈下的 Input port name。

[ -min ]: 表示设置的延时值是最小值。

[ -max ]: 表示设置的延时值是最大值。

**[-add]**: 表示可以在同一个外反馈路径上设置多个延时值，如果之前该路径上面已经定义了一个延时值，并且用户未加-add 参数，那么后面的定义的延时值就会将前面的覆盖掉。

**<delay\_value>**: 延时值，为浮点数，单位是 ns。

**[-disable]**: 使本条约束不生效。

例子 1、在 Port:clkout 和 Port:clkfb 之间设置 1.0ns 的芯片外外反馈延时：

```
set_external_delay -from [get_ports clkout] -to [get_ports clkfb] 1.0
```

## 6.3 IO 时序约束命令

### 6.3.1 set\_input\_delay

**set\_input\_delay** 命令指定了设计接口处输入 port 上输入路径相对于某个时钟边沿的延时，命令格式如下：

```
set_input_delay
  -clock <clock_name>
  [-clock_fall]
  [-rise] [-fall]
  [-max] [-min]
  [-add_delay]
  [-source_latency_included]
  <delay_value>
  <port_list>
  [-disable]
```

**-clock** : 表示其关联的 capture 时钟的名字（-clock 必须定义）。使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

**[-clock\_fall]** : 表示只考虑触发时钟为下降沿的情况。

**[-rise]** : 即只考虑 input 处的输入信号为上升沿的情况。

**[-fall]** : 即只考虑 input 处的输入信号为下降沿的情况。

注意：如果用户二者都没有定义，那么软件将同时考虑 input 处的输入信号的上升或下降的情况。

**[-max]** : 表示用户定义的延时值是最大值。

**[-min]** : 表示用户定义的延时值是最小值。

注意：如果用户二者都没有定义，那么软件将用户定义的延时值同时在最大和最小延时分析时使用。

**[-add\_delay]** : 如果用户之前已经对该 input port 做了 set\_input\_delay 的约束，那么加这个选项表示软件将保留之前的设定的约束值，对于 -max 的情况，在计算时延时值将使用相同情况下按所有值中的用最大值计算，对于 -min 的情况，在计算时将按照使用相同情况下按所有值中的最小值计算。

**[-source\_latency\_included]** : 表示该延时值（数据路径的延迟）已经减掉了时序路径的时钟源延迟，这种情况下，相对于该延时值来说，时钟路径源的延时值为 0。如果没有包含此选项，那么需要在前面通过 set\_clock\_latency 来指定时钟路径的 source latency。如果没有指定，那么默认为 0。

**<delay\_value>** : 延时值，为一个浮点数。

**<port\_list>** : input port，用户必须通过 get\_ports 得到。

**[-disable]** : 使本条约束不生效。

注意：set\_input\_delay 命令对于延时值这一参数的计算方式和输入数据的同步方式有关，接下来分别介绍系统同步和源同步接口下的计算。

**系统同步接口（System Synchronous Interface）** : 发送端和接收端都由共同的系统时钟驱动，则称为系统同步输入（system synchronous input），系统同步接口时钟信号完全依靠系统板级来同步，数据传输延时无法确定，不适用于高速数据传输。

在系统同步接口下，只有边沿对齐模式（时钟和数据在同一时刻变化），根据数据速度和捕获边沿分为上升沿、下降沿和双边沿三种：

边沿对齐模式是最基本的输入延迟模式，时钟在下周期捕获数据，需要同时考虑板上走线和连接线延迟，此时使用边沿对齐模式约束计算公式。

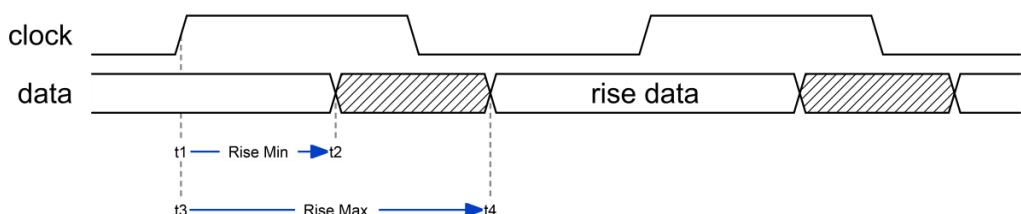


图 6-14 System Edge Rise 波形图

$$\text{Rise Min} = \text{tco\_min} + \text{trce\_dly\_min}$$

$$\text{Rise Max} = \text{tco\_max} + \text{trce\_dly\_max}$$

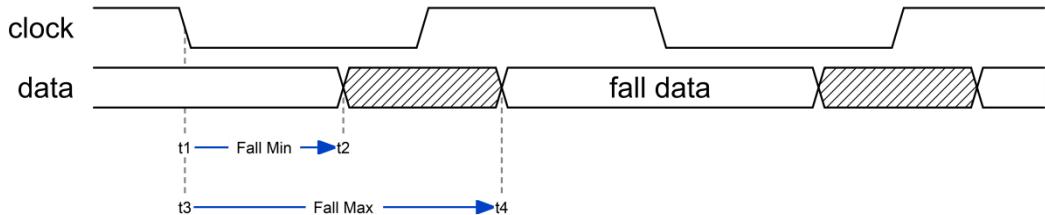


图 6-15 System Edge Fall 波形图

$$\text{Fall Min} = \text{tco\_min} + \text{trce\_dly\_min}$$

$$\text{Fall Max} = \text{tco\_max} + \text{trce\_dly\_max}$$

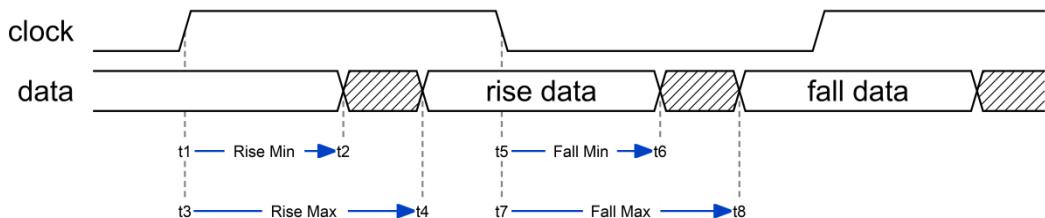


图 6-16 System Edge Dual 波形图

$$\text{Rise Min} = \text{trco\_min} + \text{trce\_dly\_min}$$

$$\text{Rise Max} = \text{trco\_max} + \text{trce\_dly\_max}$$

$$\text{Fall Min} = \text{tfco\_min} + \text{trce\_dly\_min}$$

$$\text{Fall Max} = \text{tfco\_max} + \text{trce\_dly\_max}$$

**源同步接口（Source Synchronous Interface）：**当发送端发送数据的时候，同时发送一路与输入数据同源的时钟信号，输入的数据和时钟保持确定的相位关系，能够支持高速率数据传输。

在源同步接口下，有中心对齐模式（时钟在数据有效窗口中心翻转），直接边沿对齐模式（时钟在数据有效窗口开始时翻转）和 PLL 相位提前边沿对齐模式（时钟在数据有效窗口结束时翻转），根据数据速度和捕获边沿分为上升沿、下降沿和双边沿三种：

中心对齐模式是最常用的源同步模式，使用场景为时钟在数据中心变换，此时使用中心

对齐模式约束计算公式。

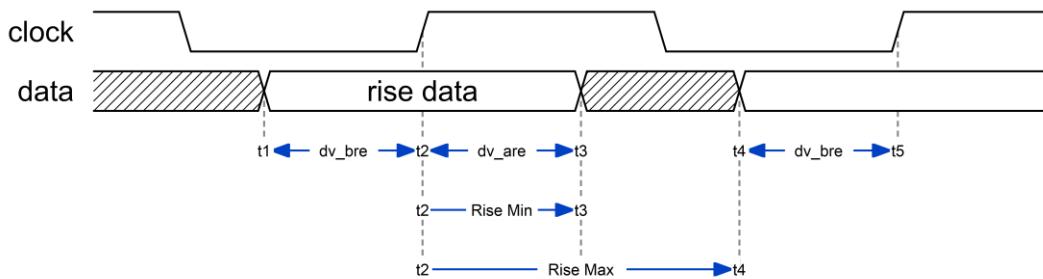


图 6-17 Source Center Rise 波形图

$$\text{Rise Min} = \text{dv\_are}$$

$$\text{Rise Max} = \text{period} - \text{dv\_bre}$$

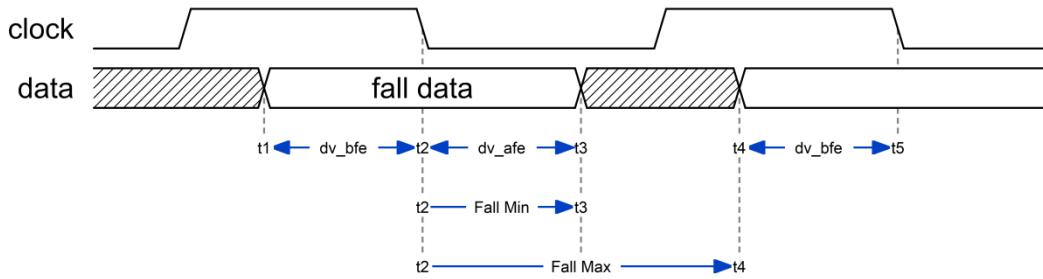


图 6-18 Source Center Fall 波形图

$$\text{Fall Min} = \text{dv\_afe}$$

$$\text{Fall Max} = \text{period} - \text{dv\_bfe}$$

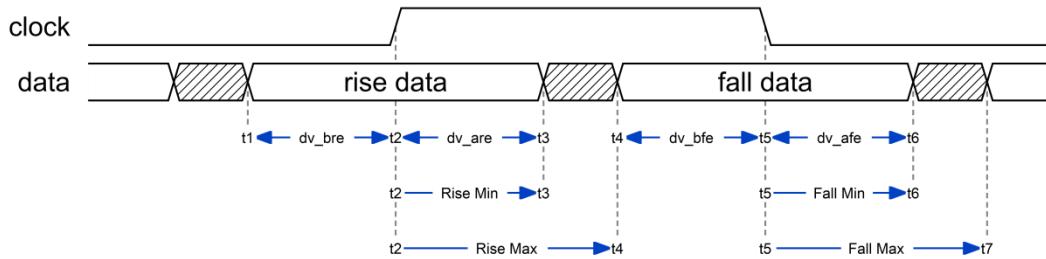


图 6-19 Source Center Dual 波形图

$$\text{Rise Min} = \text{dv\_are}$$

$$\text{Rise Max} = (\text{period}/2) - \text{dv\_bfe}$$

$$\text{Fall Min} = \text{dv\_afe}$$

$$\text{Fall Max} = (\text{period}/2) - \text{dv\_bre}$$

直接边沿对齐模式使用场景为时钟和数据边沿对齐。对于这种情况，为了保证数据被准

确捕获，需要根据情况对时钟和数据使用 IODELAY 进行延迟以达到类似中心对齐的效果，此时使用直接边沿对齐模式约束计算公式。

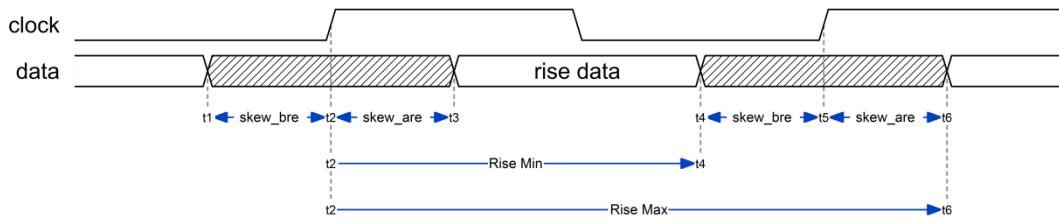


图 6-20 Source Edge Direct Rise 波形图

$$\text{Rise Min} = \text{period} - \text{skew\_bre}$$

$$\text{Rise Max} = \text{period} + \text{skew\_are}$$

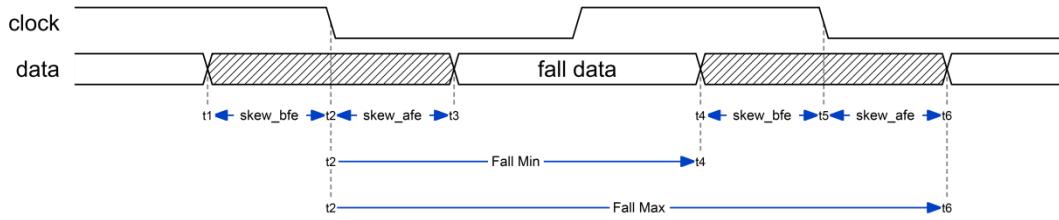


图 6-21 Source Edge Direct Fall 波形图

$$\text{Rise Min} = \text{period} - \text{skew\_bfe}$$

$$\text{Rise Max} = \text{period} + \text{skew\_afe}$$

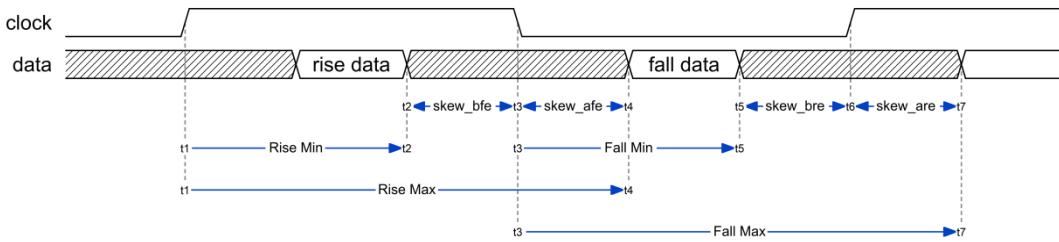


图 6-22 Source Edge Direct Dual 波形图

$$\text{Rise Min} = (\text{period}/2) - \text{skew\_bfe}$$

$$\text{Rise Max} = (\text{period}/2) + \text{skew\_afe}$$

$$\text{Fall Min} = (\text{period}/2) - \text{skew\_bre}$$

$$\text{Fall Max} = (\text{period}/2) + \text{skew\_are}$$

PLL 相位提前边沿对齐模式和直接边沿对齐使用场景类似，使用 PLL 对时钟相位提前以正确捕获数据，此时使用 PLL 相位提前边沿对齐模式约束计算公式。

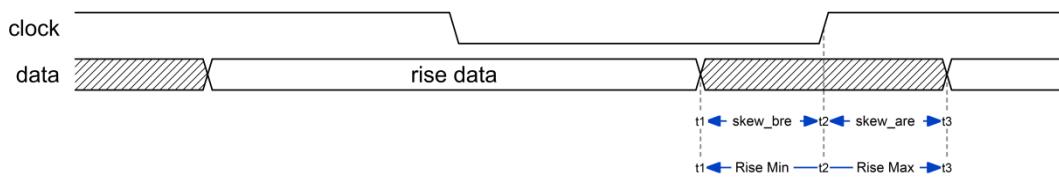


图 6-23 Source Edge PLL Rise 波形图

Rise Min = - skew\_bre

Rise Max = skew\_are

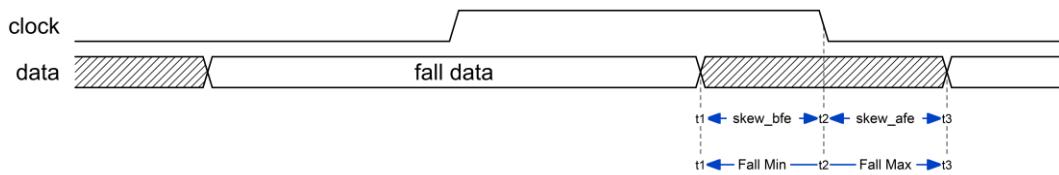


图 6-24 Source Edge PLL Fall 波形图

Fall Min = - skew\_bfe

Fall Max = skew\_afe

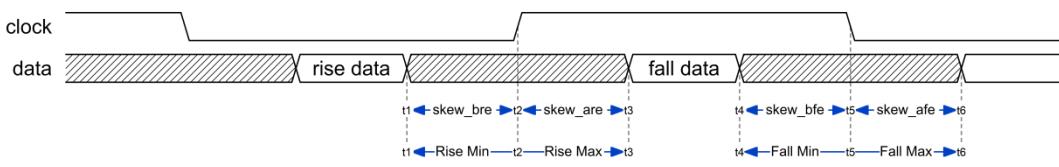


图 6-25 Source Edge PLL Dual 波形图

Rise Min = - skew\_bre

Rise Max = skew\_are

Fall Min = - skew\_bfe

Fall Max = skew\_afe

计算表达式中缩写详细含义如下：

tco\_min : Minimum clock to output delay (external device) 外部器件最小时钟到输出端口延迟

trco\_min : Minimum clock to output delay from rising edge (external device) 上升沿外部器件最小时钟到输出端口延迟

tfco\_min : Minimum clock to output delay from falling edge (external device) 下降沿外部器

件最小时钟到输出端口延迟

tco\_max : Maximum clock to output delay (external device) 外部器件最大时钟到输出端口延迟

trco\_max : Maximum clock to output delay from rising edge (external device) 上升沿外部器件最大时钟到输出端口延迟

tfco\_max : Maximum clock to output delay from falling edge (external device) 下降沿外部器件最大时钟到输出端口延迟

trce\_dly\_min : (Minimum board trace delay) 最小板级延迟

trce\_dly\_max : (Maximum board trace delay) 最大板级延迟

dv\_bre : (Data valid before the rising clock edge) 时钟上升沿前数据最小有效时间

dv\_are : (Data valid after the rising clock edge) 时钟上升沿后数据最小有效时间

dv\_bfe : (Data valid before the falling clock edge) 时钟下降沿前数据最小有效时间

dv\_afe : (Data valid after the falling clock edge) 时钟下降沿后数据最小有效时间

skew\_bre : (Data invalid before the rising clock edge) 时钟上升沿前数据最大无效时间

skew\_are : (Data invalid after the rising clock edge) 时钟上升沿后数据最大无效时间

skew\_bfe : (Data invalid before the falling clock edge) 时钟下降沿前数据最大无效时间

skew\_afe : (Data invalid after the falling clock edge) 时钟下降沿后数据最大无效时间

### 6.3.2 set\_output\_delay

set\_output\_delay 命令指定了设计接口处输出 port 上输出路径相对于某个时钟边沿的延时，命令格式如下：

```
set_output_delay
    -clock <clock_name>
    [-clock_fall]
    [-rise] [-fall]
    [-max] [-min]
    [-add_delay]
    [-source_latency_included]
    <delay_value>
    <port_list>
```

[**-disable**]

**-clock** : 表示其关联的 launch 时钟的名字 (-clock 必须定义)。使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

[**-clock\_fall**] : 表示只考虑触发时钟为下降沿的情况。

[**-rise**] : 即只考虑 output 处的输入信号为上升沿的情况。

[**-fall**] : 即只考虑 output 处的输入信号为下降沿的情况。

注意：如果用户二者都没有定义，那么软件将同时考虑 output 处的输入信号的上升或下降的情况。

[**-max**] : 表示用户定义的延时值是最大值。

[**-min**] : 表示用户定义的延时值是最小值。

注意：如果用户二者都没有定义，那么软件将用户定义的延时值同时在最大和最小延时分析时使用。

[**-add\_delay**] : 如果用户之前已经对该 output port 做了 set\_output\_delay 的约束，那么加这个选项表示软件将保留之前的设定的约束值，对于 -max 的情况，在计算时延时值将使用相同情况下按所有值中的用最大值计算，对于 -min 的情况，在计算时将按照使用相同情况下按所有值中的最小值计算。

[**-source\_latency\_included**] : 表示该延时值（数据路径的延迟）已经减掉了时序路径的时钟源延迟，这种情况下，相对于该延时值来说，时钟路径时钟源的延时值为 0。如果没有包含此选项，那么需要在前面通过 set\_clock\_latency 来指定时钟路径的 source latency。如果没有指定，那么默认为 0。

**<delay\_value>** : 延时值，为一个浮点数。

**<port\_list>** : output port，用户必须通过 get\_ports 得到。

[**-disable**] : 使本条约束不生效。

注意：set\_output\_delay 命令对于延时值这一参数的计算方式和输出数据的同步方式有关，接下来分别介绍系统同步和源同步接口下的计算。

**系统同步接口（System Synchronous Interface）** : 发送端和接收端都由共同的系统时钟驱动，则称为系统同步输入（system synchronous input），系统同步接口时钟信号完全依靠系统板级来同步，数据传输延时无法确定，不适用于高速数据传输。

在系统同步接口下，只有建立/保持模式（计算参数时基于 FPGA 外部数据有效窗口时序特性），根据数据速度和捕获边沿分为上升沿、下降沿和双边沿三种：

建立/保持模式是根据目标器件对于建立和保持时间要求和考虑板上走线或者连接线延迟后设置用户设计的输出延迟，使用建立/保持模式约束计算公式。

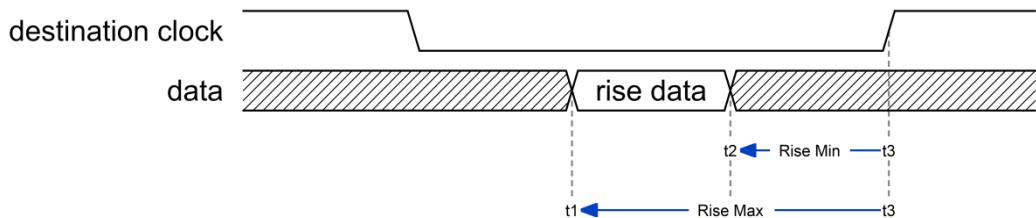


图 6-26 System Setup/Hold Rise 波形图

$$\text{Rise Min} = \text{trce\_dly\_min} - \text{thd}$$

$$\text{Rise Max} = \text{trce\_dly\_max} + \text{tsu}$$

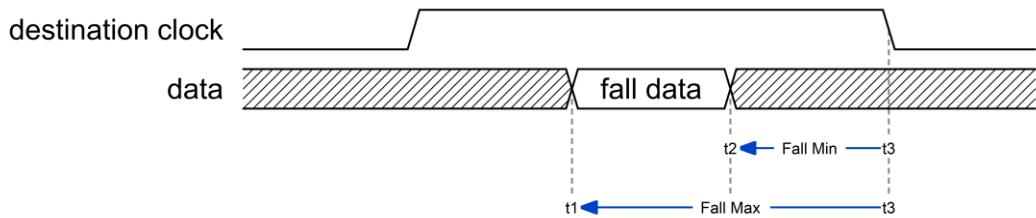


图 6-27 System Setup/Hold Fall 波形图

$$\text{Fall Min} = \text{trce\_dly\_min} - \text{thd}$$

$$\text{Fall Max} = \text{trce\_dly\_max} + \text{tsu}$$

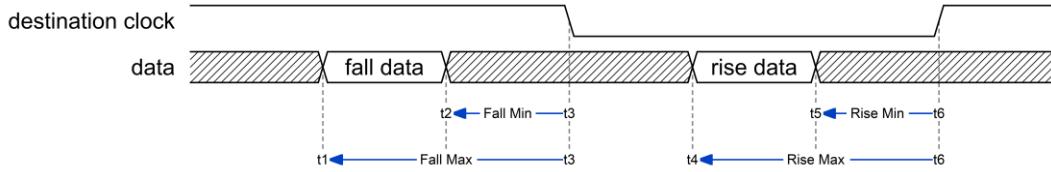


图 6-28 System Setup/Hold Dual 波形图

$$\text{Rise Min} = \text{trce\_dly\_min} - \text{thd\_r}$$

$$\text{Rise Max} = \text{trce\_dly\_max} + \text{tsu\_r}$$

$$\text{Fall Min} = \text{trce\_dly\_min} - \text{thd\_f}$$

$$\text{Fall Max} = \text{trce\_dly\_max} + \text{tsu\_f}$$

源同步接口（Source Synchronous Interface）：当发送端发送数据的时候，同时发送一路与输入数据同源的时钟信号，输出的数据和时钟保持确定的相位关系，能够支持高速率数据传输。

在源同步接口下，有建立/保持模式（计算参数时基于 FPGA 外部数据有效窗口时序特性）和时钟偏斜模式（计算参数时基于 FPGA 输出 port 时钟偏斜要求），根据数据速度和捕获边沿分为上升沿、下降沿和双边沿三种：

建立/保持模式（计算参数时基于 FPGA 外部数据有效窗口时序特性）

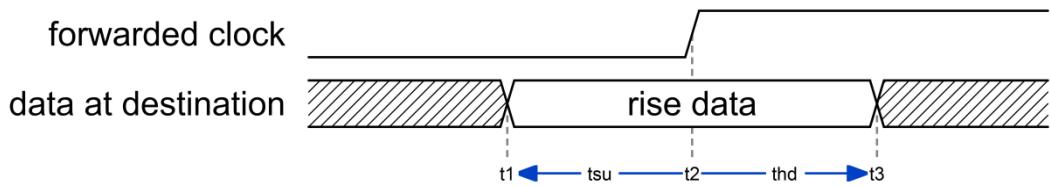


图 6-29 Source Setup/Hold Rise 波形图

$$\text{Rise Min} = \text{trce\_dly\_min} - \text{thd}$$

$$\text{Rise Max} = \text{trce\_dly\_max} + \text{tsu}$$

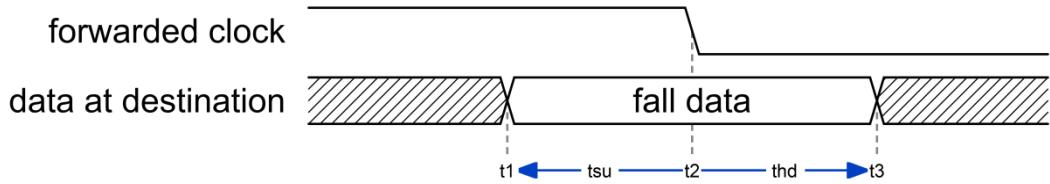


图 6-30 Source Setup/Hold Fall 波形图

$$\text{Fall Min} = \text{trce\_dly\_min} - \text{thd}$$

$$\text{Fall Max} = \text{trce\_dly\_max} + \text{tsu}$$

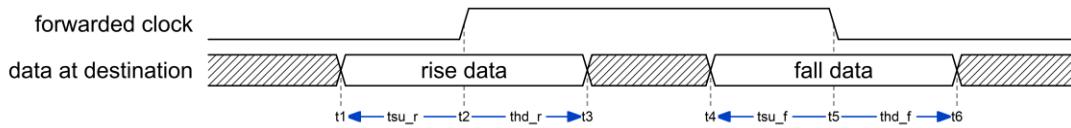


图 6-31 Source Setup/Hold Dual 波形图

$$\text{Rise Min} = \text{trce\_dly\_min} - \text{thd\_r}$$

$$\text{Rise Max} = \text{trce\_dly\_max} + \text{tsu\_r}$$

$$\text{Fall Min} = \text{trce\_dly\_min} - \text{thd\_f}$$

$$\text{Fall Max} = \text{trce\_dly\_max} + \text{tsu\_f}$$

时钟偏斜模式是给定用户设计数据相对随路时钟的时钟偏斜，确认输出延迟是否满足条件，此时使用时钟偏斜模式约束计算公式。

模式（计算参数时基于 FPGA 输出 port 时钟偏斜要求）

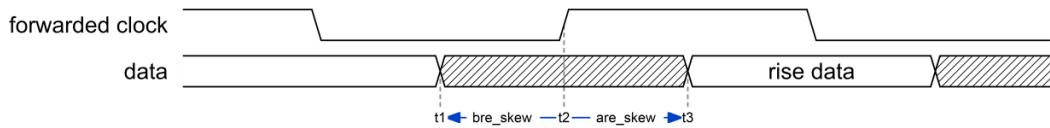


图 6-32 Source Skew Rise 波形图

$$\text{Rise Min} = \text{bre\_skew}$$

$$\text{Rise Max} = \text{period} - \text{are\_skew}$$

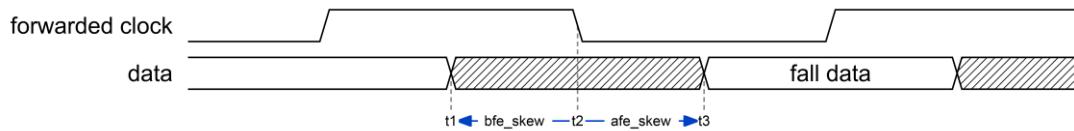


图 6-33 Source Skew Fall 波形图

$$\text{Fall Min} = \text{bfe\_skew}$$

$$\text{Fall Max} = \text{period} - \text{afe\_skew}$$

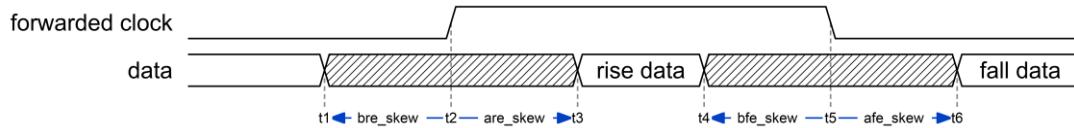


图 6-34 Source Skew Dual 波形图

$$\text{Rise Min} = \text{bfe\_skew}$$

$$\text{Rise Max} = (\text{period}/2) - \text{afe\_skew}$$

$$\text{Fall Min} = \text{bre\_skew}$$

$$\text{Fall Max} = (\text{period}/2) - \text{are\_skew}$$

计算表达式中缩写详细含义如下：

tsu : (Destination device setup time requirement) 目标器件建立时间要求

tsu\_r : (Destination device setup time requirement for rising edge) 目标器件上升沿建立时间

## 要求

tsu\_f : (Destination device setup time requirement for falling edge) 目标器件下降沿建立时间要求

thd : (Destination device hold time requirement) 目标器件保持时间要求

thd\_r : (Destination device hold time requirement for rising edge) 目标器件上升沿保持时间要求

thd\_f : (Destination device hold time requirement for falling edge) 目标器件下降沿保持时间要求

bre\_skew : (Skew requirement before rising edge) 上升沿前时钟偏斜要求

bfe\_skew : (Skew requirement before falling edge) 下降沿前时钟偏斜要求

are\_skew : (Skew requirement after rising edge) 上升沿后时钟偏斜要求

afe\_skew : (Skew requirement after falling edge) 下降沿后时钟偏斜要求

trce\_dly\_min : (Minimum board trace delay) 最小板级延迟

trce\_dly\_max : (Maximum board trace delay) 最大板级延迟

## 6.4 assertion 约束命令

### 6.4.1 set\_max\_skew

这个命令用来约束一组用户指定的 path 之间的最大 skew。被约束的 path 的起点和终点必须满足 startpoint 和 endpoint 的要求。set\_max\_skew 常用于格雷码传输、配置寄存器、带有 CE/MUX/MUX 保持电路的多比特跨时钟实现。set\_max\_skew 的优先级高于 set\_false\_path、set\_clock\_groups，当设置了 set\_false\_path 或者 set\_clock\_groups，set\_max\_skew 不受影响，可以生效；当设置了 set\_max\_delay、set\_min\_delay、set\_multicycle\_path 之后，set\_max\_skew 仍然有效，且不会与它们的优先级发生冲突。

set\_max\_skew :

[ -from <from\_list> | -rise\_from <rise\_from\_list> | -fall\_from <fall\_from\_list> ]

[ -through <through\_list> ]\*

[ -to <to\_list> | -rise\_to <rise\_to\_list> | -fall\_to <fall\_to\_list> ]

<skew\_value>

[ -disable ]

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参数列表，\*表示该参数可以重复，| 表示或符号，同级的多个选项只能使用一个：

[ -from ] : from\_list 表示 path 的 startpoint，注意，startpoint 必须是下列几类 point 中的一种：

1、clock，通过 get\_clocks 得到，需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建；如果定义了 clock，那么 startpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin，或者是与该 clock 相关的所有 input（通过用户 sdc 中的 set\_input\_delay 设定，否则该 startpoint 是无效的）。

2、io port, input 或者是 inout，通过 get\_ports 得到。

3、时序单元，通过 get\_cells 得到，表示所有时序单元的 clock pin。

4、时序单元的 clock pin，通过 get\_pins 得到。

[ -rise\_from ] : 该选项只能指定为时钟 clocks，表示该时序路径起点被时钟上升沿触发，此处的时钟上升沿指的是时钟定义点的上升沿。

[ -fall\_from ] : 该选项只能指定为时钟 clocks，表示该时序路径起点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿。

注意：格式定义中用了三个或符号，如下所示：

[ -from from\_list | -rise\_from rise\_from\_list | -fall\_from fall\_from\_list ] 表示 -from from\_list、-rise\_from rise\_from\_list、-fall\_from fall\_from\_list 只能使用其中的任意一个。

[ -to ] : to\_list 表示 path 的 endpoint，注意，endpoint 必须下列几类 point 中的一种：

1、clock，通过 get\_clocks 得到，需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建；如果定义了 clock，那么 endpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin 所对应的 data pin，或者是与该 clock 相关的所有 output（通过用户 sdc 中的 set\_output\_delay 设定，否则该 endpoint 是无效的）。

2、io port, output 或者是 inout，通过 get\_ports 得到。

3、时序单元，通过 get\_cells 得到，表示所有时序单元的 data pin。

4、时序单元的 data pin，通过 get\_pins 得到。

[ -rise\_to ] : 该选项只能指定为时钟 clocks，表示该时序路径终点被时钟上升沿触发，此

处的时钟上升沿指的是时钟定义点的上升沿。

**[-fall\_to]**：该选项只能指定为时钟 clocks，表示该时序路径终点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿。

注意：格式定义中用了三个或符号，如下所示：

**[-to to\_list | -fall\_to fall\_to\_list | -rise\_to rise\_to\_list]**表示 -to to\_list、-fall\_to fall\_to\_list、-rise\_to rise\_to\_list 只能使用其中的任意一个。

**[-through]**：through\_list 表示 path 必须经过的中间点。

注意：[-through through\_list]\*参数后面都带了\*，表示这个参数可以重复。

**<skew\_value>**：指定这条命令约束的一组 path 的最大 skew 值，单位是 ns。

**[-disable]**：使本条约束不生效。

## 6.5 exception 约束命令

### 6.5.1 set\_max\_delay

这个命令用来指定用户希望某条 path 的最大延迟，但特别要注意的是，这条 path 的起点和终点必须满足 startpoint 和 endpoint 的要求。这条命令是一条 exception 命令，因为一般情况下，能满足 setup 检查的最大延迟，是根据时钟路径的延迟，时钟周期以及 setup 检查的约束值确定的，而如果用户指定了 set\_max\_delay，那么此时能否满足用户要求就和时钟路径、时钟周期及 setup 检查的约束值没有关系了，只要从 startpoint 出来的数据到 endpoint 的延迟小于用户约束的最大值即可。

set\_max\_delay 命令格式如下所示：

set\_max\_delay:

**[-from <from\_list> | -rise\_from <rise\_from\_list> | -fall\_from <fall\_from\_list>]**

**[-through <through\_list>]\***

**[-to <to\_list> | -rise\_to <rise\_to\_list> | -fall\_to <fall\_to\_list>]**

**[-datapath\_only]**

**<delay\_value>**

**[-disable]**

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参

数列表，\*表示该参数可以重复，|表示或符号，同级的多个选项只能使用一个：

[**-from**]：from\_list 表示 path 的 startpoint。注意，startpoint 必须是下列几类 point 中的一种：

①**clock**：通过 get\_clocks 得到，如果定义了 clock，那么 startpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin，或者是与该 clock 相关的所有 input（通过用户 sdc 中的 set\_input\_delay 设定，否则该 startpoint 是无效的），使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②**io port**：input 或者是 inout，通过 get\_ports 得到。

③**时序单元**：通过 get\_cells 得到，表示所有时序单元的 clock pin。

④**时序单元的 clock pin**：通过 get\_pins 得到。

[**-rise\_from**]：该选项只能指定为时钟 clocks，表示该时序路径起点被时钟上升沿触发，此处的时钟上升沿指的是时钟定义点的上升沿。

[**-fall\_from**]：该选项只能指定为时钟 clocks，表示该时序路径起点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿。

[**-to**]：to\_list 表示 path 的 endpoint，注意，endpoint 必须下列几类 point 中的一种：

①**clock**：通过 get\_clocks 得到，如果定义了 clock，那么 endpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin 所对应的 data pin，或者是与该 clock 相关的所有 output（通过用户 sdc 中的 set\_output\_delay 设定，否则该 endpoint 是无效的），使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②**io port**：output 或者是 inout，通过 get\_ports 得到。

③**时序单元**：通过 get\_cells 得到，表示所有时序单元的 data pin。

④**时序单元的 data pin**：通过 get\_pins 得到。

[**-rise\_to**]：该选项只能指定为时钟 clocks，表示该时序路径终点被时钟上升沿触发，此处的时钟上升沿指的是时钟定义点的上升沿。

[**-fall\_to**]：该选项只能指定为时钟 clocks，表示该时序路径终点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿

**-through through\_list**：through\_list 表示 path 必须经过的中间点。考虑到信号在传递过程中的极性可能会发生变化，比如非门。另外提供了下面的两个参数，中间 point 必须是下列几类 point 中的一种：

注意: [-through through\_list]\*参数后面都带了\*, 表示这个参数可以重复。

①组合单元: 通过 get\_cells 得到, 表示该组合单元的任意 input pin。

②组合单元的 input: 通过 get\_pins 得到。

③net: 通过 get\_nets 得到, 表示路径会经过这条 net 的 driver pin。

[-datapath\_only] : 表示指定的 max delay path 不关心时钟的 skew 和 jitter, 只考虑 data path, 包括第一个触发器的 CLK ->Q 延时, 第二个触发器的 setup time, 以及两个触发器之间的组合路径延时。注: 如果一条 max delay path 被设置为 datapath\_only, 由于不考虑时钟的 skew 和 jitter, 那么这一路径的 hold path 也不会被分析。

<delay\_value> : 延时值, 为一个浮点数

[-disable] : 使本条约束不生效。

命令如下:

```
#从触发器 UFF2 的 Q 端到触发器 UFF3 的 D 端的所有路径最大延时是 0.6ns
```

```
set_max_delay 0.6 -from UFF2/Q -to UFF3/D
```

```
#在 SYS_CLK 和 CFG_CLK 时钟域间的所有路径的最大延时是 1.2ns
```

```
set_max_delay 1.2 -from [get_clocks SYS_CLK] -to [get_clocks CFG_CLK]
```

#从触发器 UFF2 的 Q 端到触发器 UFF3 的 D 端的所有数据路径最大延时是 0.6ns, 不考虑 clock skew 和 jitter。

```
set_max_delay 0.6 -from UFF2/Q -to UFF3/D -datapath_only
```

## 6.5.2 set\_min\_delay

这个命令用来指定用户希望某条 path 的最小延迟, 这条 path 的起点和终点必须满足 startpoint 和 endpoint 的要求。这条命令是一条 exception 命令, 因为一般情况下, 能满足 hold 检查的最小延迟, 是根据时钟路径的延迟以及 hold 检查的约束值确定的, 而如果用户指定了 set\_min\_delay, 那么此时能否满足用户要求就和时钟路径及 hold 检查的约束值没有关系了, 只要从 startpoint 出来的数据到 endpoint 的延迟大于用户约束的最小值即可。

set\_min\_delay 命令格式如下所示:

set\_min\_delay:

[-from from\_list]

[-rise\_from from\_list]

[-fall\_from from\_list]

[ -through through\_list]\*

[ -to to\_list]

[ -rise\_to to\_list]

[ -fall\_to to\_list]

<delay\_value>

[ -disable]

命令中各个参数的含义说明如下，方括号表示该参数为可选值，尖括号表示参数值或参数列表，\*表示该参数可以重复，| 表示或符号，同级的多个选项只能使用一个：

[ -from] : from\_list 表示 path 的 startpoint，注意，startpoint 必须是下列几类 point 中的一种：

①clock: 通过 get\_clocks 得到，如果定义了 clock，那么 startpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin，或者是与该 clock 相关的所有 input（通过用户 sdc 中的 set\_input\_delay 设定，否则该 startpoint 是无效的），使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②io port: input 或者是 inout，通过 get\_ports 得到。

③时序单元：通过 get\_cells 得到，表示所有时序单元的 clock pin。

④时序单元的 clock pin: 通过 get\_pins 得到。

[ -rise\_from] : 该选项只能指定为时钟 clocks，表示该时序路径起点被时钟上升沿触发，此处的时钟上升沿指的是时钟定义点的上升沿。

[ -fall\_from] : 该选项只能指定为时钟 clocks，表示该时序路径起点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿。

[ -to] : to\_list 表示 path 的 endpoint，注意，endpoint 必须下列几类 point 中的一种：

①clock: 通过 get\_clocks 得到，如果定义了 clock，那么 endpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin 所对应的 data pin，或者是与该 clock 相关的所有 output（通过用户 sdc 中的 set\_output\_delay 设定，否则该 endpoint 是无效的），使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②io port: output 或者是 inout，通过 get\_ports 得到。

③时序单元：通过 get\_cells 得到，表示所有时序单元的 data pin。

④时序单元的 data pin: 通过 get\_pins 得到。

**[-rise\_to]** : 该选项只能指定为时钟 clocks, 表示该时序路径终点被时钟上升沿触发, 此处的时钟上升沿指的是时钟定义点的上升沿。

**[-fall\_to]** : 该选项只能指定为时钟 clocks, 表示该时序路径终点被时钟下降沿触发, 此处的时钟下降沿指的是时钟定义点的下降沿。

**-through through\_list\*** : through\_list 表示 path 必须经过的中间点。

注意: [-through through\_list]\*参数后面都带了\*, 表示这个参数可以重复。

**<delay\_value>** : 延时值, 为一个浮点数

**[-disable]** : 使本条约束不生效。

命令如下:

```
#从触发器 UFF2 的 Q 端到触发器 UFF3 的 D 端的所有路径最小延时是 0.15ns
```

```
Set_min_delay 0.15 -from UFF2/Q -to UFF3/D
```

```
#在 SYS_CLK 和 CFG_CLK 时钟域间的所有路径的最小延时是 1.2ns
```

```
set_min_delay 0.4 -from [get_clocks SYS_CLK] -to [get_clocks CFG_CLK]
```

### 6.5.3 set\_multicycle\_path

set\_multicycle\_path 命令格式如下所示:

set\_multicycle\_path:

[ -from from\_list ]

[ -rise\_from from\_list ]

[ -fall\_from from\_list ]

[ -through through\_list]\*

[ -to to\_list ]

[ -rise\_to to\_list ]

[ -fall\_to to\_list ]

[ -start ] [ -end ]

[ -setup ] [ -hold ]

**<path\_multiplier>**

[ -disable ]

命令中各个参数的含义说明如下, 方括号表示该参数为可选值, 尖括号表示参数值或参数列表, \*表示该参数可以重复, | 表示或符号, 同级的多个选项只能使用一个:

[ -setup]: 表示 setup 分析时，时钟周期使用指定的倍数。Setup 指定之后，hold 也会受到影响。默认情况下，是使用 setup。

[ -hold]: 表示 hold 分析时，时钟周期使用指定的倍数。

[ -start]: 表示这个多周期的设置是和 launch clock 相关的，假设命令设置的周期数目为 N，那么对于 setup 而言，launch clock 需要使用默认分析边沿的前 N-1 个周期处的边沿（hold 分析的边沿也要移动）；对于 hold 而言，则是使用 launch clock 默认分析边沿的后 N 个周期处的边沿。一般用在 fast to slow 时钟域内，如下图所示：如下图所示：

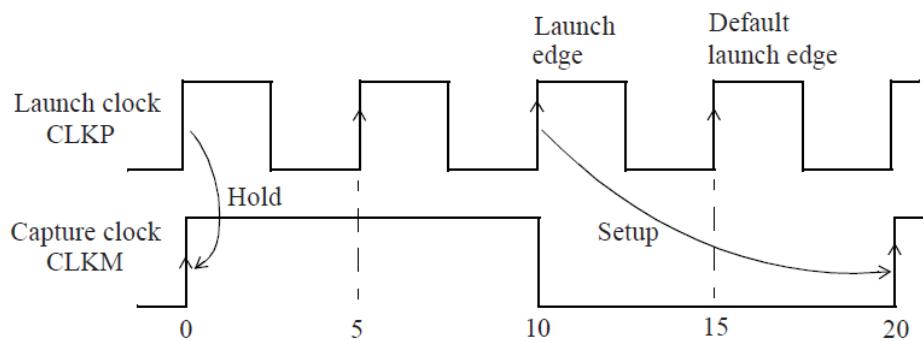


图 6-35 Set Multicycle Path 示例 (start)

上图中默认的时序分析边沿对是 setup: launch clock 15, capture clock 20; hold: launch clock 0, capture clock 0。如果要把 setup 分析是 launch clock 的边沿设定在 10 处（launch clock 往前一个周期），可以这样写：

```
set_multicycle_path 2 -setup -from [get_clocks CLKP] -to [get_clocks CLKM] -start
```

此命令表示：此命令设置了 setup 和 start 选项，即需要分析 launch clock 默认分析边沿的前一个周期处的边沿（hold 边沿也会跟着改变）。设置完该命令之后，时序分析的边沿对分别是，setup: launch clock 10, capture clock 20; hold: launch clock 15, capture clock 20。如果需要 hold 分析如图所示的边沿，还需要将 launch clock 的 hold 分析边沿往后移动一个周期，通过以下命令可以实现：

```
set_multicycle_path 1 -hold -from [get_clocks CLKP] -to [get_clocks CLKM] -start
```

此命令在上一个命令的基础上，launch clock 中将 hold 分析的边沿往后移动了一个周期（setup 不动），结果是 hold: launch clock 0, capture clock 0。（图中 launch 20, capture 20 和 launch 0, capture 0 是一样的边沿对）。

[ -end]: 表示这个多周期的设置是和 capture clock 相关的，假设命令设置的周期数目为 N，那么对于 setup 而言，capture clock 需要使用默认分析边沿的后 N-1 个周期处的边沿（hold 分

析的边沿也要移动)；对于 hold 而言，则是使用 capture clock 默认分析边沿的前 N 个周期处的边沿。一般用在 slow to fast 时钟域内，如下图所示：

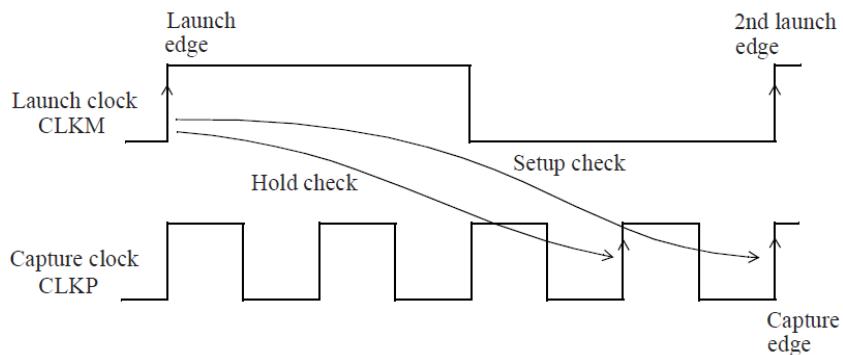


图 6-36 Set Multicycle Path 示例 (end)

上图中的默认的 setup 分析的边沿对是 launch clock 在图中的第一个上升沿和 capture clock 的第二个上升沿，hold 分析的边沿对则是 launch clock 在图中的第一个上升沿和 capture clock 的第一个上升沿。若要改为如图所示的边沿对去分析，则需要将 capture clock 的 setup 和 hold 分析的边沿均往后移动 3 个时钟周期。由于 setup 移动后 hold 会跟着移动，所以可以添加如下命令实现：

```
set_multicycle_path 4 -setup -from [get_clocks CLKM] -to [get_clocks CLKP] -end
```

**[-from]** : from\_list 表示 path 的 startpoint，注意，startpoint 必须是下列几类 point 中的一种：

①clock: 通过 get\_clocks 得到，如果定义了 clock，那么 startpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin，或者是与该 clock 相关的所有 input（通过用户 sdc 中的 set\_input\_delay 设定，否则该 startpoint 是无效的），使用 clock 时需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②io port: input 或者是 inout，通过 get\_ports 得到（注意：该 input 或者是 inout 必须是定义了 set\_input\_delay 的 port，否则该 startpoint 是无效的）。

③时序单元：通过 get\_cells 得到，表示所有时序单元的 clock pin。

④时序单元的 clock pin: 通过 get\_pins 得到。

**[-rise\_from]** : 该选项只能指定为时钟 clocks，表示该时序路径起点被时钟上升沿触发，此处的时钟上升沿指的是时钟定义点的上升沿。

**[-fall\_from]** : 该选项只能指定为时钟 clocks，表示该时序路径起点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿。

[**-to**] : to\_list 表示 path 的 endpoint, 注意, endpoint 必须下列几类 point 中的一种:

①**clock**: 通过 get\_clocks 得到, 如果定义了 clock, 那么 endpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin 所对应的 data pin, 或者是与该 clock 相关的所有 output (通过用户 sdc 中的 set\_output\_delay 设定否则该 endpoint 是无效的), 使用 clock 时需要遵循先声明后使用的原则, 即: clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②**io port**: output 或者是 inout, 通过 get\_ports 得到 (注意: 该 output 或者是 inout 必须是定义了 set\_output\_delay 的 port, 否则该 endpoint 是无效的)。

③**时序单元**: 通过 get\_cells 得到, 表示所有时序单元的 data pin。

④**时序单元的 data pin**: 通过 get\_pins 得到。

**[-rise\_to]**: 该选项只能指定为时钟 clocks, 表示该时序路径终点被时钟上升沿触发, 此处的时钟上升沿指的是时钟定义点的上升沿。

**[-fall\_to]**: 该选项只能指定为时钟 clocks, 表示该时序路径终点被时钟下降沿触发, 此处的时钟下降沿指的是时钟定义点的下降沿。

**[-through]** : through\_list 表示 path 必须经过的中间点。

注意: [-through <through\_list>]\* 参数后面都带了\*, 表示这个参数可以重复。

**<path\_multiplier>** : 表示设置的周期数目, 是一个大于等于 0 的整数。当设置为 0 时, 表示指定的 path 为 zero cycle path。

**[-disable]** : 使本条约束不生效。

#### 6.5.4 set\_false\_path

set\_false\_path 命令格式

如下所示:

set\_false\_path:

**[-from <from\_list> | -rise\_from <rise\_from\_list> | -fall\_from <fall\_from\_list>]**

**[-through through\_list]\***

**[-to <to\_list> | -rise\_to <rise\_to\_list> | -fall\_to <fall\_to\_list>]**

**[-setup] [-hold]**

**[-disable]**

命令中各个参数的含义说明如下, 方括号表示该参数为可选值, 尖括号表示参数值或参数列表, \*表示该参数可以重复, | 表示或符号, 同级的多个选项只能使用一个:

[-setup] : 表示进行的是 setup 分析

[-hold] : 表示进行的是 hold 分析

[ -from] : from\_list 表示 path 的 startpoint, 注意, startpoint 必须是下列几类 point 中的一种:

①clock: 通过 get\_clocks 得到, 如果定义了 clock, 那么 startpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin, 或者是与该 clock 相关的所有 input (通过用户 sdc 中的 set\_input\_delay 设定, 否则该 startpoint 是无效的), 使用 clock 时需要遵循先声明后使用的原则, 即: clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②io port: input 或者是 inout, 通过 get\_ports 得到 (注意: 该 input 或者是 inout 必须是定义了 set\_input\_delay 的 port, 否则该 startpoint 是无效的)。

③时序单元: 通过 get\_cells 得到, 表示所有时序单元的 clock pin。

④时序单元的 clock pin: 通过 get\_pins 得到。

[ -rise\_from] : 该选项只能指定为时钟 clocks, 表示该时序路径起点被时钟上升沿触发, 此处的时钟上升沿指的是时钟定义点的上升沿。

[ -fall\_from] : 该选项只能指定为时钟 clocks, 表示该时序路径起点被时钟下降沿触发, 此处的时钟下降沿指的是时钟定义点的下降沿。

[ -to] : “to\_list”表示 path 的 endpoint, 注意, endpoint 必须下列几类 point 中的一种:

①clock: 通过 get\_clocks 得到, 如果定义了 clock, 那么 endpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin 所对应的 data pin, 或者是与该 clock 相关的所有 output (通过用户 sdc 中的 set\_output\_delay 设定否则该 endpoint 是无效的), 使用 clock 时需要遵循先声明后使用的原则, 即: clock 需要先通过 create\_generated\_clock 或 create\_clock 创建。

②io port: output 或者是 inout, 通过 get\_ports 得到 (注意: 该 output 或者是 inout 必须是定义了 set\_output\_delay 的 port, 否则该 endpoint 是无效的)。

③时序单元: 通过 get\_cells 得到, 表示所有时序单元的 data pin。

④时序单元的 data pin: 通过 get\_pins 得到。

[ -rise\_to] : 该选项只能指定为时钟 clocks, 表示该时序路径终点被时钟上升沿触发, 此处的时钟上升沿指的是时钟定义点的上升沿。

[ -fall\_to] : 该选项只能指定为时钟 clocks, 表示该时序路径终点被时钟下降沿触发, 此处的时钟下降沿指的是时钟定义点的下降沿。

[**-through**] : through\_list 表示 path 必须经过的中间点。

[**-disable**] : 使本条约束不生效。

注意: [-through through\_list]\*参数后面都带了\*, 表示这个参数可以重复。

约束保存文件根据是否使用综合工具, 保存的文件类型不同, 使用综合工具(集成版)时, 约束保存在约束文件(fdc)中; 不使用综合工具(后端版)时, 约束会保存在时序约束文件(sdc)中。

命令如下:

```
#使 ff12 到 ff34 的时序路径失效
set_false_path -from ff12 -to ff34

#使所有到 CLK1 驱动的 endpoint 的 Hold 检查分析失效
set_false_path -hold -to [get_clocks CLK1]
```

## 6.6 other 约束命令

### 6.6.1 set\_disable\_timing

set\_disable\_timing 用于 disable 某段时序弧。命令格式如下:

```
set_disable_timing
[-from <from_pin_name>]
[-to <to_pin_name>]
<object_list>
[-disable]
```

命令中各参数含义说明如下, 方括号表示该参数为可选值, 尖括号表示该参数为必选值:

**-from:** 表示时序弧的起点, 只有 object\_list 指定的对象为 lib\_cell 和 Inst 时可用, 在指定 cell 中只 disable 起点为此 from\_pin 的 timing arc, from\_pin\_name 应对应 object\_list 中 lib\_cell 或 Inst 的 pin name。

**-to:** , 表示时序弧的终点, 只有 object\_list 指定的对象为 lib\_cell 和 Inst 时可用, 在指定 cell 中只 disable 终点为此 to\_pin 的 timing arc,to\_pin\_name 应对应 object\_list 中 lib\_cell 或 Inst 的 pin name。

**<object\_list>:** 表示指定对象的 list, 对象可以为 lib\_cell、inst、pin、port, 目前不支持 hierarchical inst。

[**-disable**]：使本条约束不生效。

例子 1、**disable Inst:FF** 的 timing arc，不指定**-from** 或 **-to** 默认只会打断 clock pin 和输出 pin 之间的 timing arc：

```
set_disable_timing [get_cells FF]
```

例子 2、**disable Inst:LUT I0 和 Z 之间的 timing arc** 以打断 combinational loop：

```
set_disable_timing -from I0 -to Z [get_cells LUT]
```

## 6.7 报告命令

### 6.7.1 report\_timing

```
report_timing
```

```
[-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_type delay_type]
```

```
[-nworst paths_per_endpoint]
```

```
[-max_path count]
```

[**-from**]：from\_list 表示 path 的 startpoint，注意，startpoint 必须是下列几类 point 中的一种：

1、clock，通过 get\_clocks 得到，需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建；如果定义了 clock，那么 startpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin，或者是与该 clock 相关的所有 input（通过用户 sdc 中的 set\_input\_delay 设定，否则该 startpoint 是无效的）。

2、io port, input 或者是 inout，通过 get\_ports 得到。

3、时序单元，通过 get\_cells 得到，表示所有时序单元的 clock pin。

4、时序单元的 clock pin，通过 get\_pins 得到。

[**-rise\_from**]：该选项只能指定为时钟 clocks，表示该时序路径起点被时钟上升沿触发，此处的时钟上升沿指的是时钟定义点的上升沿。

[**-fall\_from**]：该选项只能指定为时钟 clocks，表示该时序路径起点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿。

注意：格式定义中用了三个或符号，如下所示：

[**-from from\_list | -rise\_from rise\_from\_list | -fall\_from fall\_from\_list**]表示-**from** **from\_list**、**-rise\_from rise\_from\_list**、**-fall\_from fall\_from\_list** 只能使用其中的任意一个。

[**-to**] : **to\_list** 表示 path 的 endpoint，注意，endpoint 必须下列几类 point 中的一种：

1、clock，通过 get\_clocks 得到，需要遵循先声明后使用的原则，即：clock 需要先通过 create\_generated\_clock 或 create\_clock 创建；如果定义了 clock，那么 endpoint 是该 clock network 能够 fanout 到的所有 register 的 clock pin 所对应的 data pin，或者是与该 clock 相关的所有 output（通过用户 sdc 中的 set\_output\_delay 设定否则该 endpoint 是无效的）。

2、io port, output 或者是 inout，通过 get\_ports 得到。

3、时序单元，通过 get\_cells 得到，表示所有时序单元的 data pin。

4、时序单元的 data pin，通过 get\_pins 得到。

[**-rise\_to**] : 该选项只能指定为时钟 clocks，表示该时序路径终点被时钟上升沿触发，此处的时钟上升沿指的是时钟定义点的上升沿。

[**-fall\_to**] : 该选项只能指定为时钟 clocks，表示该时序路径终点被时钟下降沿触发，此处的时钟下降沿指的是时钟定义点的下降沿。

注意：格式定义中用了三个或符号，如下所示：

[**-to to\_list | -fall\_to fall\_to\_list | -rise\_to rise\_to\_list**]表示-**to** **to\_list**、**-fall\_to** **fall\_to\_list**、**-rise\_to** **rise\_to\_list** 只能使用其中的任意一个。

[**-through**] : **through\_list** 表示 path 必须经过的中间点。

注意：[-through through\_list]\*参数后面都带了\*，表示这个参数可以重复。

**-delay\_type**: 表示 path 的延迟类型，有三种，min、max、min\_max 三种，其中 min 表示此分析进行是 hold 分析，max 表示此分析进行的是 setup 分析，min\_max 则要同时进行 setup 和 hold 的分析。

**-nworst paths\_per\_endpoint** : 表示每个 path 组的每个 endpoint 的需要报告的 path 数目。

**-max\_path count** : 每个 path 组需要报告的 path 的数目。

## 6.8 命令的语法格式

### 6.8.1 tcl 的命令格式

下面介绍下 tcl 的命令格式:

Tcl 的命令由以下几个部分组成:

[1] Commands 即命令名, 比如的 set\_input\_delay, 一个命令中可以包含其他的子命令, 分号或者是换行符作为命令的结束符。子命令以右边大括号做为结束符, 见下面子命令。

[2] Words 即字符, 字符以空格作为分隔符。

[3] Double quotes 即双引号, 作用是将多个字符看做是一个字符串而非单个字符。在双引号的范围内, 不论是分号, 右边大括号, 还是空格键或换行符都会被当做字符看待, 子命令, 变量, 以及反斜杠依然有效。

[4] Braces。即大括号, 所有的子命令, 变量以在大括号内都会被当做字符看待。

[5] Command substitution 子命令, 以左中括号开头, 并以右中括号结束。

[6] Variable substitution 变量定义。以\$作为其起始符。在解析时, 变量的值会将变量替换。

[7] Backslash substitution 即反斜杠。和 C 及其他语言的反斜杠一样, tcl 语言的反斜杠也是用在转义符中。

[8] Comments.注释符, 用#开头。

### 6.8.2 List 的问题

下面看 from\_list、rise\_from\_list、fall\_from\_list、to\_list、fall\_to\_list、rise\_to\_list、through\_list、rise\_through\_list、 fall\_through\_list 的写法, 在前面介绍过, 需要通过 get\_pins 等命令还获取, 那么应该按照 tcl 的语法如下书写:

```
set_min_delay -rise_through [get_ports clk]
```

如果希望获取多个 port, 那么可以这么写:

```
set_min_delay -rise_from [get_ports {clk rst_n}] -rise_to [get_ports {clk1 rst_n1}]
```

如果的 list 来自 port, 也来自 pin, 那么应该分开写, 假设 clk 来自 port, rst\_n 来自 pin, 并且具备有如下关系:

```
pin clk ->port rst_n1;
```

```
pin clk ->pin clk1;
```

```
port rst_n -> port rst_n1;
```

```
port rst_n -> pin clk1;
```

需要四条命令来支持。

```
set_min_delay -rise_through [get_pins clk] -rise_through [get_ports rst_n1]
```

```
set_min_delay -rise_through [get_pins clk] -rise_through [get_pins clk1]
```

```
set_min_delay -rise_through [get_ports rst_n] -rise_through [get_pins clk1]
```

```
set_min_delay -rise_through [get_ports rst_n] -rise_through [get_ports rst_n1]
```

可以看到，当存在多种类型的 list 对象时，需要的命令会呈指数增长，所以在约束的时候，尽量使用同一类型的 list 对象，比如 from 全用 pin，to 全用 port 等，而不要 from 或者 to 同时用到 pin，port。其他的 list 情况一致，不再详述。

## 7 局部动态重配

### 7.1 启用动态重配

#### 7.1.1 操作演示

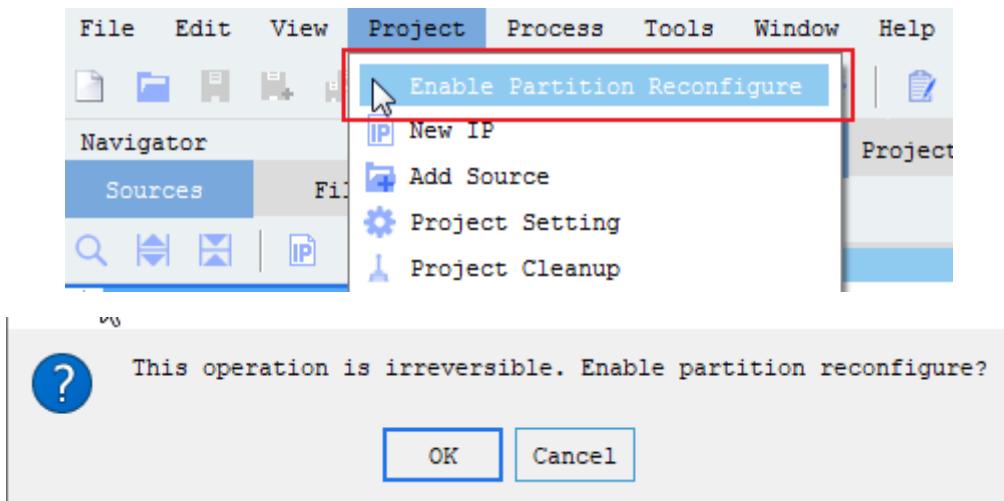


图 7-1 启动局部动态重配

针对某工程启用局部动态重配模式后，该工程不能恢复成普通模式，但是其他工程不受影响，即此操作以工程为单位不可逆。该功能当前仅支持 PG2L100H 与 PG2T390H 两个器件。

在 PDS 中也支持使用 tcl 方式开启动态重配模式，在 PDS 的命令栏中输入 enable\_pr\_flow 命令，即可将符合条件的工程转换为动态重配工程。或者在 PDS\_shell 中输入 enable\_pr\_flow - syn\_tool ads，也可以达到同样的目的。

#### 7.1.2 界面说明

启用局部动态重配模式后，相比普通模式，PDS 界面会在左上角的“Sources”部件增加“Partition Definition”，在“Console”部件增加“Multiple Flows”。

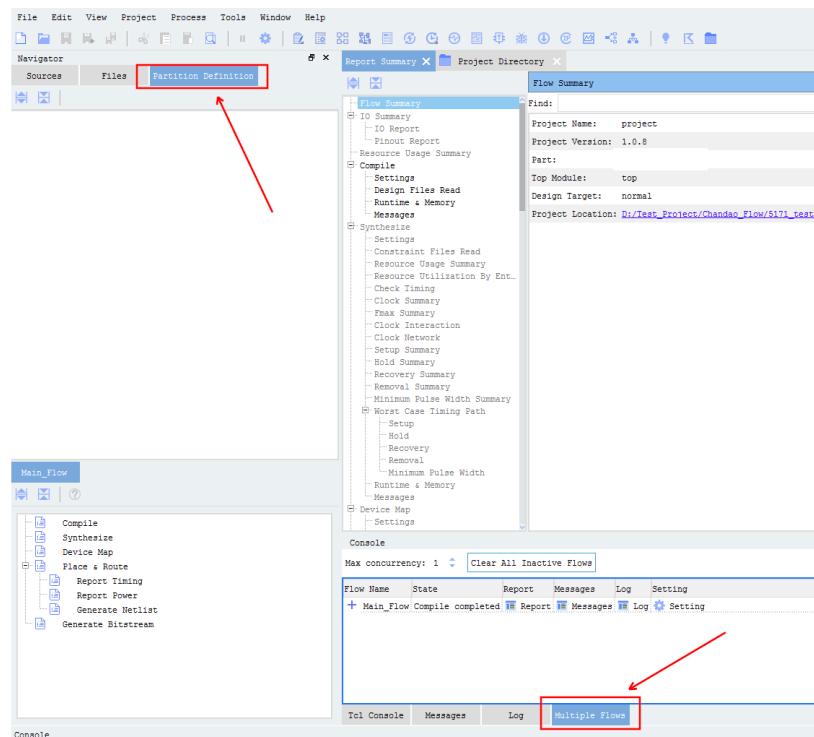


图 7-2 界面显示

## 7.2 创建 Partition Definition

### 7.2.1 操作演示

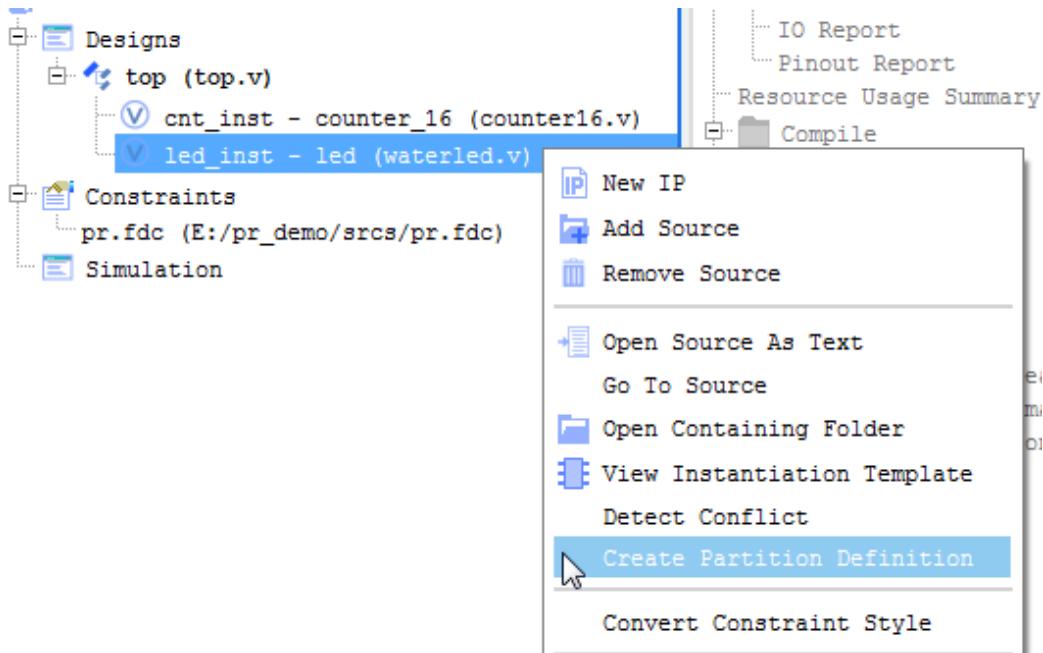


图 7-3 创建 Partition Definition 操作

在“Sources”部件的 Module Hierarchy 右键点击相应节点，选择“Create Partition Definition”。

### 7.2.2 界面说明

相对普通模式，启用局部动态重配后，会在“Navigator”部件新增“Partition Definition”部件，用以显示和管理 PD、RM、RM 的 Module Hierarchy；在“Console”部件新增“Multiple Flows”部件，用以显示、管理及操作流程。后续章节会分别针对各新增部件进行更详细的说明。

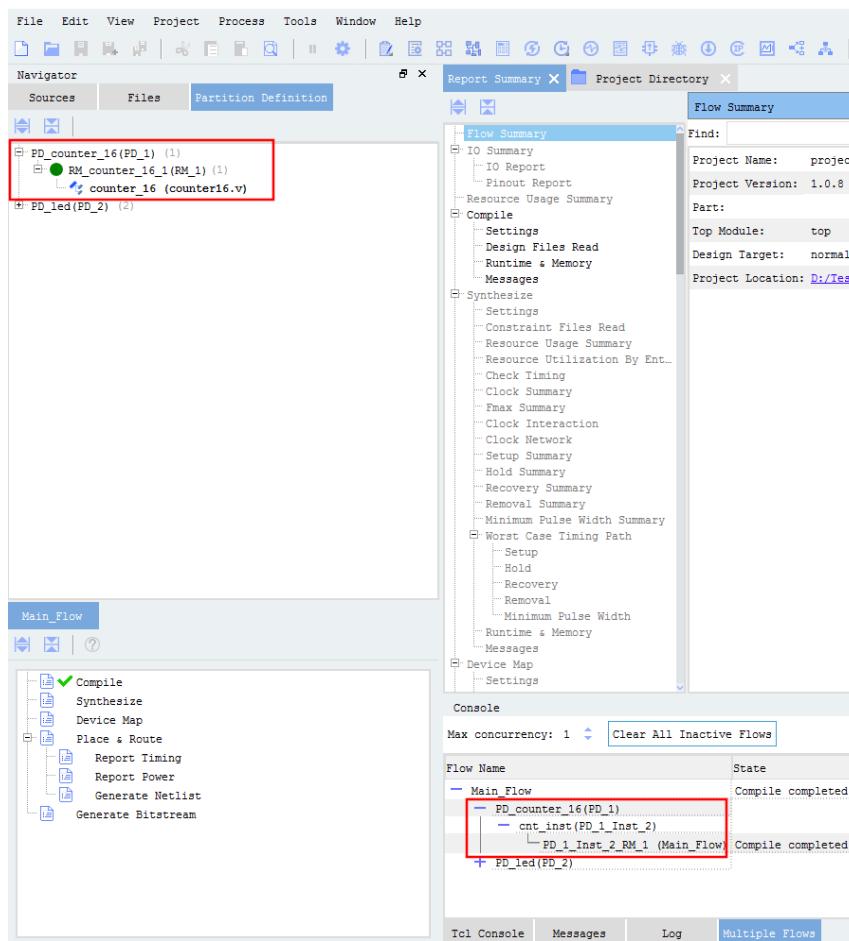


图 7-4 界面说明

### 7.2.3 Partition Definition Instance 图标

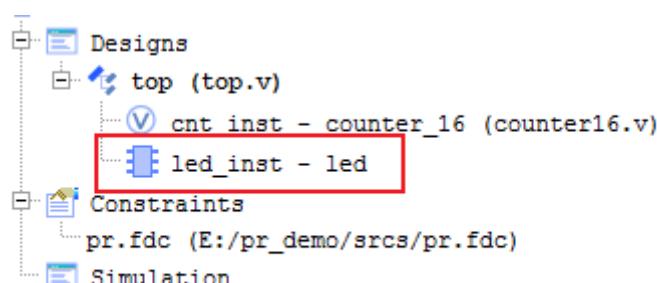


图 7-5PD\_led 图标

led\_inst 是 PD “PD\_led”的 Instance，用上图红框中的图标标识。

#### 7.2.4 Partition Definition

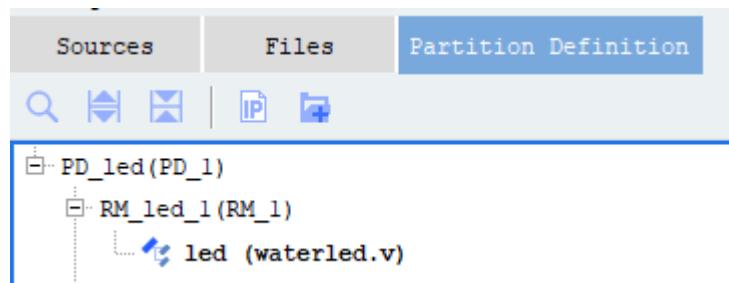


图 7-6 Partition Definition 界面

创建 PD (Partition Definition) 后，会在“Partition Definition”部分显示 PD、PD 管理的 RM (Reconfigure Module) 以及 RM 的 Module Hierarchy。

“PD\_led”是 PD 名称，“RM\_led”是 RM 名称；“PD\_1”是 PD 索引名称，“RM\_1”是 RM 索引名称。

#### 7.2.5 PD 的默认 RM

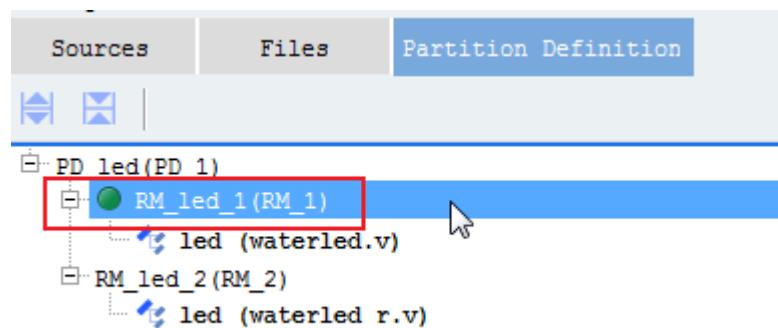


图 7-7 PD 的默认 RM

如上图所示，RM\_led\_1(RM\_1)是 PD\_led(PD\_1)的默认 RM，使用红框中的图标来标识。在运行主流程时，会将 PD 的默认 RM 配置给每个 PD 的 Instance 来运行。

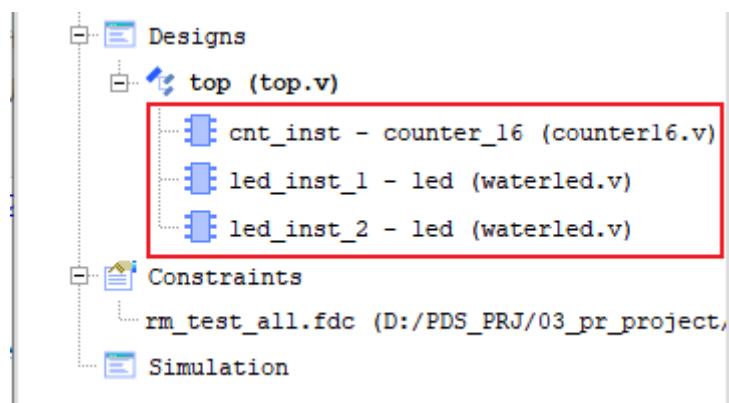


图 7-8 流程运行时

例如，上图所示的例子中，运行主流程时，会将 RM\_led\_1(PD\_1\_RM\_1)配置给

PD\_led(PD\_1)的2个Instance: led\_inst\_1 和 led\_inst\_2。

## 7.2.6 Multiple Flows

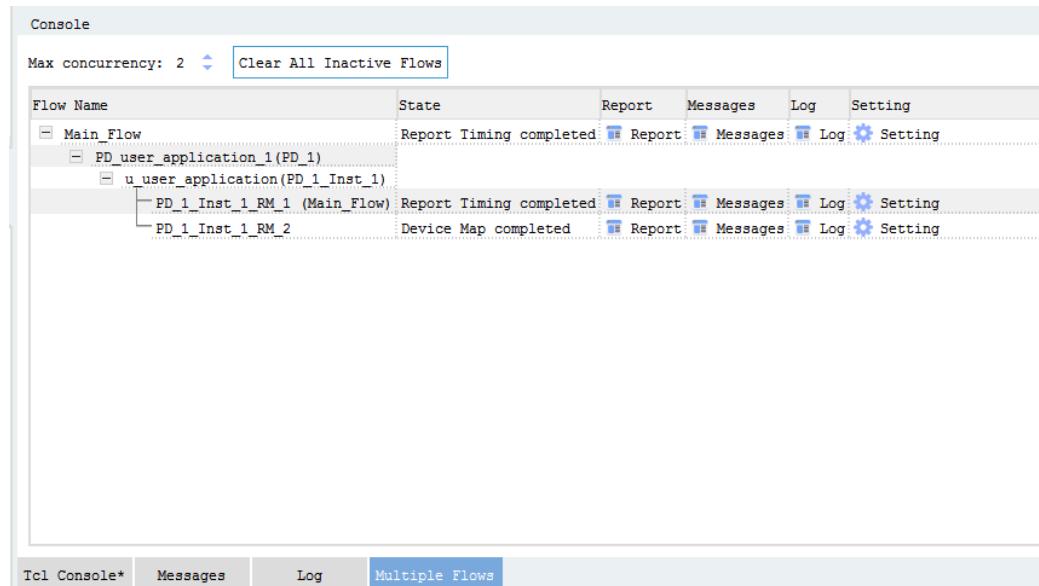


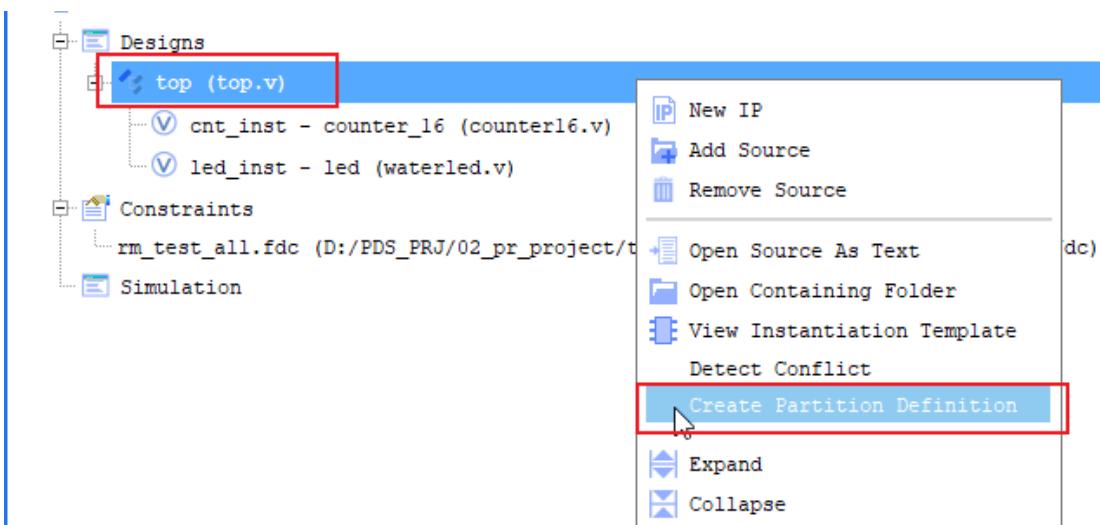
图 7-9 Multiple Flow 窗口

“Main Flow”是主流程，即包含静态区域和动态区域的完整流程；“PD\_led”是PD名称，“led\_inst”是PD\_led的实例名称；“PD\_1\_Inst\_1\_RM\_2”是局部重配流程，意思是led\_inst (PD\_1\_Inst\_1)选用PD\_1\_RM\_1运行流程，即PD\_led (PD\_1)的实例led\_inst选用PD\_1 (PD\_led)管理的RM\_1 (RM\_led\_1)运行流程。

## 7.2.7 创建 PD 的限制

(1) 不能针对 top module 创建 PD。

(2)



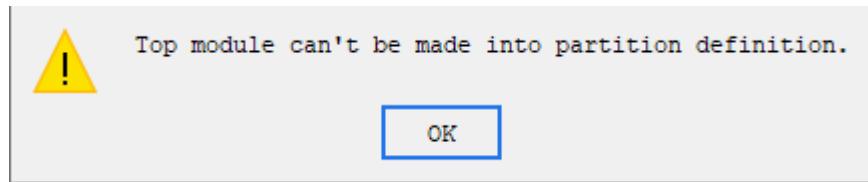


图 7-10 禁止操作 1

(3) 不能针对非激活分支(即该分支的 top module 并非工程的 top module)的节点创建 PD。

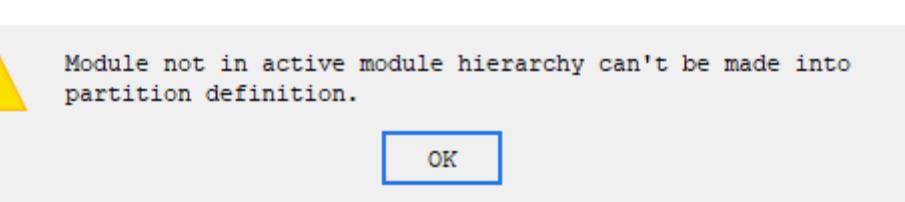
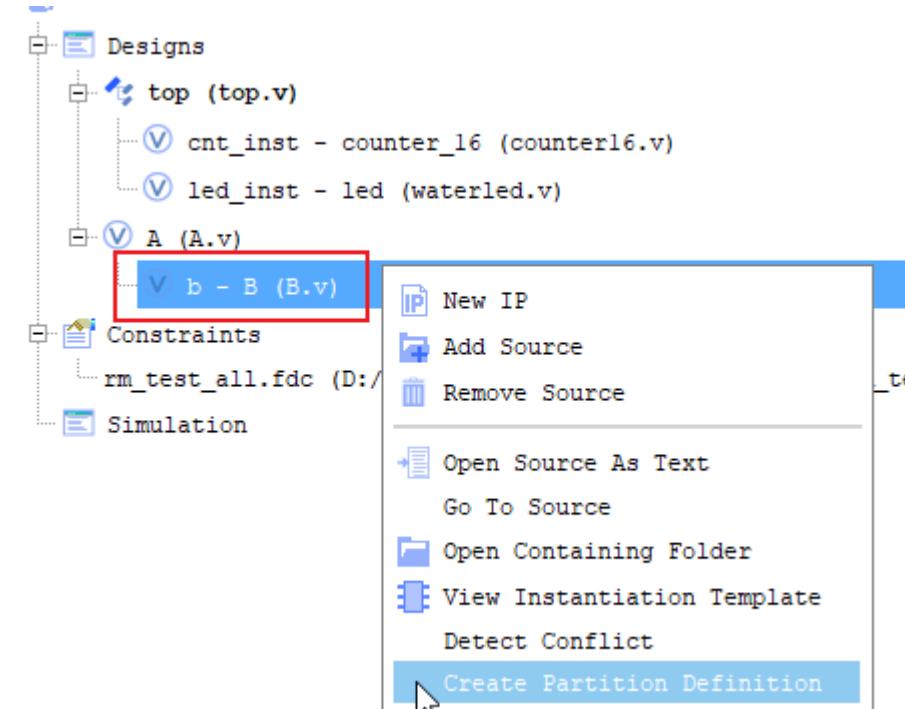


图 7-11 禁止操作 2

(4) 需要创建成 PD 的 module 及其层级下的所有 module 所在的每个 design 文件只能有一个 module 定义。

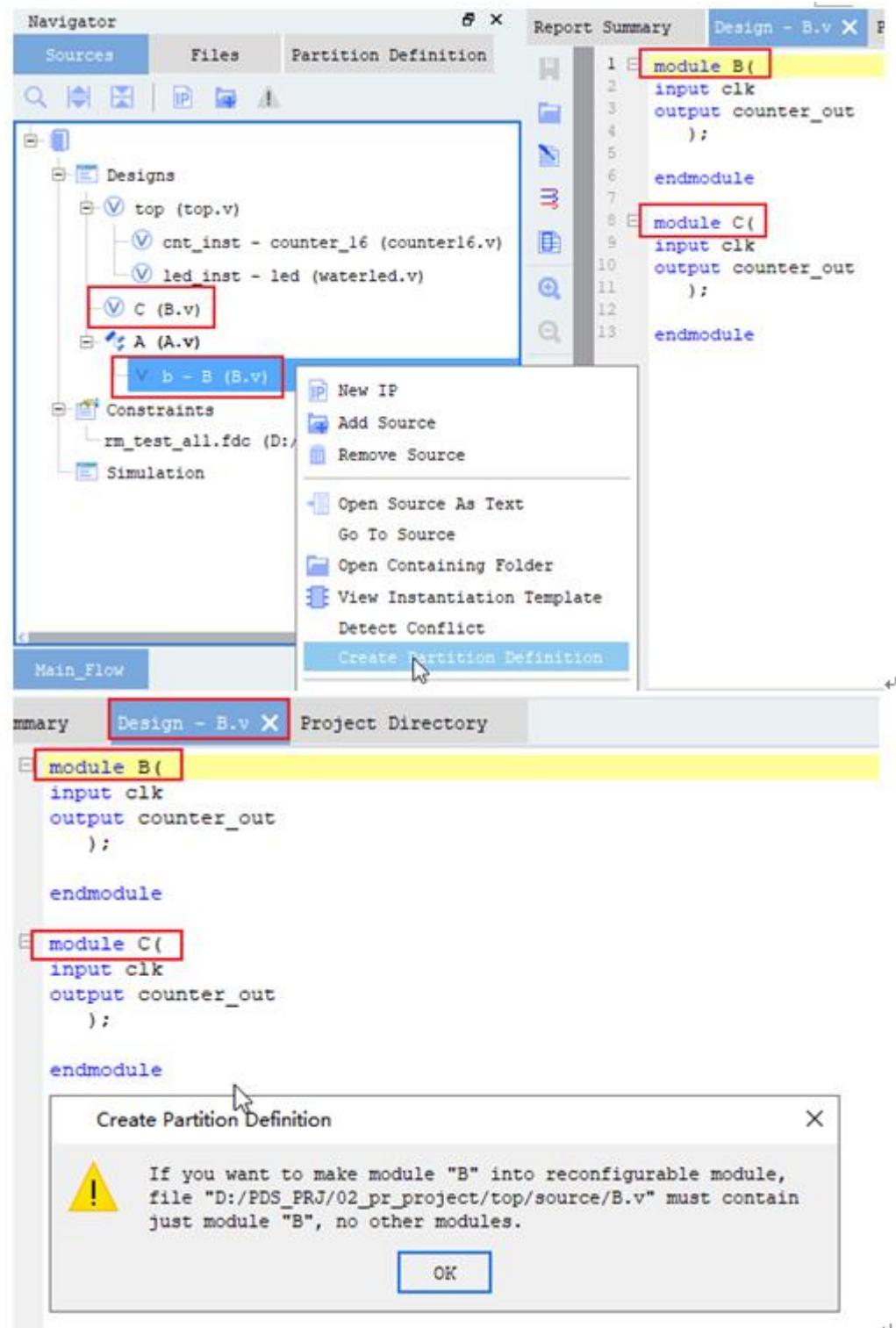


图 7-12 禁止操作 3

(5) IP 不能直接被创建成 PD。

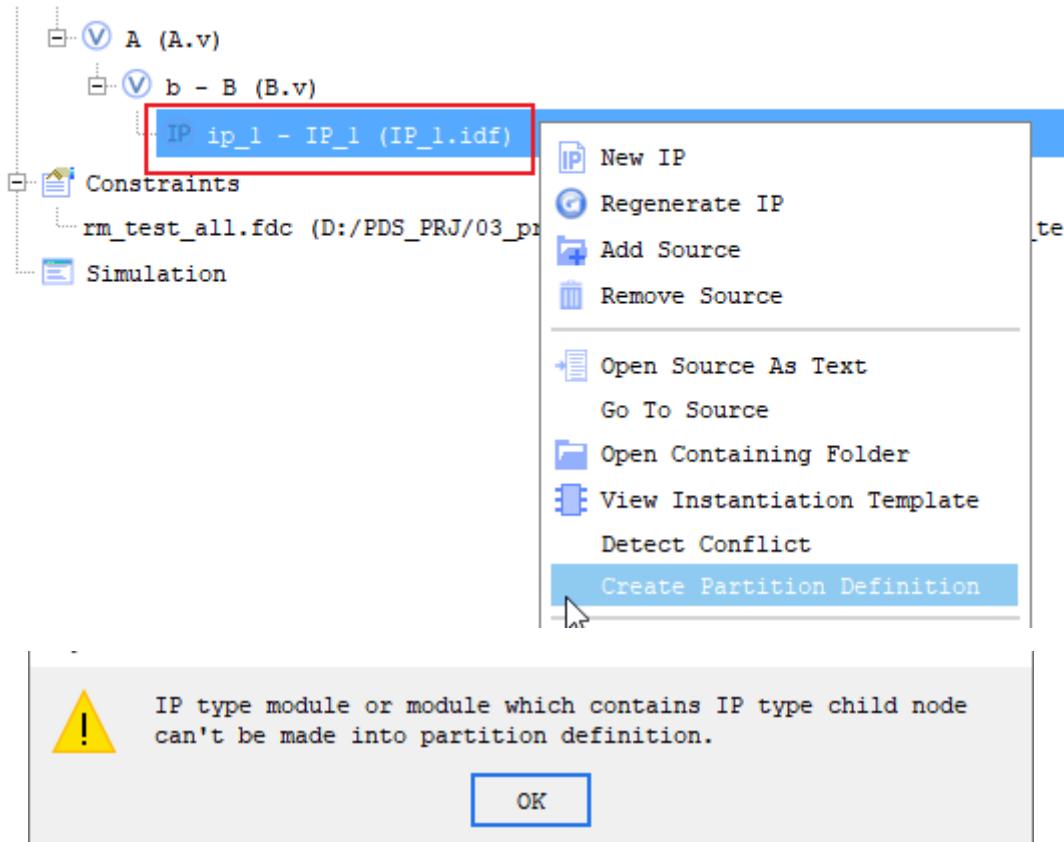


图 7-13 禁止操作 4

(6) 子节点包含 PD 实例的 module 不能被创建成 PD。

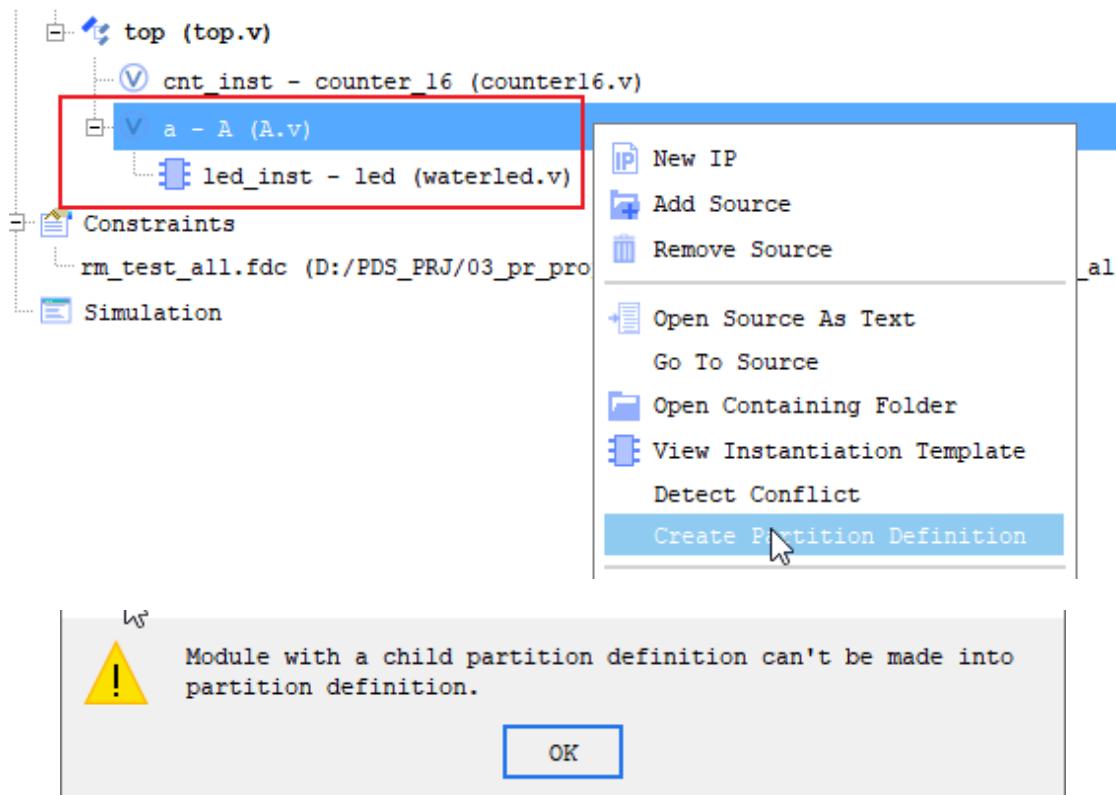


图 7-14 禁止操作 5

(7) Module 实例化时传递的参数值跟 module 定义里对应参数的初始值不相等的情况下，该 module 不能被创建成 PD。

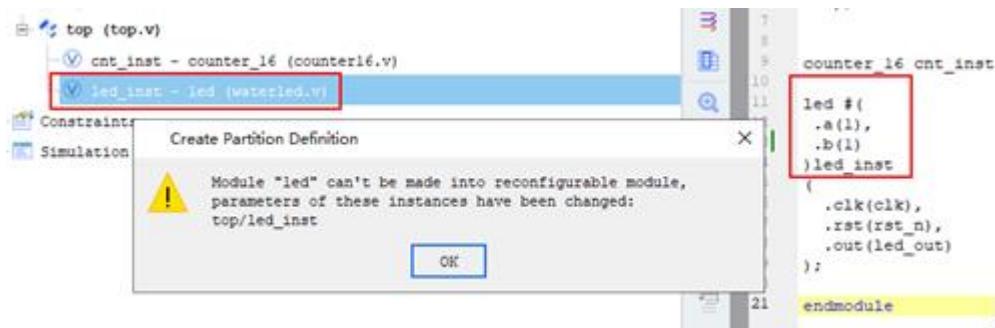


图 7-15 禁止操作 6

(8) 针对 Module 创建 PD 时，如果该 module 层级下有 module 在其它层级中被实例化，则无法创建 PD。

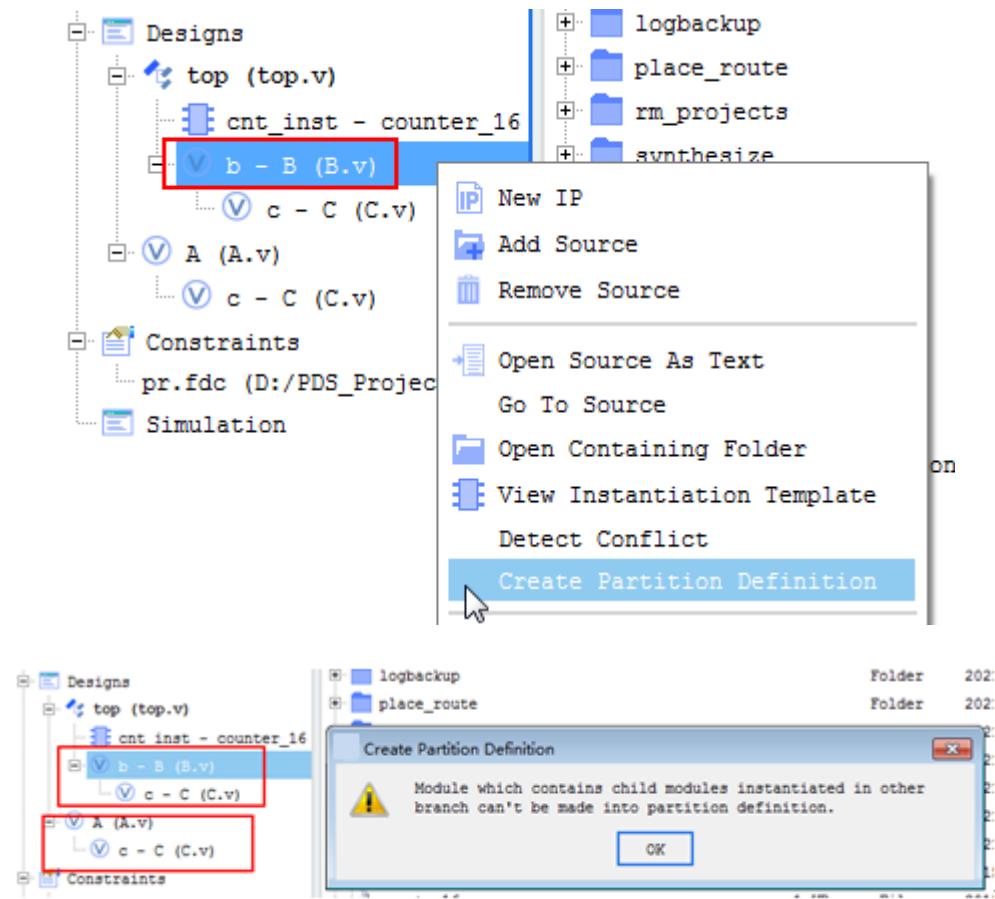


图 7-16 禁止操作 7

### 7.2.8 查看和编辑 PD 属性

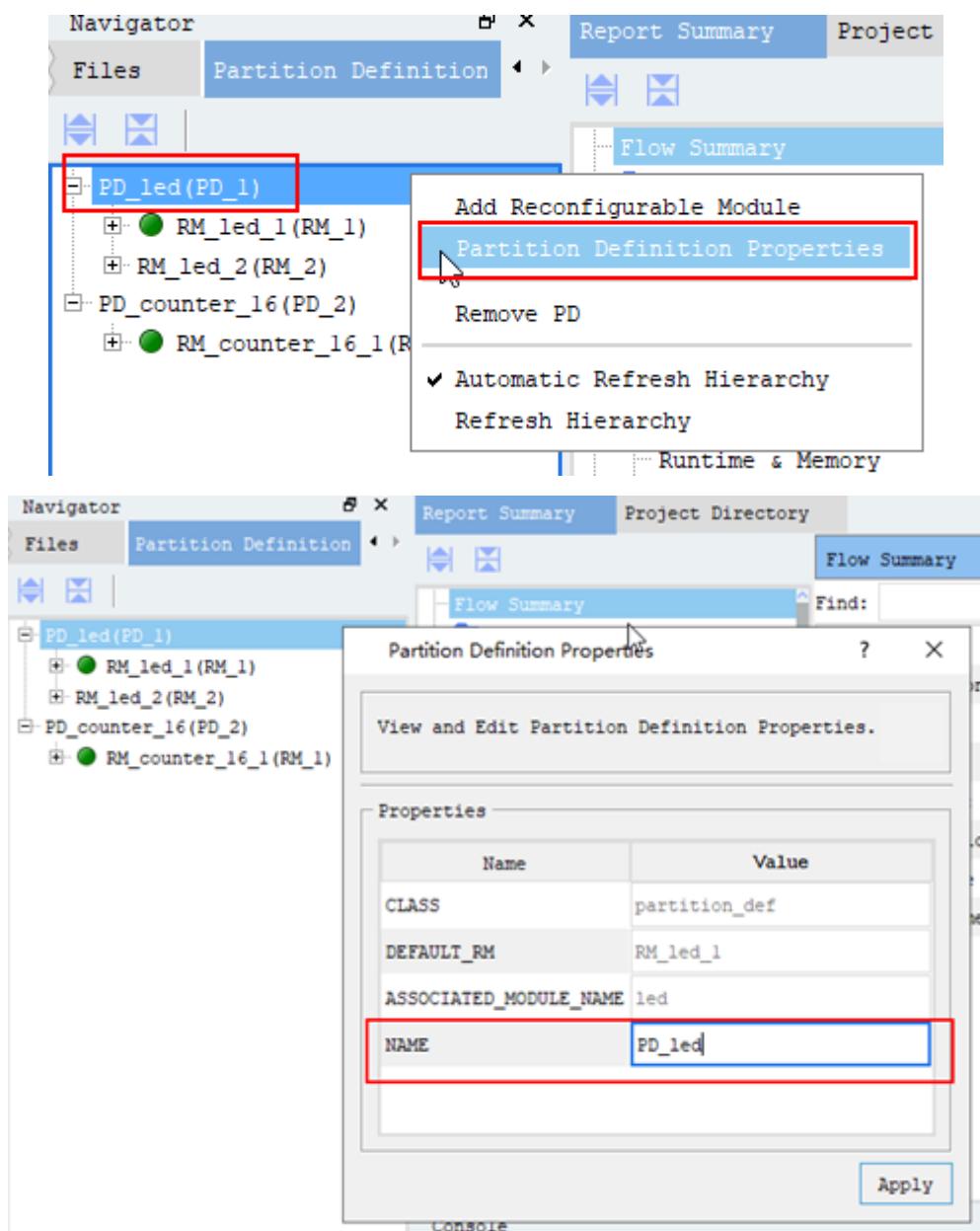


图 7-17 查看和编辑 PD 属性操作

### 7.3 删 除 Partition Definition

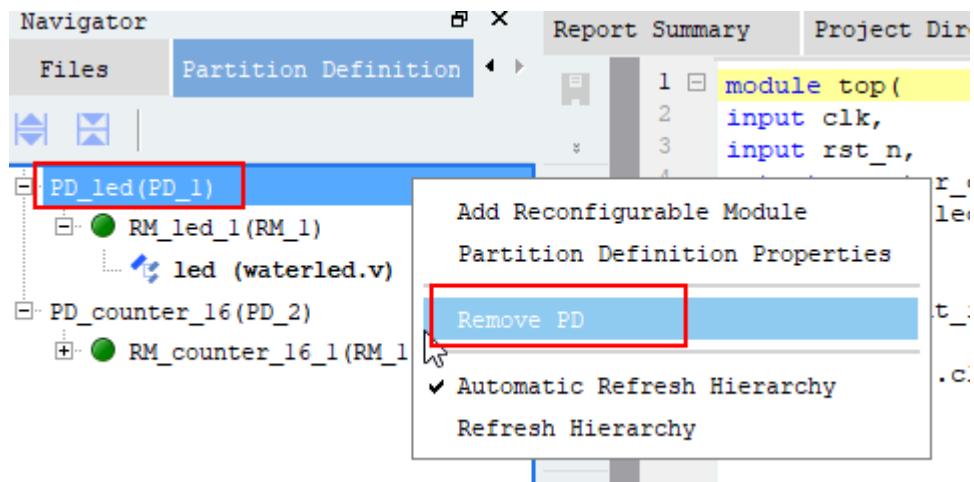


图 7-18 remove PD

删除前界面如下图所示：

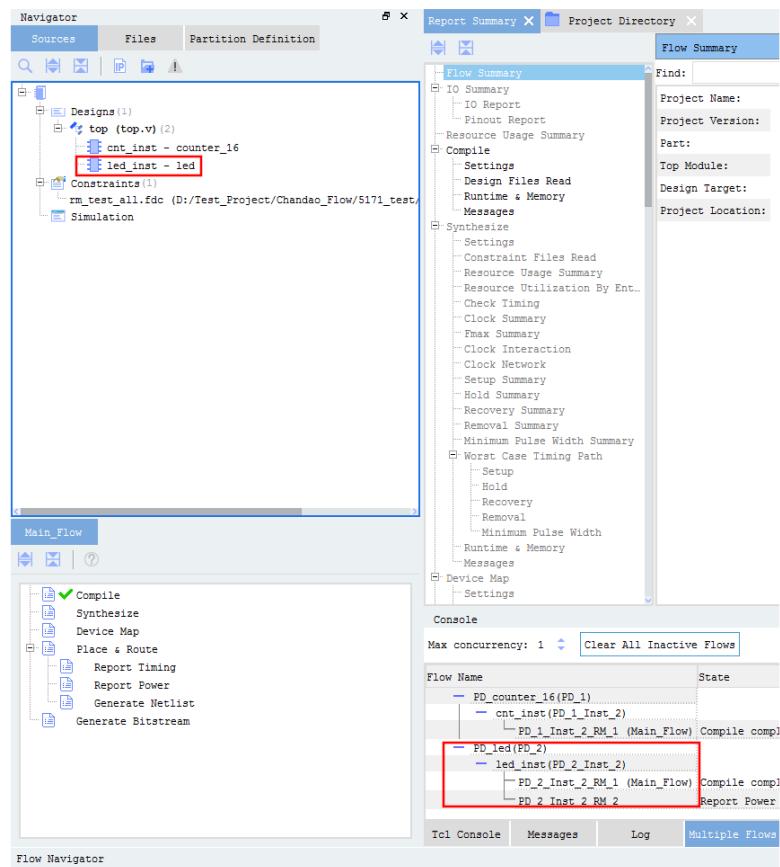


图 7-19 删除前界面

删除后界面如下图所示

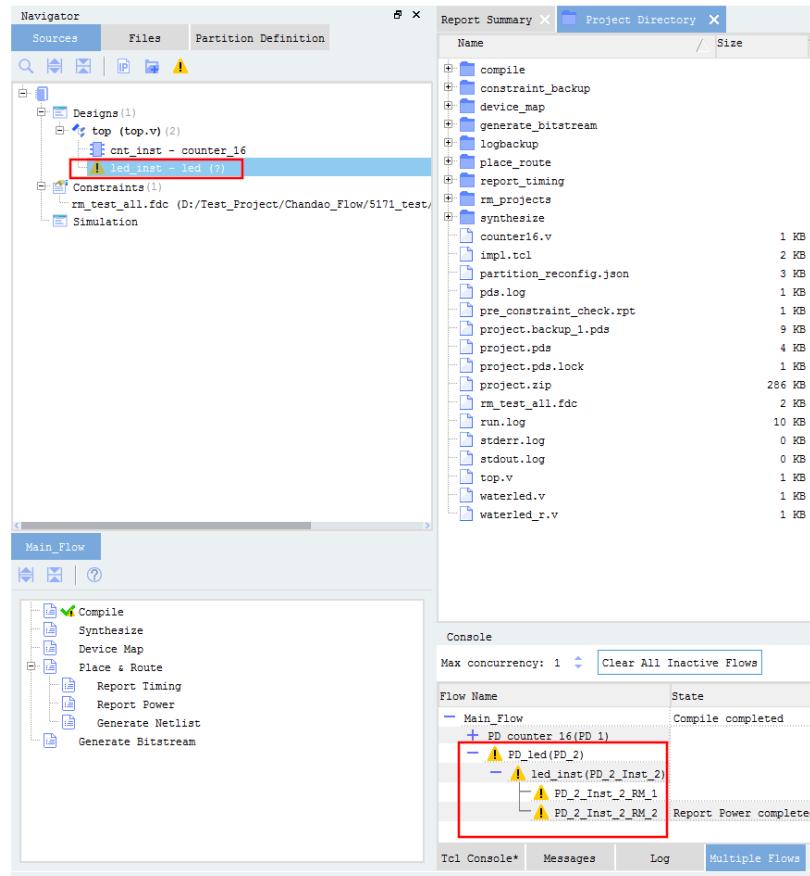


图 7-20 删除后界面

这里的“!”图标表示 RM\_Flow 对应的 PD\_Inst 或 RM 或 PD 不存在。

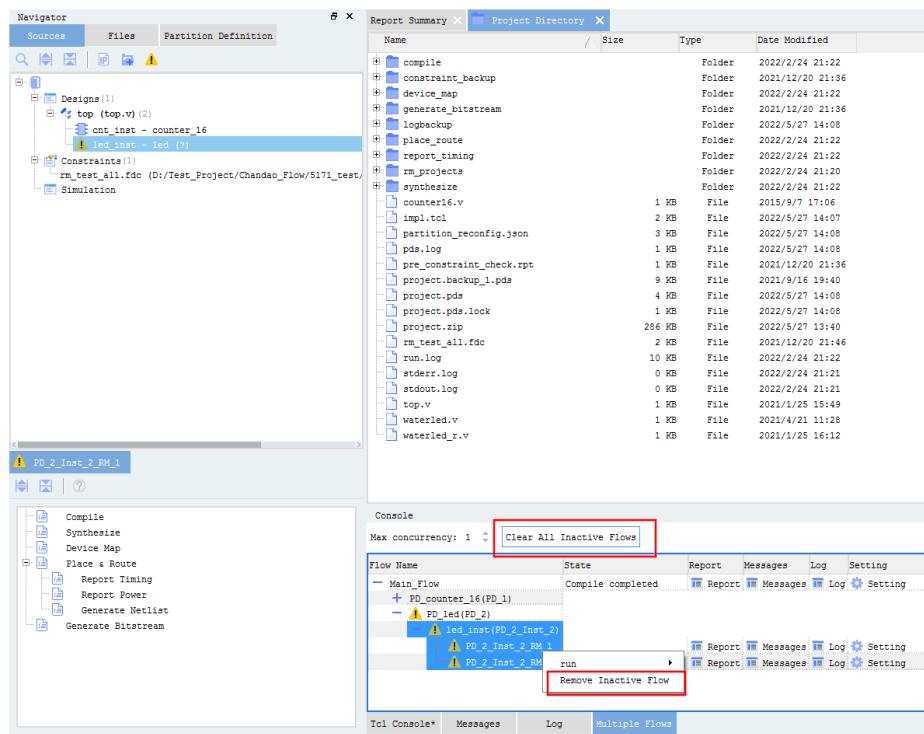


图 7-21 remove inactive flow

可通过以上 2 个操作接口删除 Inactive Flow。

注意：因为创建 PD 时把相应的 design 文件从主工程删除并添加到 PD 的默认 RM 管理，所以删除 PD 后，工程里没有了 PD 的默认 RM 管理的 design 文件，需要用户重新添加这些 design 文件到主工程。

后续会对删除 PD 功能进行优化，用户删除 PD 时提示用户选择 design 文件恢复到主工程。

## 7.4 添加 Reconfigure Module

### 7.4.1 操作演示

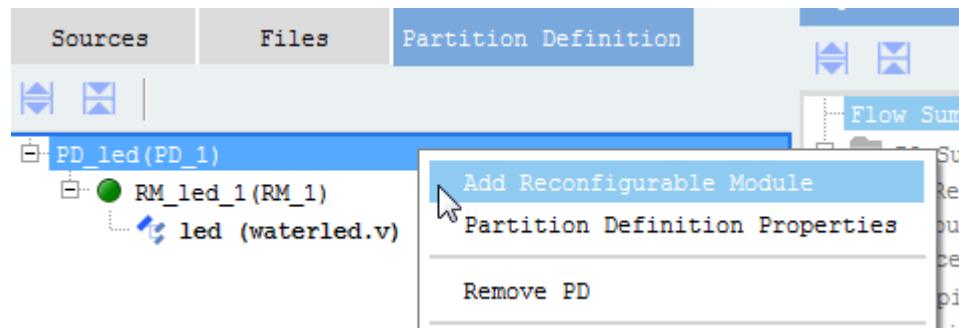


图 7-22 添加 RM

在相应 PD 上点击鼠标右键，选择“Add Reconfigurable Module”。

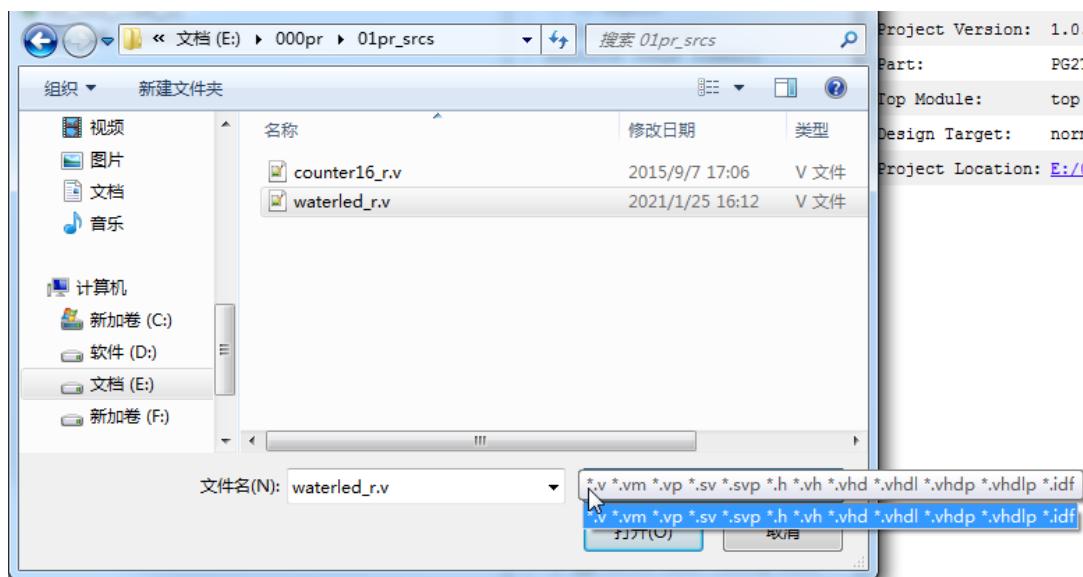


图 7-23 添加页面

选择需要添加到 RM 的 design 文件。

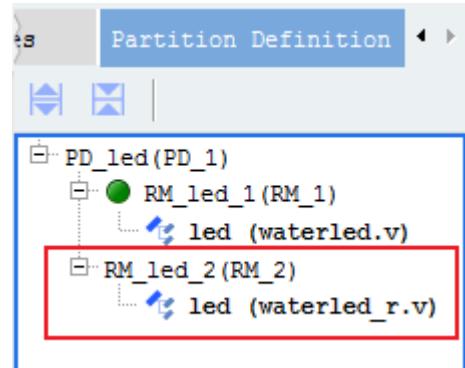


图 7-24 添加 RM 后

#### 7.4.2 查看和更改 RM 属性

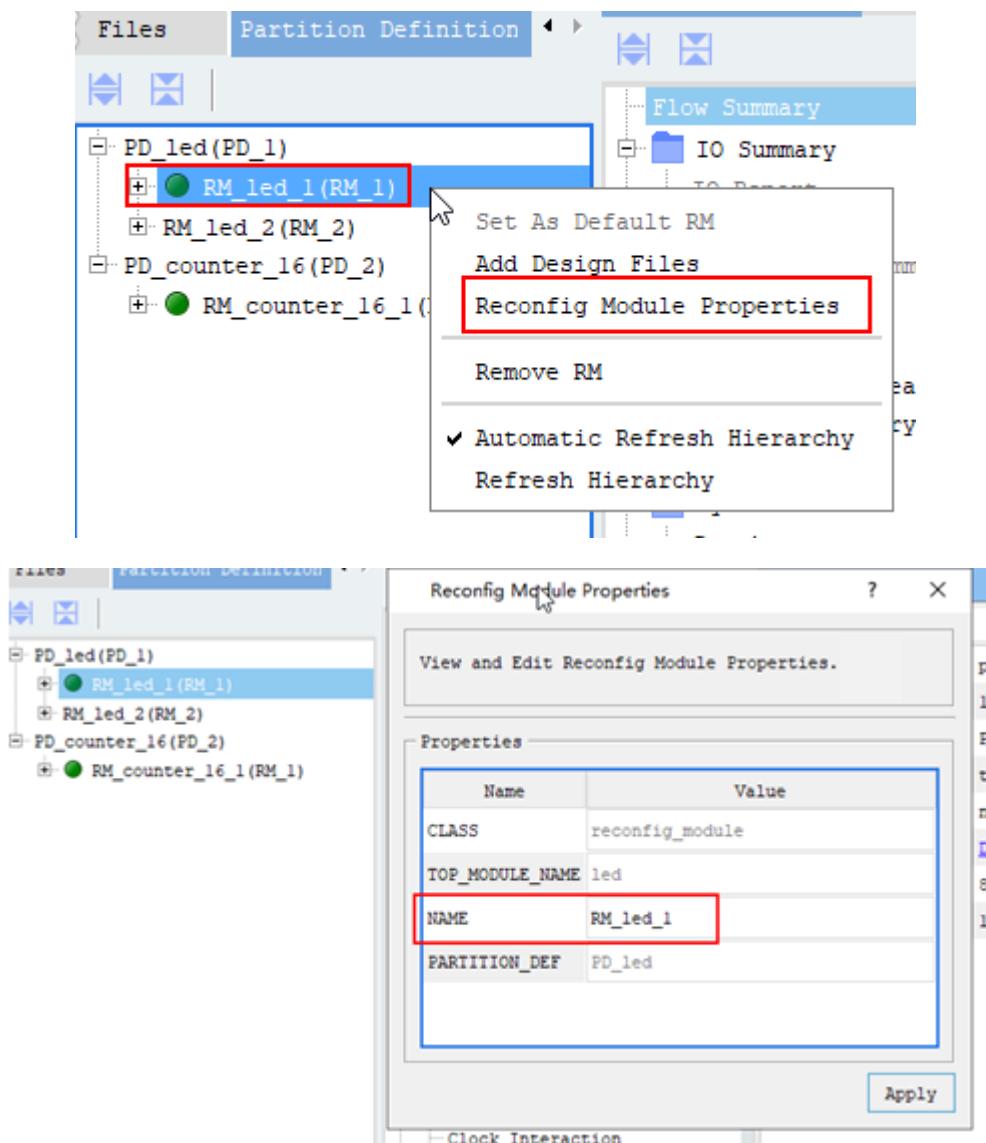


图 7-25 查看和更改 RM

## 7.5 删 除 Reconfigure Module

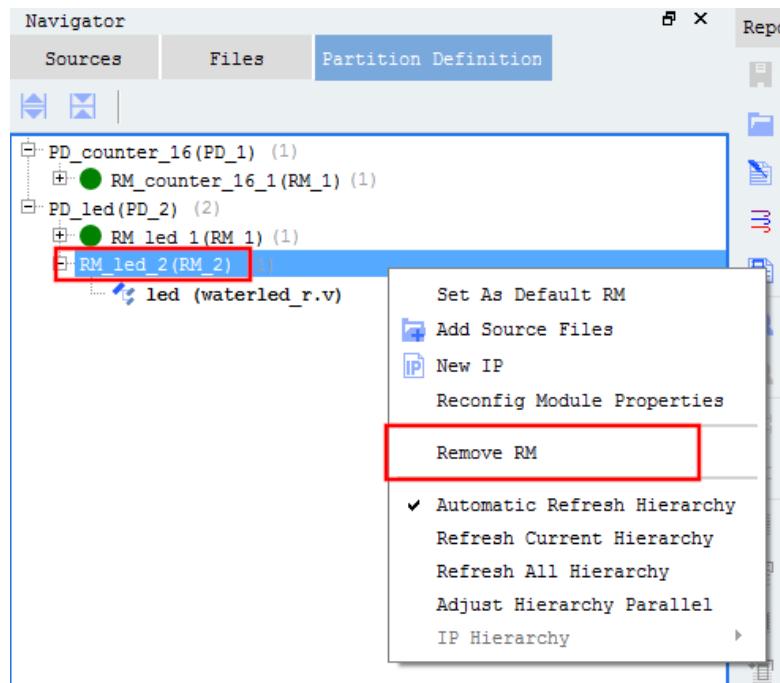


图 7-26 删除 RM

删除前界面如下图所示

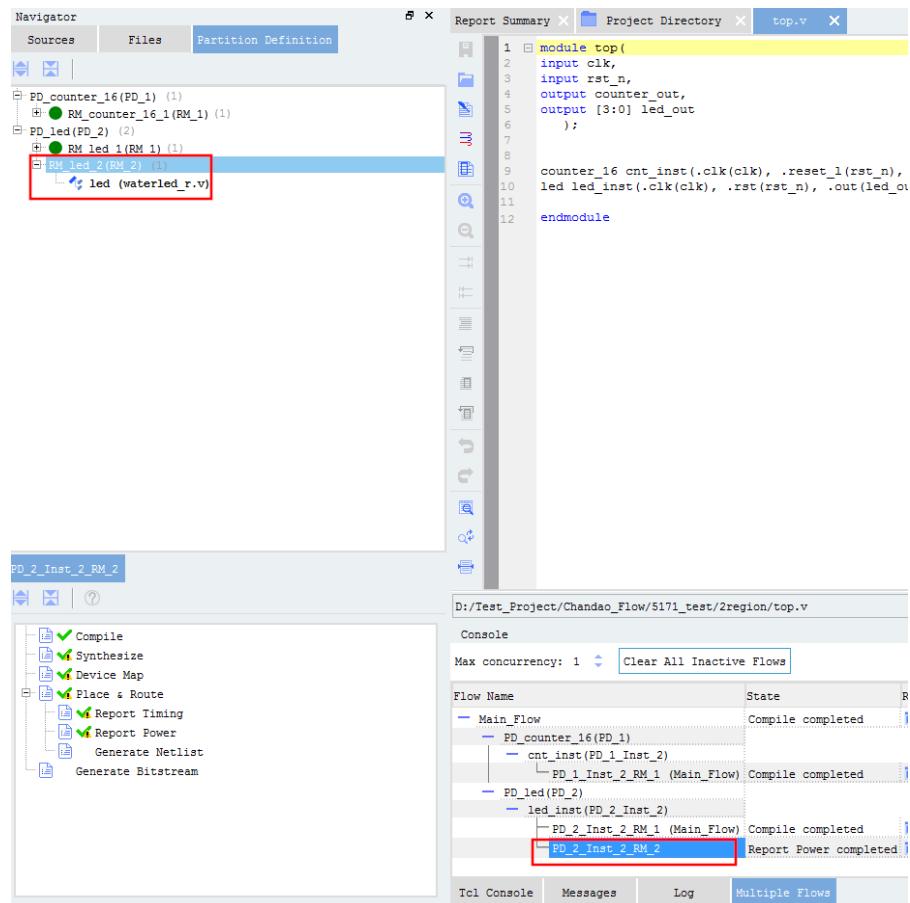


图 7-27 删除 RM 前

删除后界面如下图所示

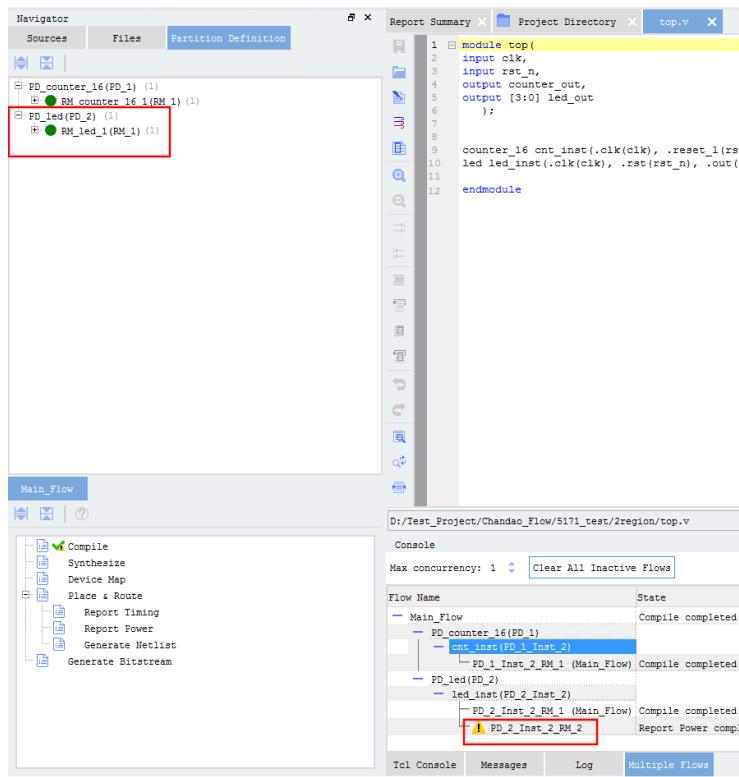


图 7-28 删除 RM 后

这里的“!”图标表示 RM\_Flow 对应的 PD\_Inst 或 RM 或 PD 不存在。

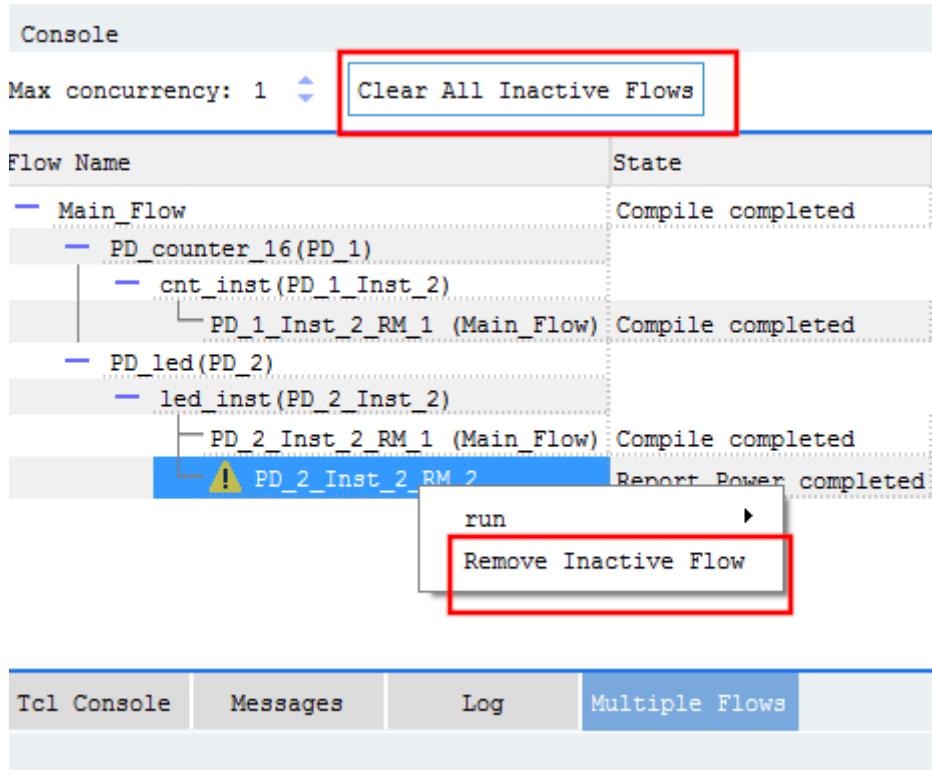


图 7-29 删除 IF

可通过以上 2 个操作接口删除 Inactive Flow。

## 7.6 RM 增删 Design 文件

### 7.6.1 RM 添加 Design 文件

在 RM 上点击右键添加 Design 文件

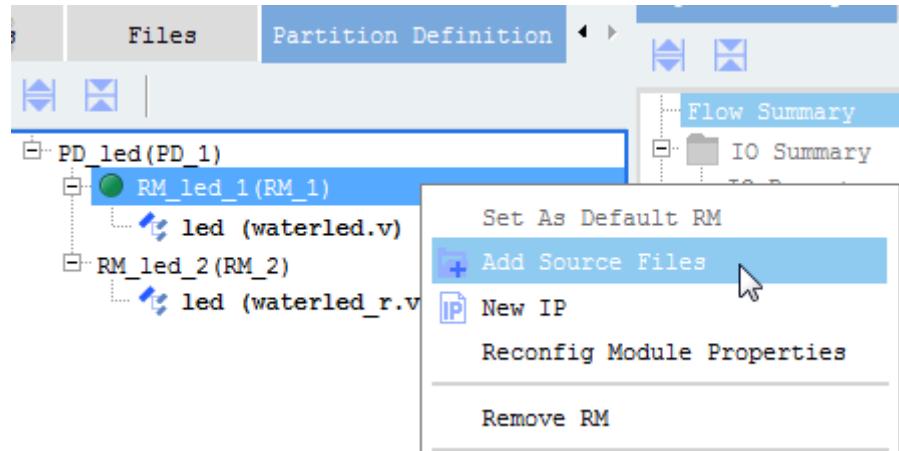


图 7-30 RM 添加 Design

在 RM 的 Module Hierarchy 节点上点击右键添加 Design 文件

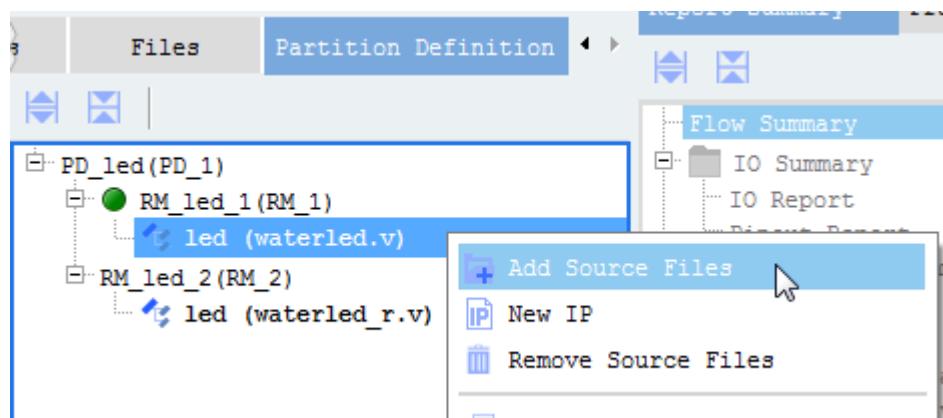


图 7-31 RM 的 module heir 节点添加 Design

### 7.6.2 RM 的 Top Module

目前,同一 PD 的每个 RM 的 top module,接口必须跟该 PD 的默认 RM(即该 PD 的 RM1)的接口一致,包括 module 的端口数量、类型、顺序和名称。

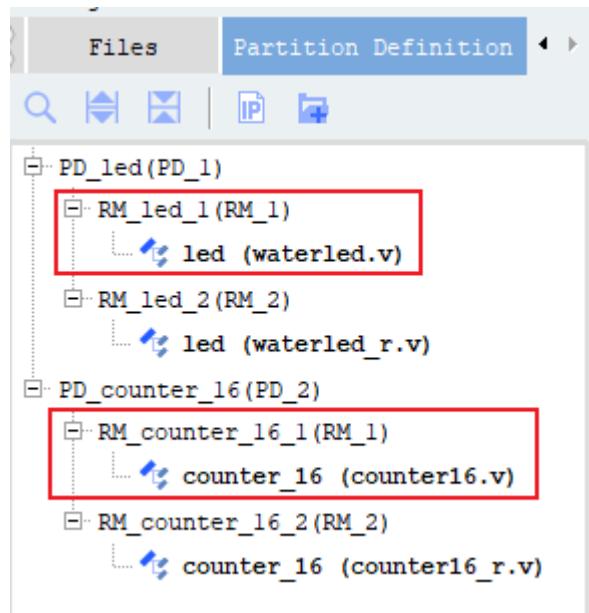


图 7-32 RM 的 top module

例如，RM\_led\_1（PD\_1\_RM\_1）的 top module 的接口如下

`led(input clk, input rst, output out);`

则 RM\_led\_2（PD\_1\_RM\_2）的 top module 的接口也要跟上述接口完全一致。

例如，RM\_counter\_16\_1（PD\_2\_RM\_1）的 top module 的接口如下

`module counter_16(input clk, input reset, output reg[3:0] counter);`

则 RM\_counter\_16\_2（PD\_2\_RM\_2）的 top module 的接口也要跟上述接口完全一致。

### 7.6.3 RM 删除 Design 文件

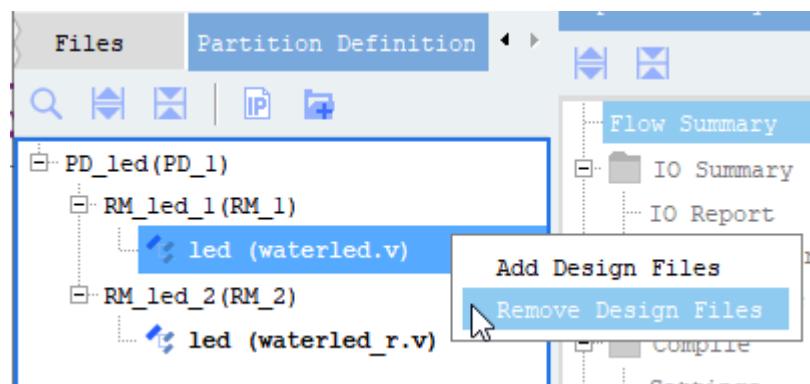


图 7-33 RM 删除 Design

## 7.7 添加/删除 PD\_Instance

### 7.7.1 添加 PD\_Instance

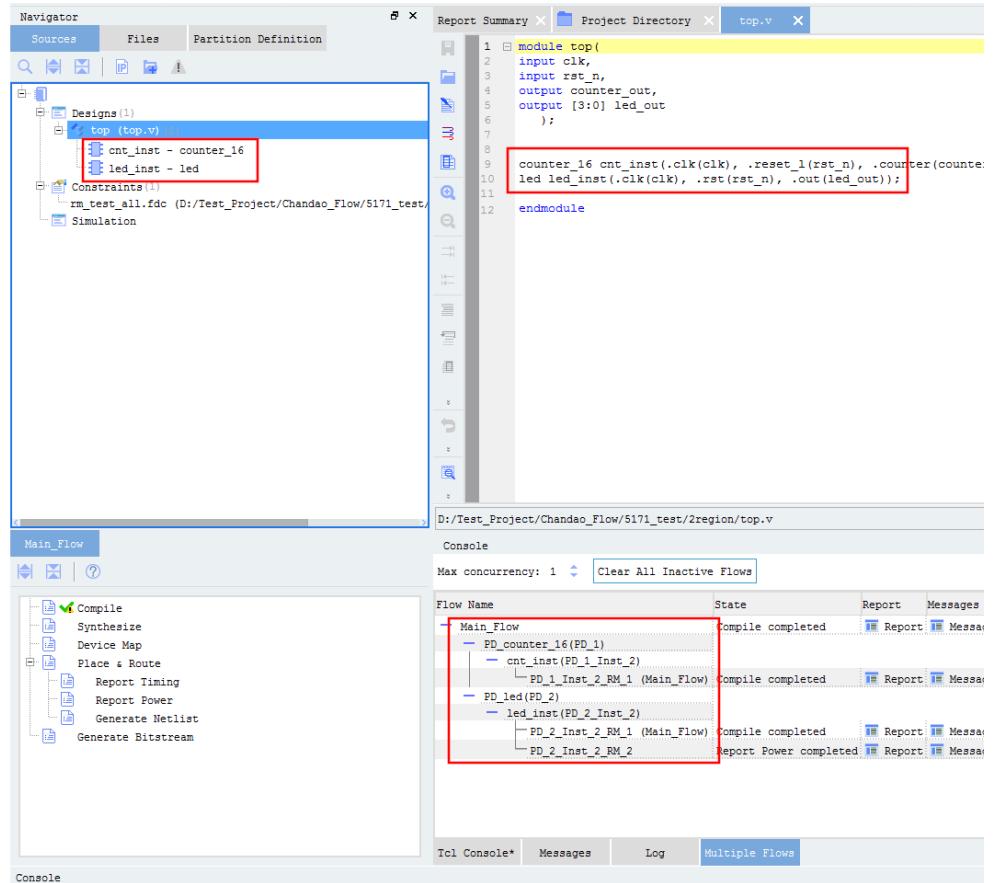


图 7-34 原 design 和界面

原 design 和界面如上图所示，修改 design 文件，增加一个 pd\_inst 之后，design 和界面如下图所示

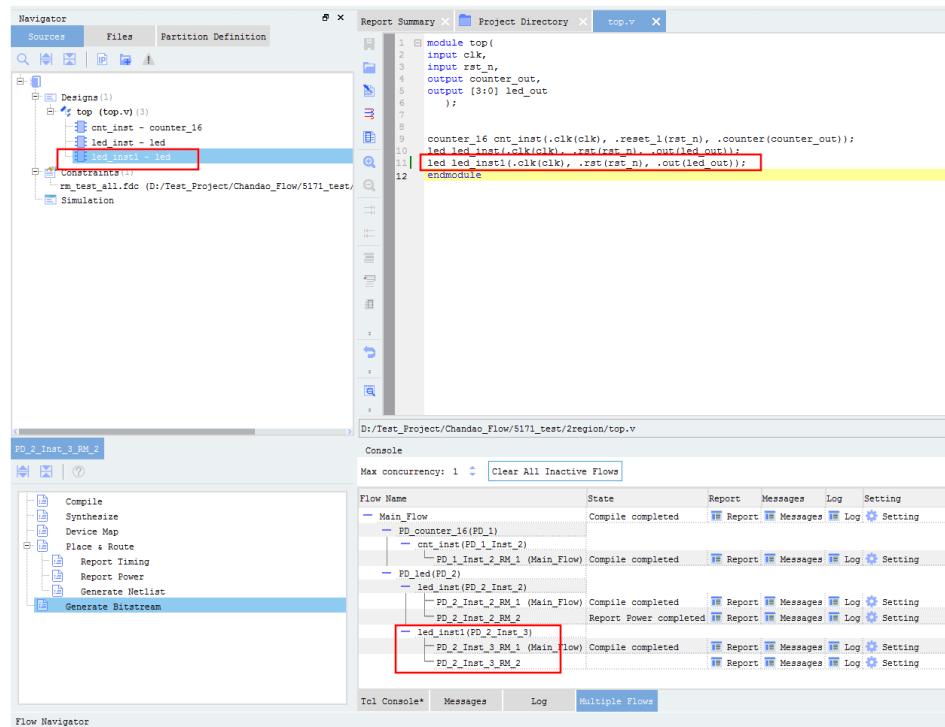


图 7-35 添加后

### 7.7.2 删 除 PD\_Instance

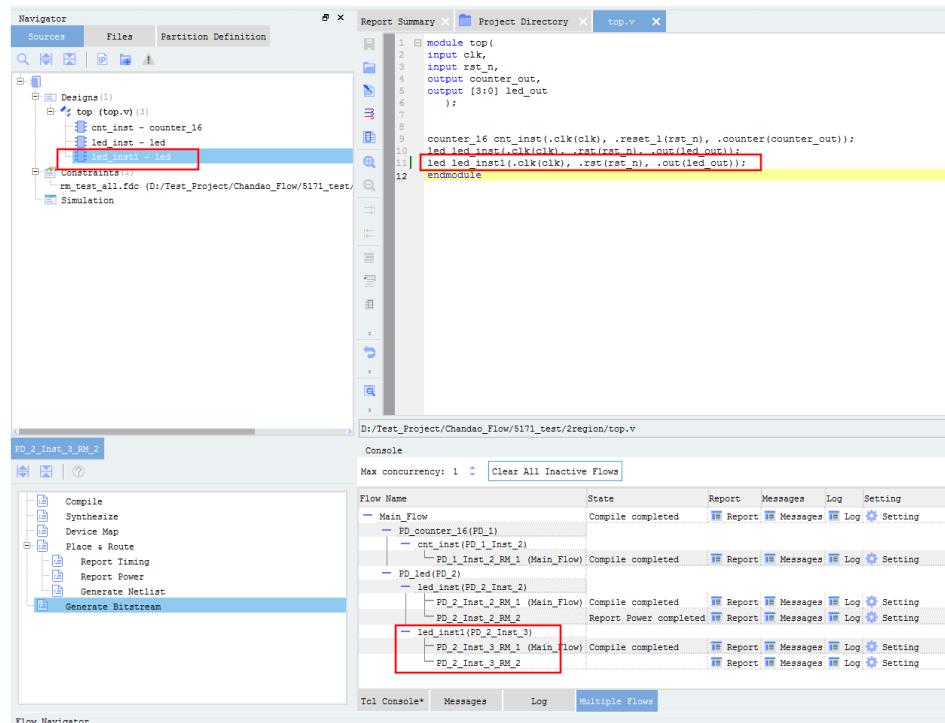


图 7-36 原 design 和界面

原 design 和界面如上图所示，修改 design 文件，删除一个 pd\_inst 之后，design 和界面如下图所示

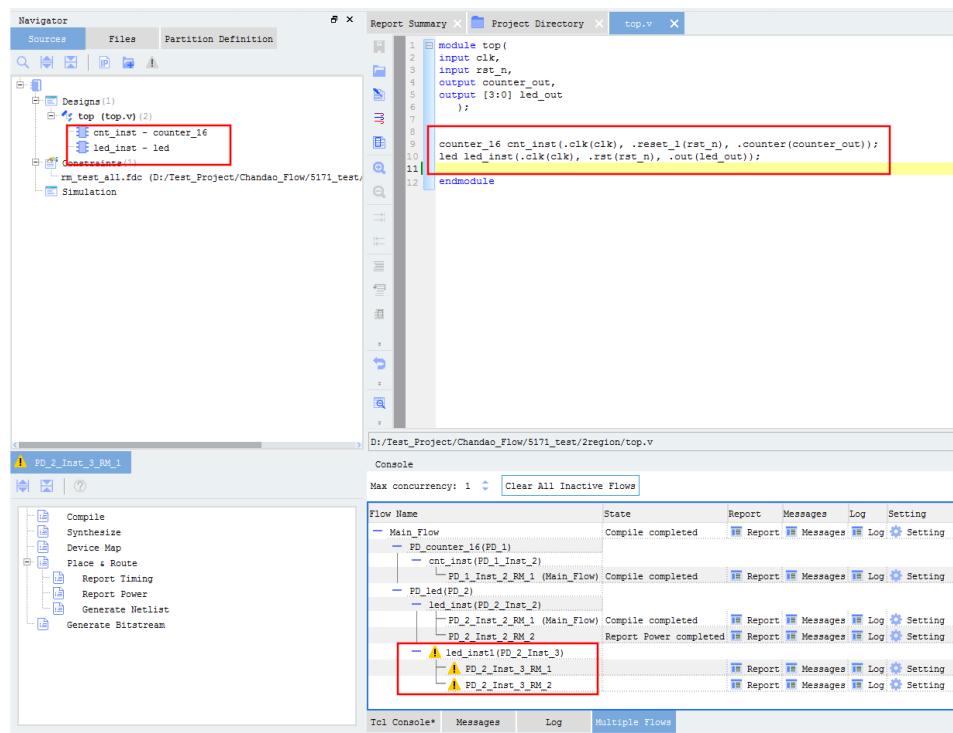


图 7-37 删除后

这里的“!”图标表示 RM\_Flow 对应的 PD\_Inst 或 RM 或 PD 不存在。

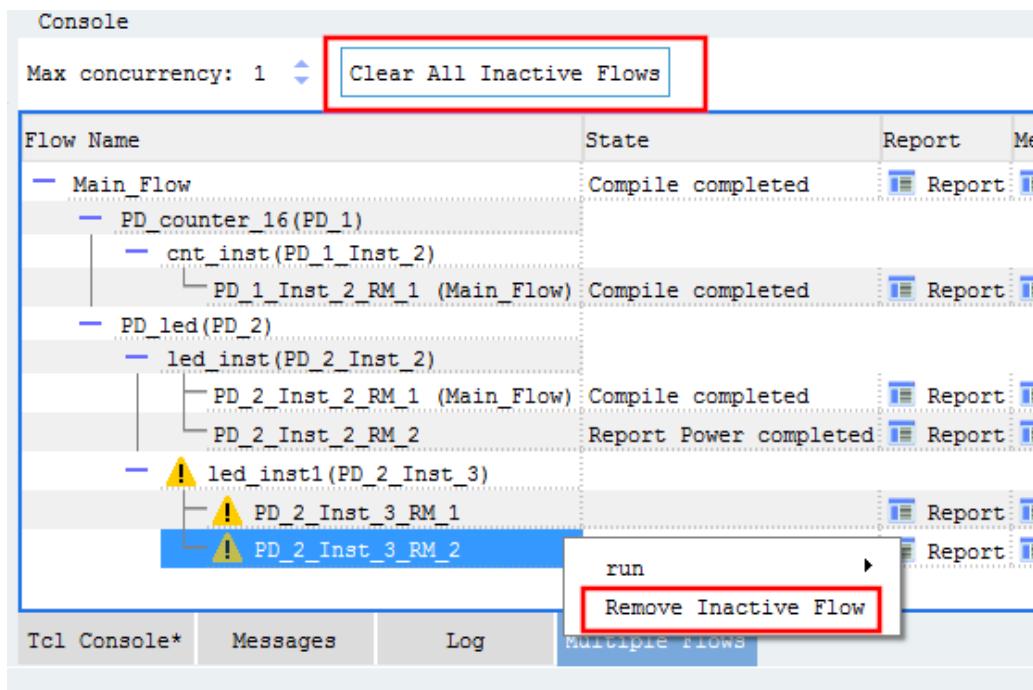


图 7-38 删除 Inactive Flow

可通过以上 2 个操作接口删除 Inactive Flow。

## 7.8 约束可重配区域

目前对动态区域进行物理约束，可在 UCE 中按照如下操作完成：

(1) 在 UCE 中绘制一个区域(region1)，绘制方法与绘制区域约束的区域一致。具体如下：

点击 Region Mode 按钮，进去区域模式。在 Floorplan 上右键点击菜单 Draw Region，此时鼠标变为十字型形状，按住鼠标，在 FloorPlan 上框出一片区域。鼠标释放后，弹窗”new region”中点击按钮 OK，即创建区域 region1 成功。

(2) 将 active 的 RM Instance (rm\_inst) 约束在步骤 1 绘制的区域(region1)中，方法与将 Instance 约束到区域中一致。具体如下：

点击 UCE 的 Design Browser 按钮(形状为字母 N)，在 Design Browser 中选中 rm\_inst，将该 Instance 拖拽到 region1 中即可。

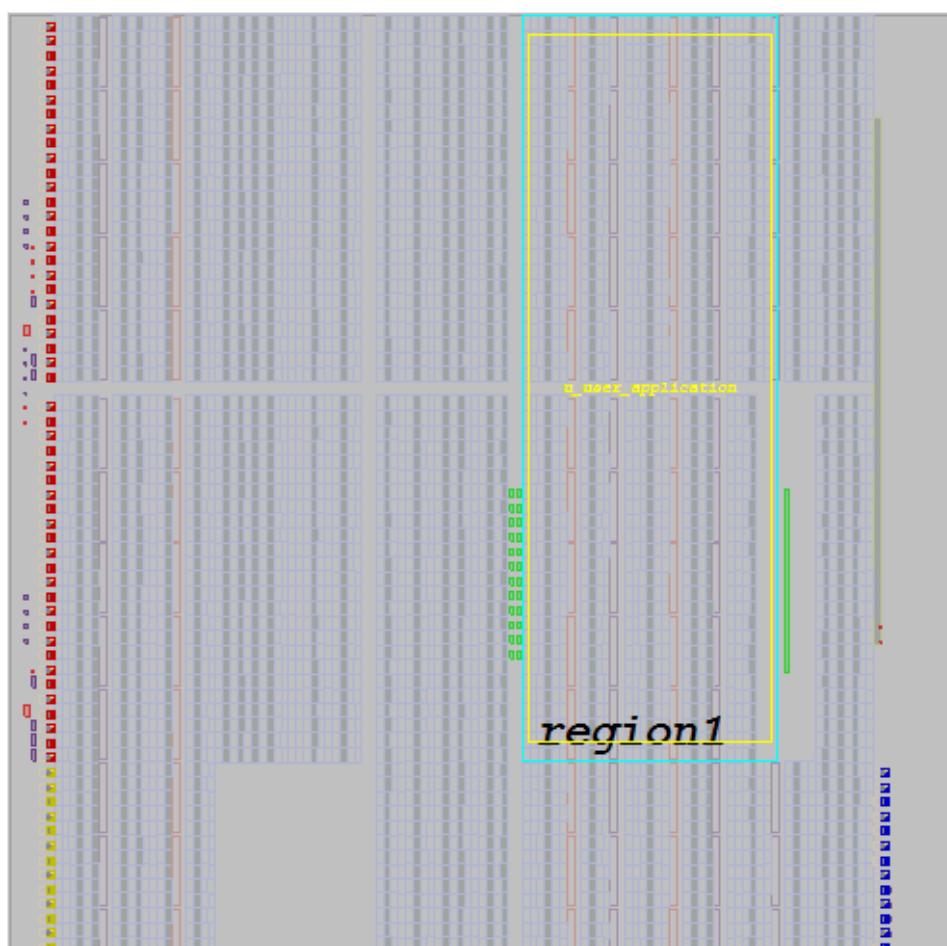


图 7-39 region1

(3) 为 region1 添加 PAP\_EXCLUSIVE\_REGION 属性，说明 region1 是一个动态区，软件会按照一定的规则对用户绘制的动态区进行一定的拓展。

具体操作为：在 Floorplan 上点击 Region，右键菜单中 Set Exclusive Region 和 Unset Exclusive Region，分别对应将 region 设置属性 Exclusive region 和取消属性 Exclusive region。

设置该属性以后，Region 样式为密集实粒阴影类型，范围内由白色粒子高度密集覆盖。下方黑体显示该 Region 的名称，Region 前有标示“RM:”表明该属性用于局部动态重配。

在 fdc 中添加 PAP\_EXCLUSIVE\_REGION 属性，需要先声明 Region 后对 Region 设置该属性，即：

```
define_global_attribute {PAP_REGION} {region1(21,24,42,43);}
```

```
define_global_attribute {PAP_EXCLUSIVE_REGION} {region1}
```

(4) 芯片上并非所有资源都可以被重配，所以并非所有的资源都可以被划分到可重配区域内。芯片的最左侧两列和最右侧两列 SRB 以及这些 SRB 所在 tile 内的全部资源、全局时钟资源以及和这些资源使用同一帧位流的 SRB tile 内的所有资源不能划分为可重配区域。避开上述资源，可以得到在配置约束时，UCE 中允许划分到可重配区域的资源的 X 方向的坐标范围。PG2L100H 的范围是[6, 63], [68, 111], PG2T390H 的范围是[6, 111], [116, 211]。

另外，CCS，ADC、PCIE 以及上述三种资源附近的 SRB 所在 tile 内的资源不能被划分到可重配区域内，即 PG2L100H 的[26, 47, 50, 149], [96, 114, 150, 199]和 PG2T390H 的[26, 47, 100, 199], [192, 213, 150, 249]不能划分为可重配区域。上文中坐标格式为[x1, x2, y1, y2]，且上述所有坐标均为 floor plan 坐标，所有区间均为闭区间。

约束会进行自适配。若包含非法区域软件会按 Y 轴进行自适配切割，可重配范围可能有所缩小。若无法切割成一个区域会给出提示。同时会适配到时钟域的上下边界，可重配范围可能有所扩大。

## 7.9 Run Flows

### 7.9.1 流程与 PD、PD Instance 及 RM 的对应关系

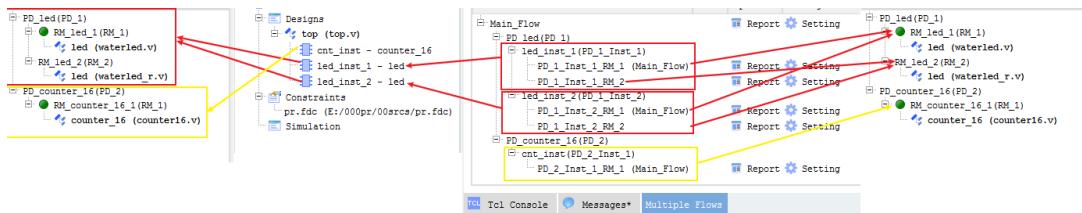


图 7-40 流程与 PD、PD Instance 及 RM 的对应关系

上图中，PD\_led (PD\_1) 有 2 个 Instance: led\_inst\_1 和 led\_inst\_2; PD\_counter\_16 (PD\_2) 有 1 个 Instance: cnt\_inst。

PD\_led (PD\_1) 管理 2 个 RM: RM\_led\_1 (PD\_1\_RM\_1) 和 RM\_led\_2 (PD\_2\_RM\_2)；  
PD\_counter (PD\_2) 管理 1 个 RM: RM\_counter\_16\_1 (PD\_2\_RM\_1)。

“Multiple Flows”部件中，PD\_led(PD\_1)节点下有led\_inst\_1和led\_inst\_2两个节点，意思是PD\_led (PD\_1)有两个Instance: led\_inst\_1 和 led\_inst\_2。led\_inst\_1 节点下有2个Flow 节点: PD\_1\_Inst\_1\_RM\_1 和 PD\_1\_Inst\_1\_RM\_2; PD\_1\_Inst\_1\_RM\_1 的意思是 led\_inst\_1 选用 PD\_1\_RM\_1 运行流程,PD\_1\_Inst\_1\_RM\_2 的意思是 led\_inst\_2 选用 PD\_1\_RM2 运行流程。

### 7.9.2 切换流程树

在“Multiple Flows”部件点击相应的流程名，左侧会切换到相应的流程树，可通过流程树“Run”、“Rerun”和“Rerun All”。

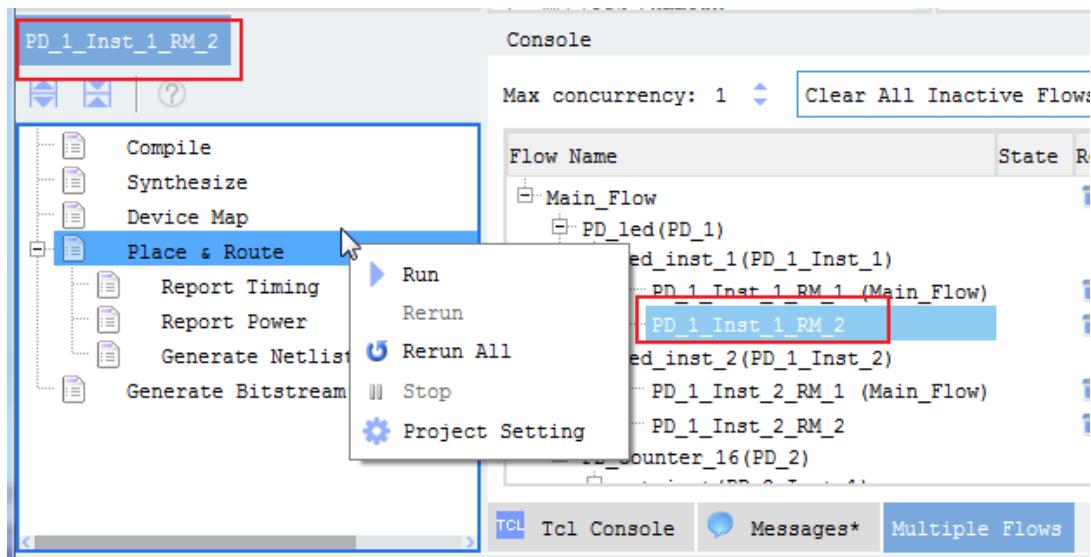


图 7-41 切换流程树

### 7.9.3 流程依赖

RM 流程的“Synthesize”依赖于主流程的“Report Timing”，即运行 RM 流程的“Synthesize”之前会自动将主流程运行至“Report Timing”。

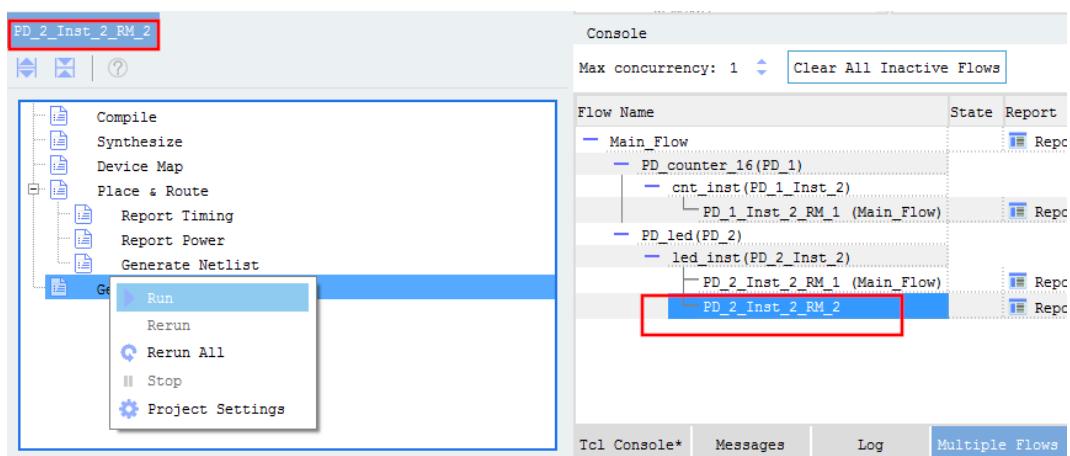


图 7-42 流程依赖

例如运行上图中“PD\_1\_Inst\_1\_RM\_2”流程的“Generate Bitstream”，在运行 Synthesize

之前，会自动运行“Main Flow”至“Report Timing”。

#### 7.9.4 流程关系

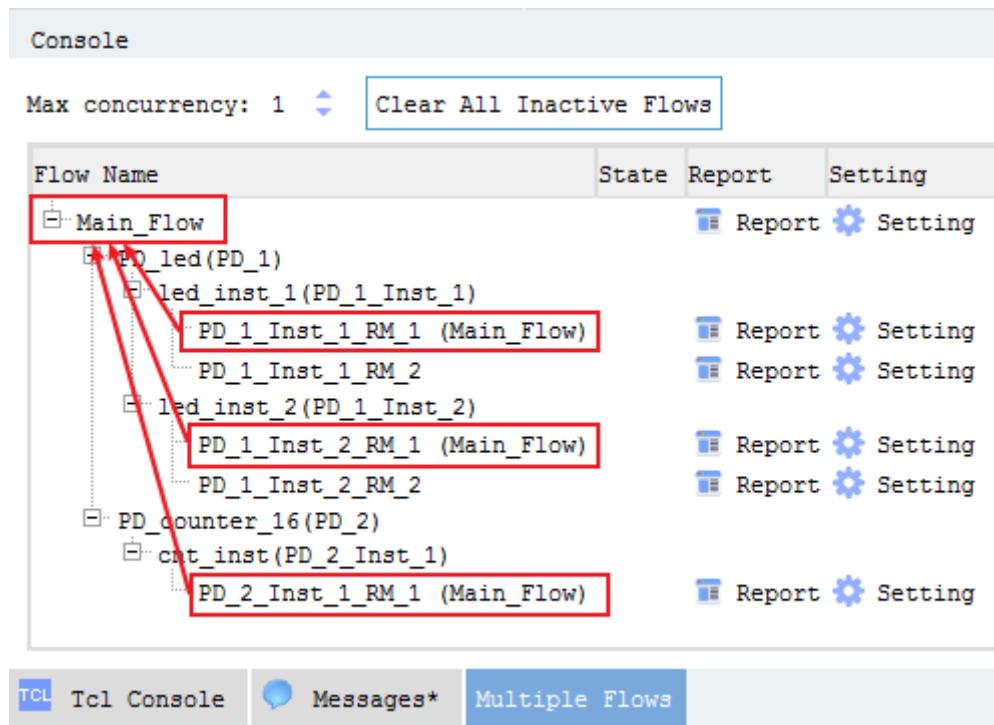


图 7-43 流程关系

如上图所示，PD\_1\_Inst\_1\_RM\_1、PD\_1\_Inst\_2\_RM\_1 和 PD\_2\_Inst\_1\_RM\_1 这 3 个流程，实质上都是 Main Flow，运行这些流程，就是运行主流程。因为运行主流程时，会将 PD 的 RM\_1 配置到 PD 的每个 Instance 去运行。

#### 7.9.5 流程并行运行

如果流程间没有依赖关系或者满足流程依赖条件后，流程间可以并行运行。

所有的重配流程之间都没有依赖关系，所以重配流程间可以并行运行。

重配流程的 Synthesize 依赖于主流程的 Report Timing，所以主流程和重配流程的 Compile 可以并行运行，当重配流程要运行 Synthesize 时，重配流程会等待主流程运行完 Report Timing，这时重配流程的 Synthesize 和之后的步骤，就可以跟主流程的 Report Timing 之后的步骤并行运行（重配流程的 Generate Bitstream 依赖于主流程的 Generate Bitstream，所以这两个步骤不能并行运行）。

这就意味着，当有流程运行的时候，依然可以启动其他流程进行运行。如下图所示：

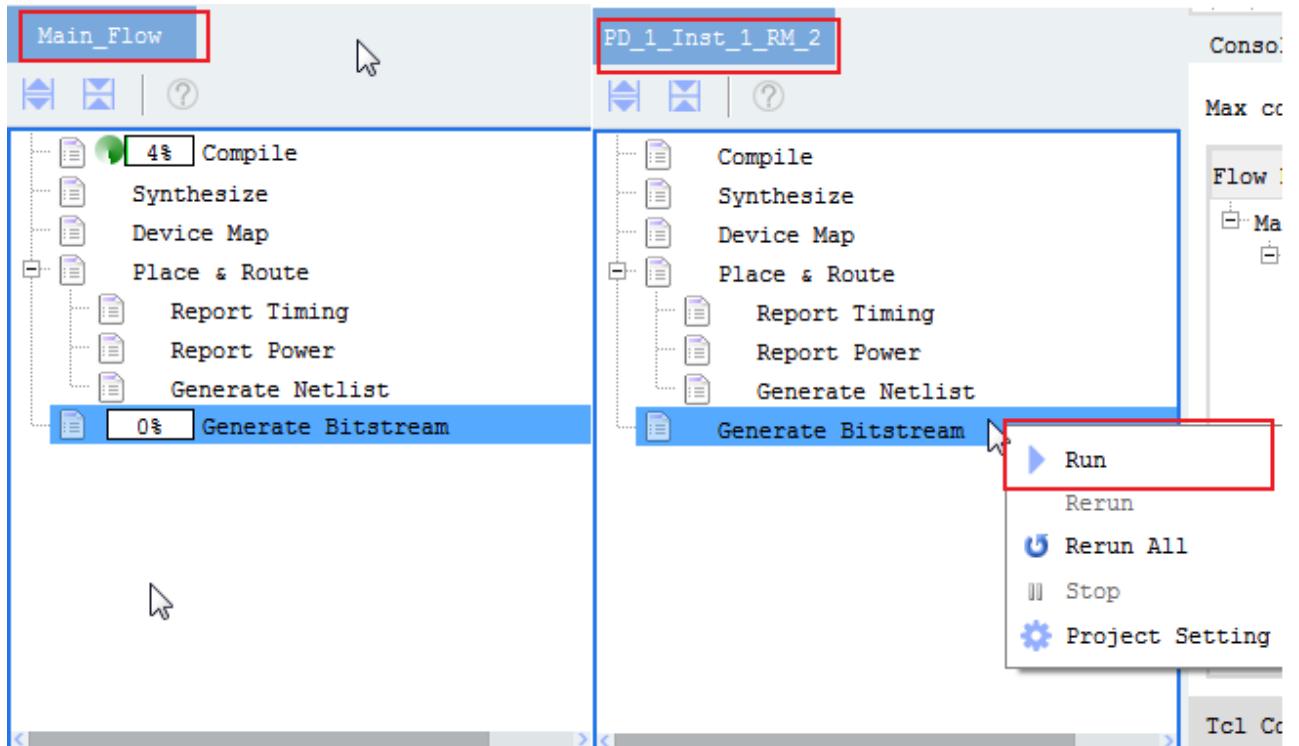


图 7-44 运行多流程

或者同时运行多个流程，如下图所示：

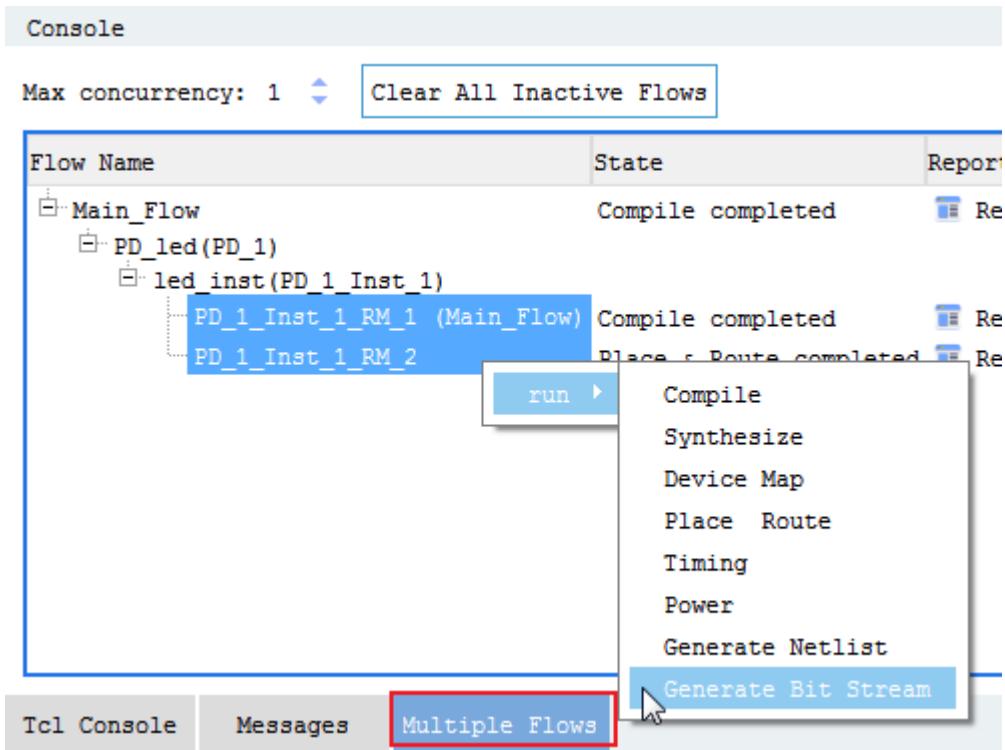


图 7-45 同时启动多流程

## 7.10 组件相关

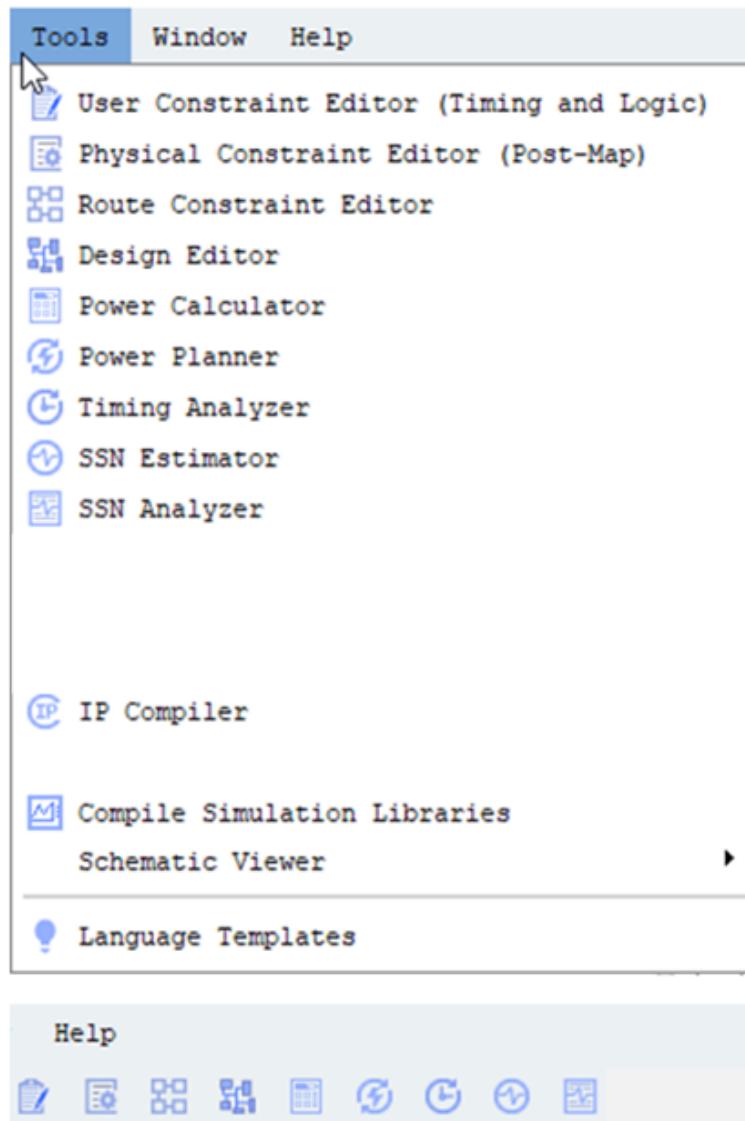


图 7-46 主流程支持组件

只有主流程支持以上组件。

## 7.11 Project Setting

### 7.11.1 操作演示

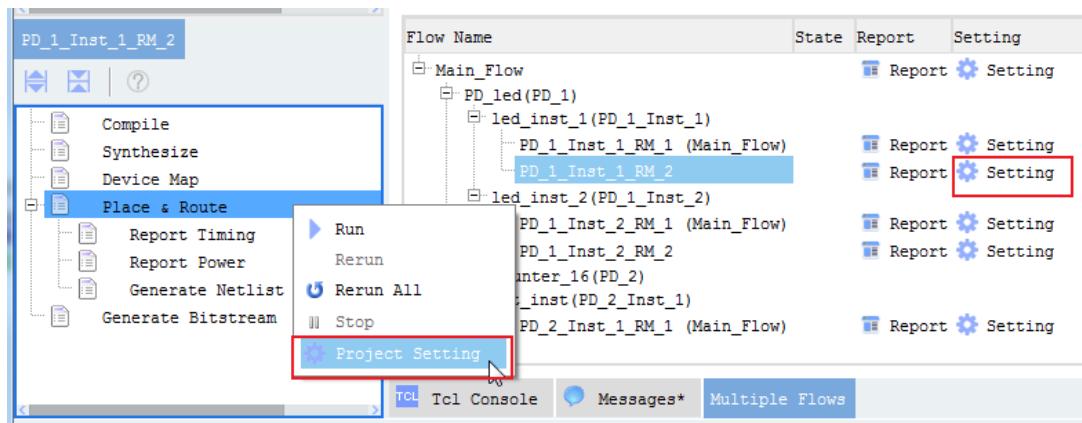


图 7-47 打开 project setting

- (1) 在流程树上点击鼠标右键，选择“Project Setting”。
- (2) 在“Multiple Flows”部件点击相应流程的“Setting”按钮。

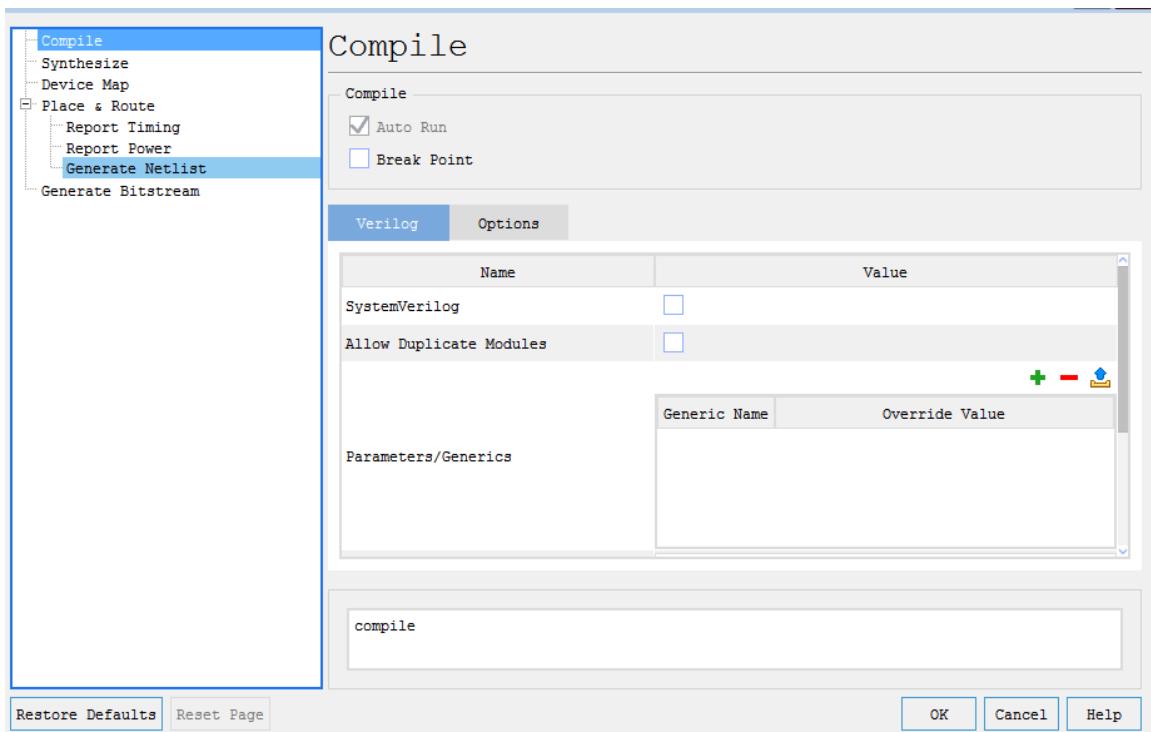


图 7-48project setting 界面

### 7.11.2 相关说明

- (1) 每个流程都有相应的“Project Setting”。
- (2) 更改任一流程的器件或综合工具设置，会同步设置到所有流程。
- (3) PD\_1\_Inst\_1\_RM\_1、PD\_1\_Inst\_1\_RM\_2 和 PD\_2\_Inst\_1\_RM\_1 这 3 个流程，实质上是 Main Flow，所以这 3 个流程的“Project Setting”就是 Main Flow 的“Project Setting”。

(3) RM 流程目前没有“Simulation”设置，Main Flow 有。

(4) RM 流程 Project Setting 下无 Part 页面

## 7.12 Report Summary

### 7.12.1 操作演示

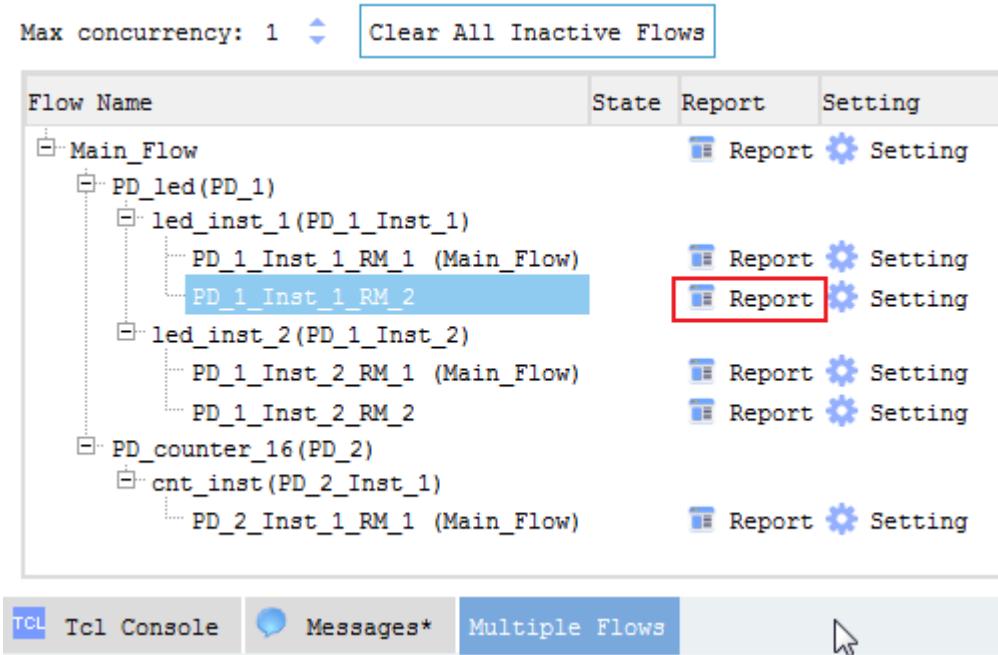


图 7-49 点击 report

点击“Multiple Flows”相应流程的“Report”，会显示相应流程的“Report Summary”，如下图所示。

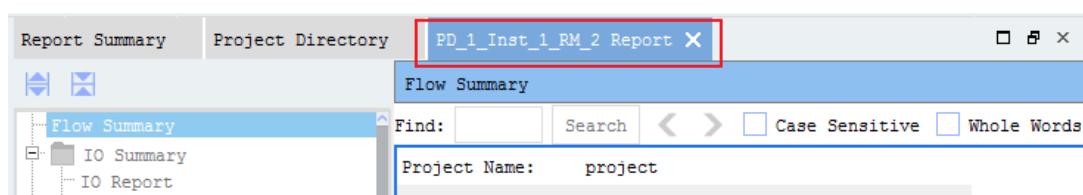


图 7-50 report summary

## 7.13 Messages

### 7.13.1 操作演示

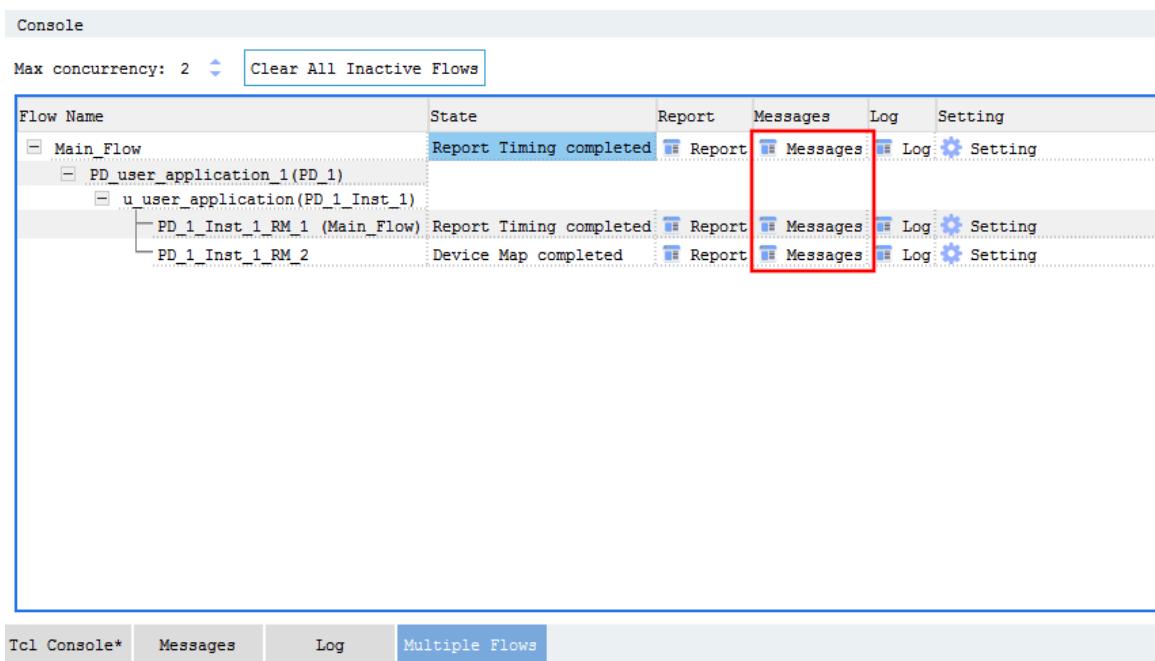


图 7-51 点击 Messages

点击“Multiple Flows”相应流程的“Messages”，会显示相应流程的“Messages”，如下图所示，显示 RM2 流程 Messages。

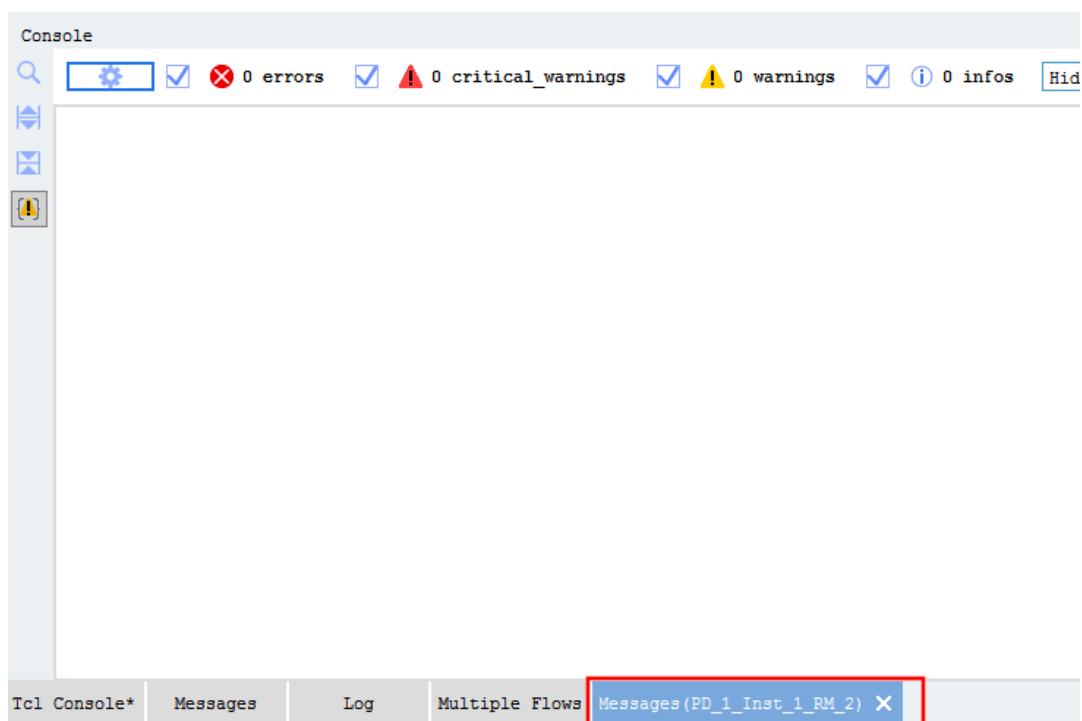


图 7-52 显示 RM2 流程 Messages

## 7.14 Log

### 7.14.1 操作演示

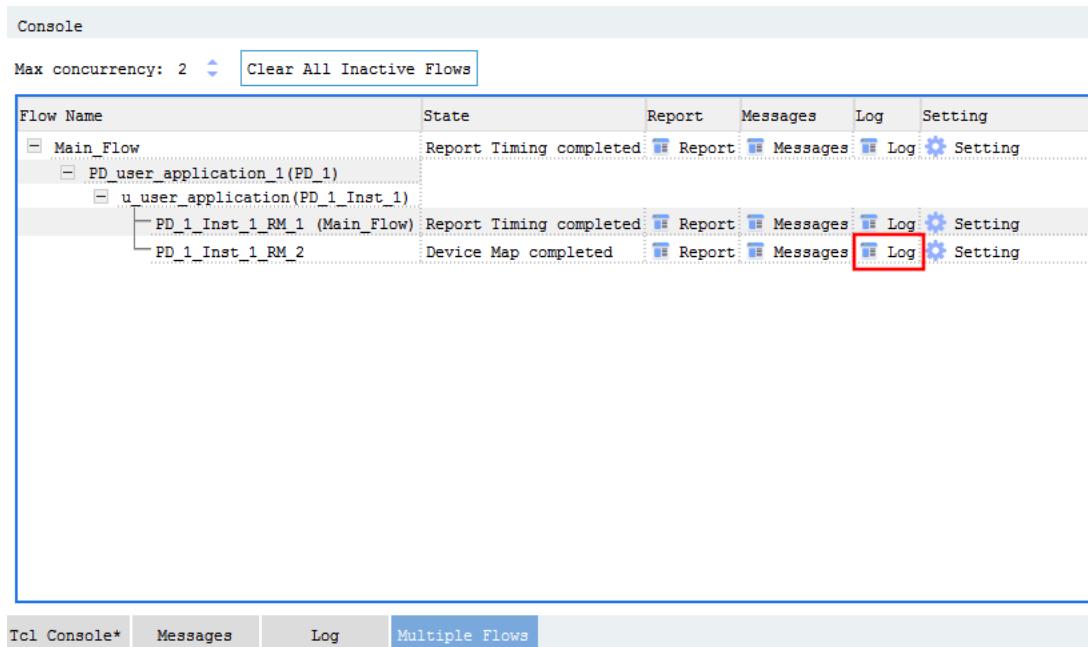


图 7-53 点击 Log

点击“Multiple Flows”相应流程的“Log”，会显示相应流程的“Log”，如下图所示，显示 RM2 流程 Log。

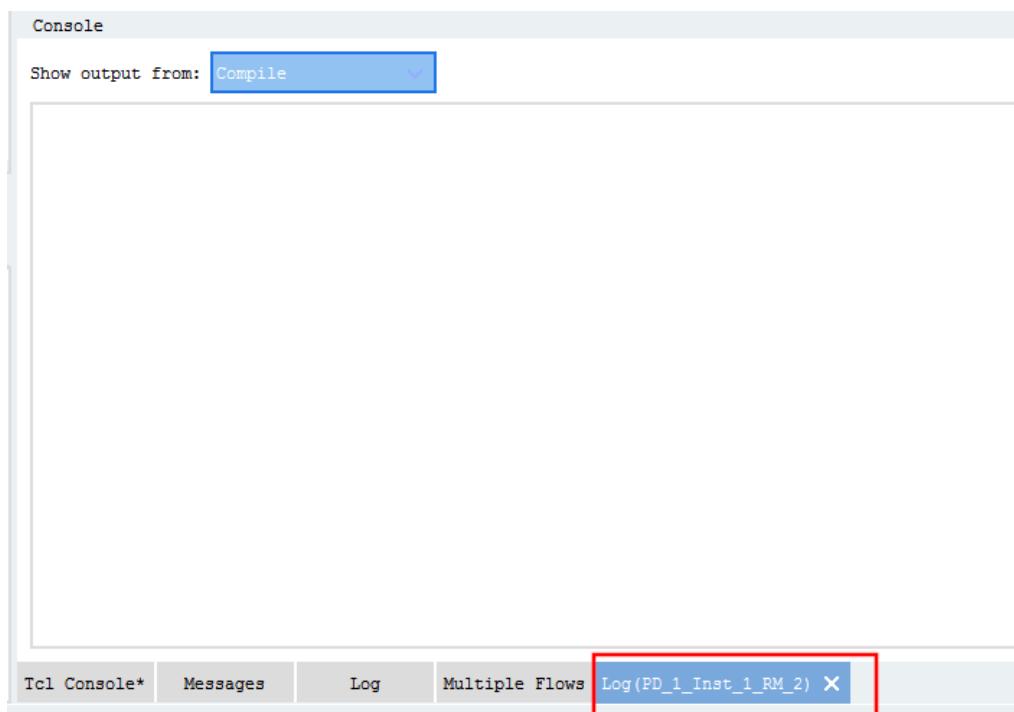


图 7-54 显示 RM2 流程 Log

## 7.15 重配流程的工程文件

重配流程的工程文件会保存在工程主目录的 rm\_projects 文件夹下，这些工程文件是为了记录重配流程的数据，当打开工程时，可以读取重配流程的数据。

重配工程文件时局部重配工程的一部分；打开局部动态重配模式的工程请打开工程主目录下的主工程文件，不能单独打开 rm\_projects 文件夹下的重配工程文件。

## 7.16 位流生成与编程

### 7.16.1 位流文件说明

在 main flow 和 rm flow 中都会生成三个位流文件。

- (1) 完整位流文件：具体以 design name 命名的文件，包含所有区域（静态与动态）的数据信息，可单独进行编程，可实际观察相应的逻辑功能。
- (2) 动态位流文件：文件名为 rm.sbit，包含动态逻辑对应的功能且对应动态区划分的逻辑区域
- (3) 静态位流文件：文件名为 static.sbit，包含除动态区数据外其他所有数据。

编程下载时，软件 configuration 会自动识别 rm.sbit 采用动态重配的方式进行编程。

## 7.17 支持器件

目前 PDS 的局部动态重配工程支持 PG2L100H 与 PG2T390H 两款器件，即仅可以选择上述器件转化为动态重配模式或者新建动态重配工程

## 8 软件原则

### 8.1 文件和文件夹命名规则

PDS 套件文件夹命名规则：只允许字母数字下划线（\_）杠（-）点（.）@ 和空格（ ），但空格不能出现在路径名首尾。

文件命名规则：只允许字母数字下划线（\_）杠（-）点（.）。

特别的，IPC 文件命名规则：只允许字母数字下划线（\_）。

在 PDS 主界面以及各个组件的对话框，在保存文件，或者填写路径名时，都会按照此规则进行判断，不符合的会进行提示。

### 8.2 PDS 套件中打开工具软件规则

PDS 打开各工具软件时，一个工具只打开一个实例；如打开 DE 时，只能打开一个 DE。

在有工具软件打开时，不能创建工程，关闭工程，另存为工程，关闭软件；需要先关闭相关软件后才能进行相应操作（在这种情况下进行相应操作会提示关闭工具软件后再操作）。

各工具软件的输入改变时，应提示用户相应改变，并让用户选择是否重新加载相应输入。

### 8.3 流程与工具软件的关系

流程在运行时，不允许打开工具软件（已经打开的工具软件不受影响）。

流程没有运行时，允许同时打开各工具软件，且保持其目前的流程依赖关系。如打开 PCE 时，dev\_map 及之前步骤要先运行完毕（如果打开 PCE 时没有运行至 map，流程会先运行至 map，再打开 PCE；打开 DE 时，PnR 及之前步骤要先运行完毕，再打开 DE）。

## 9 常用命令列表

### 9.1 add\_constraint 指定约束文件

选项	
-help	打印帮助消息
-copy	拷贝指定的文件到本工程
参数	
input_filename	指定一个输入文件名

表 9-1 add\_constraint

### 9.2 add\_design 指定设计文件

选项	
-help	打印帮助消息
-library	指定逻辑库的名字. 默认为 work
-define	定义编译宏
-copy	拷贝指定的文件到本工程
参数	
input_filename	指定一个输入文件名

表 9-2 add\_designt

### 9.3 add\_fic 指定约束文件

选项	
-help	打印帮助消息
-copy	拷贝指定的文件到本工程
参数	
input_filename	指定一个输入文件名

表 9-3 add\_fic

### 9.4 add\_simulation 指定仿真文件

选项	
-help	打印帮助消息
-library	指定逻辑库的名字. 默认为 work
-define	定义编译宏

-copy	拷贝指定的文件到本工程
<b>参数</b>	
input_filename	指定一个输入文件名

表 9-4 add\_simulation

## 9.5 add\_sources 指定文件夹

选项	
-help	打印帮助消息
-sub_directories	搜索子文件夹
-copy	拷贝指定的文件到本工程
-continue	当存在报错继续添加文件
-design	指定添加文件的类型为设计文件
-simulation	指定添加文件的类型为仿真文件
-constraint	指定添加文件的类型为约束文件
-fic	指定添加文件的类型为插核文件
-ip	指定添加文件的类型为 IP 文件
-library	指定逻辑库的名字., 默认为 work
-define	定义编译宏
<b>参数</b>	
input_dir	指定一个包含输入文件的文件夹, 注意:在 tcl 标准语法中, “\”是转义字符, 您可能需要将 windows 文件夹路径中的 “\” 更改为 “/” 后再运行

表 9-5 add\_sources

## 9.6 clean 清除 action 状态

选项	
-help	打印帮助消息
-action	指定一个 action 名字
-translate	对 report timing action 生效
-configuration	对 report timing action 生效

表 9-6 clean

## 9.7 set\_arch 指定器件信息

选项	
-help	打印帮助消息

-family	指定一个家族
-device	指定一个器件
-package	指定一个封装
-speedgrade	指定一个速度等级

表 9-7 set\_arch

## 9.8 compile 编译用户设计

选项	
-help	打印帮助消息
-top_module	设置 top module 的名字
-include_path	设置工程的 include 路径

表 9-8 compile

## 9.9 dev\_map 映射设计到架构

选项	
-help	打印帮助消息
-packing_io	设置 top module 的名字
-detail	设置工程的 include 路径
-generate_inferred_clocks	勾选后会对符合条件的 design 插入时钟
-fanout	在报告中只打印 fanout 大于该数值的 Net 或 port 对象
-fanout_number	在报告中打印的 fanout 数量大于 Min fanout 数值的 Net 数量的上限
-override_pcf	重新运行时覆盖 pcf 文件
-enable_timing_commands	将 OEM 注释掉的约束重新打开
-devmap_control_set_num	用于控制同一种 control set 下，最小触发器的数目。但触发器的数目小于门限时，则将门控信号作为数据通路用 LUT 来实现。Control set 是指触发器的时钟、使能信号，RST 信号。优化该控制信号可以增加资源使用率
-devmap_const_prop	主要用于优化输入的常量逻辑。常量逻辑主要是指 VCC、GND 逻辑。该优化操作主要通过电路固有的特性，将原来的 VCC、GND 断开或优化到最优状态。
-devmap_remove_dups	该功能主要作用是将部分重复逻辑合并，属于面积优化选项。但是该操作同样会影响到综合阶段的复制工作，使时序性能下降。

表 9-9 dev\_map

## 9.10 gen\_bit\_stream 生成配置位流

选项	
-help	打印帮助消息
-create_bit_file	选择是否生成比特文件
-create_bin_file	选择是否生成无信息头的比特文件
-compress_bitstream	选择是否压缩位流文件
-optimize_compress	选择是否优化压缩位流
-write_drm_data	选择是否写入 DRM 数据
-unused_io_status	选择未使用 IO 的状态
-usercode	给用户标识寄存器赋值, 输入 8 位十六进制的 user code, 该值会被写入到位流文件中
-master_configuration_clock_frequency	主模式时钟 clk 频率选择
-system_clock_frequency	配置存储器时钟 mclk 频率选择
-enable_osc_shut_off	控制是否允许用户逻辑关断 OSC
-enable_watchdog_in_user_mode	选择是否在用户模式下使能看门狗
-enable_watchdog_in_configuration_mode	选择是否在配置模式下使能看门狗
-load_watchdog	设置看门狗的超时时间
-enable_version_select_pin	使能版本控制管脚
-enable_versionFallback	选择是否使能版本回退
-set_fallback_retry_times	设置版本回退尝试次数
-set_bitstream_retry_times	设置位流尝试次数
-rst_n_pin	选择 RST 管脚的上下拉还是悬空
-enable_external_master_configuration_clock	选择是否使能外部主配置时钟
-bpi_read_cycle	选择 BPI 读取周期
-bpi_page_size	选择 BPI 的页面大小
-bpi_mode	选择 BPI 的模式
-enable_done_synchronization	选择是否 Done 信号同步
-wake_up_time_slot_length	唤醒周期长度选择, 即 T1, T2, T3 的时间间隔
-done_time_slot	done 信号拉高时刻选择
-gwen_time_slot	gwen 信号拉高时刻选择
-grsn_time_slot	grsn 信号拉高时刻选择
-gouten_time_slot	gouten 信号拉高时刻选择
-enable_wait_dci_match	唤醒等待 DCI 匹配
-enable_wait_pll_lock	唤醒时等待 PLL
-wake_up_clock	选择唤醒时钟
-create_mask_file	选择是否生成 mask 文件, 用来决定回读文件与原始文件的比较规则。
-persist_slave_parallel_pins	控制用并行模式配置完成后, 外部并行端口是否保留用于回读

-master_configuration_clock_frequency_in_read_back	回读时钟选择
-disable_readback_crc	选择是否在回读时使能 CRC 验证
-enable_crc_clock_in_read_back	使能回读 CRC 时钟
-persist_ipal	控制配置完成后，内部并行端口是否保留用于回读
-bitstream_security	控制回读配置存储器
-encrypt_bitstream	选择是否加密位流文件
-select_key_type	配置是否使用内部 KEY
-starting_cbc_value	用户手动输入初始的 CBC 字符串，字符为 16 进制，输入长度为最多 32 个字符
-input_encryption_key_file	选择 nky 文件
-key_string	用户手动输入密钥字符串，字符为 16 进制字符，输入长度为最多 64 个字符
-sram_retention	开启 SRAM retention 测试

表 9-10 位流

## 9.11 gen\_netlist 生成网表

选项	
-help	打印帮助消息
-func_sim_only	是否使能仅生成 sim.v 文件
-sdf_annotation	勾选后在 sim.v 中调用 sdf 文件，不勾选则不调用
-process_corner	决定产生什么类型的 sdf 文件

表 9-11 生成网表

## 9.12 pnr 布局和布线设计

选项	
-help	打印帮助消息
-dc_duplication	勾选为自适应插入和复制逻辑单元，不勾选为不处理
-plc_decongestion	设置面积扩展系数
-gplace_seed	设置随机种子值
-gplace_times	设置运行多少个随机种子
-seed_step	设置种子的步长
-saved_outcome	设置保留最优的种子运行结果的数目
-parallel	设置同时执行种子的最大数目
-fast_router	勾选为使用全局布线，不勾选为不使用
-fast_router_mode	设置全局布线的模式
-share_router_control_signal	勾选为使用 Control Chain，不勾选为不使用

-use_global_ioclkgate	是否插入 IOCLKGATE
-mode	pnr 运行的模式
-optimize_multi_corner_timing	控制在布局布线阶段是否要在所有的模式中都对 design 进行优化来满足时序
-place_only	勾选表示只运行布局，不运行布线
-fix_holdViolation	勾选为修复 hold 违例，不勾选则为不修复
-fix_holdViolation_iterations	最大修复 hold 违例次数
-fix_holdViolation_threshold	使用该选项对 slack 小于该值的时序路径进行 Hold 时序优化
-fix_holdViolation_in_route	勾选为在布线中修复 hold 违例，不勾选则为不修复
-fix_holdViolation_threshold_for_clock_path_cross_srb	使用该选项对 slack 小于该值的时序路径进行 Hold 时序优化，仅当对应时钟经过 SRB 时生效
-fix_holdViolation_threshold_for_clk_tree	使用该选项对 slack 小于该值的时序路径进行 Hold 时序优化，仅当对应时钟不经过 SRB 时生效
-input_place_db_file	
-refinement_cluster	使能详细布局过程中打包开关
-refinement_iteration	详细布局收敛属性
-plc_initial_method	全局布局初始化起始位置
-early_block_placement	使能 Macro Instance 优先放置开关
-auto_dispose_macro	PDS 软件自适应选择宏模块优先放置开关
-dispose_drm_prior	当用户选择禁止 PDS 软件自适应选择宏模块优先放置时，用户选择 DRM 的放置顺序
-dispose_apm_prior	当用户选择禁止 PDS 软件自适应选择宏模块优先放置时，用户选择 APM 的放置顺序
-dispose_dram_prior	当用户选择禁止 PDS 软件自适应选择宏模块优先放置时，用户选择 DRAM 的放置顺序
-dispose_carrychain_prior	当用户选择禁止 PDS 软件自适应选择宏模块优先放置时，用户独立选择 Carry Chain 的放置顺序
-slack_prior_in_global_router	全局绕线 Slack 优先模式
-slack_weight	控制时序因素在布线过程中的权重
-pin_cost_increasing_speed	拥塞 Pin 增加的代价值
-congest_overlap_limit	控制拥塞布线模式下，最大布线迭代次数
-max_ctrl_chain_length	CE/RS 控制信号链最大长度

表 9-12 pnr

### 9.13 report\_power 报告功耗信息

选项	
-help	打印帮助消息
-tempgrade	温度等级

-process	工艺偏差
-custom_junction	是否用户指定节点温度
-ambient	环境温度
-junction	指定节点温度
-custom_ja	用户指定热系数
-theta_ja	指定热系数
-ariflow	气流条件
-heatsink	散热情况
-theta_sa	散热热阻系数
-board_thermal	板子散热情况
-output_ppr	功耗报告输出文件
-output_pps	工程的功耗报告设置文件
-output_ppf	输出到 PPP 的功耗评估文件
-vcd	用户指定的后仿真文件，该文件记录了仿真波形，通过该文件能够获取信号的翻转情况
-tb_module	用户指定的仿真文件中顶层 module 名
-design_name	用户指定的设计 design 在 test bench 中的例化名
-output_pf	IO 外部负载等电容
-clk_freq	默认时钟频率
-togg	默认翻转率
-prob	默认静态高电平概率
-io_togg	I/O 的默认翻转率
-io_prob	I/O 的默认使能概率
-ram_togg	DRM 的默认翻转率
-ram_prob	DRM 的默认使能概率
-dsp_togg	APM 的默认翻转率
-dsp_prob	APM 的默认使能概率
-hsst_togg	HSST 的默认翻转率
-hsst_prob	HSST 的默认使能概率
-vcc	VCC 电压
-vccio33	Bank VCCIO3.3 电压
-vccio25	Bank VCCIO2.5 电压
-vccio18	Bank VCCIO1.8 电压
-vccio15	Bank VCCIO1.5 电压
-vccio12	Bank VCCIO1.2 电压
-vccaux	辅助电源 VCCAUX 电压
-vcc33a	VCCAUX.Default 电压
-vccint	VCC.Default 电压
-vcc33	VCCAUX.Default 电压

表 9-13 report power

### 9.14 report\_timing 报告时序信息

选项	
-help	打印帮助消息
-delay_type	设置时序分析的类型
-nworst	设置每个时序路径终点报告的最大路径数目
-max_path	设置时序报告的最大路径数目
-input_pins	显示输入 Pins
-slack_greater_than	设置报告的 slack 最小范围
-slack_less_than	设置报告的 slack 最大范围
-disable_package_delay	设置时序报告是否计算封装 (package pin) 的延迟
-report_io_datasheet	设置时序报告是否报告 IO 的时序特性
-ignore_db_version	不检查 DB 版本
-from	指定时序报告中时序路径的 from 节点
-to	指定时序报告中时序路径的 to 节点
-through	指定时序报告中时序路径的 through 节点

表 9-14 report\_timing

### 9.15 compile\_simlib 编译仿真库

选项	
-help	打印帮助消息
-Simulator	编译库时使用的仿真器
-family	指定芯片系列进行编译仿真库
-language	指定硬件描述语言类型
-library	指定编译仿真库种类
-simulator_exec_path	仿真器可执行文件所在路径
-directory	指定保存编译仿真库的文件夹

表 9-15 仿真

## 10常见错误

### 10.1 Constraint error

错误编号	错误说明	解决方法
ConstraintEditor 0008		
ConstraintEditor 0009		
ConstraintEditor 0010		
ConstraintEditor 0011		
ConstraintEditor 0012		
ConstraintEditor 0013		
ConstraintEditor 0014		
ConstraintEditor 0015	在约束命令 def_port 中，指定约束到某个位置的关键字必须是 LOC	
ConstraintEditor 0047		
ConstraintEditor 0048		
ConstraintEditor 0049		
ConstraintEditor 0050		
ConstraintEditor 0051		
ConstraintEditor 0052		
ConstraintEditor 0053		
ConstraintEditor 0054		
ConstraintEditor 0055		
ConstraintEditor 0056		
ConstraintEditor 0057		
ConstraintEditor 0058		
ConstraintEditor 0059		
ConstraintEditor 0060		
ConstraintEditor 0061		
ConstraintEditor 0062		
ConstraintEditor 0063		

表 10-1 constraint error

### 10.2 Physical ConstraintEditor

错误编号	错误说明	解决方法
PhysicalConstraintEditor 0001	给定的坐标不在模型设定的坐标范围内	
PhysicalConstraintEditor 0002	给定的 cell 不合法或没有对	

	应的实现	
PhysicalConstraintEditor 0003	没有找到指定的 device instance	
PhysicalConstraintEditor 0004	给定的资源文件不存在或加载失败	
PhysicalConstraintEditor 0005	给定的 package pin 不合法	
PhysicalConstraintEditor 0006		
PhysicalConstraintEditor 0007		
PhysicalConstraintEditor 0016	指定的 instance 不是 logic instance	
PhysicalConstraintEditor 0017	对同一对象多次约束	
PhysicalConstraintEditor 0018	约束格式不正确，没有给关键字指定 value	
PhysicalConstraintEditor 0019		
PhysicalConstraintEditor 0020		
PhysicalConstraintEditor 0021		
PhysicalConstraintEditor 0022		
PhysicalConstraintEditor 0023		
PhysicalConstraintEditor 0024	必须指定具体的区域，用两个坐标 (x1,x2,y1,y2) 来表示	
PhysicalConstraintEditor 0025		
PhysicalConstraintEditor 0026		
PhysicalConstraintEditor 0027		
PhysicalConstraintEditor 0028		
PhysicalConstraintEditor 0029		
PhysicalConstraintEditor 0030		
PhysicalConstraintEditor 0031		
PhysicalConstraintEditor 0032		
PhysicalConstraintEditor 0033		
PhysicalConstraintEditor 0034	region 名字必须是以字符串，数字，下划线组成，并且不能以数字开头	
PhysicalConstraintEditor 0035	region 的值必须是以两个坐标 (x1,x2,y1,y2) 的形式	
PhysicalConstraintEditor 0036	指定对象在不通过区域重复定义	
PhysicalConstraintEditor 0037		
PhysicalConstraintEditor 0038		
PhysicalConstraintEditor 0039		
PhysicalConstraintEditor 0040		

PhysicalConstraintEditor 0041		
PhysicalConstraintEditor 0042		
PhysicalConstraintEditor 0043		
PhysicalConstraintEditor 0044		
PhysicalConstraintEditor 0045		
PhysicalConstraintEditor 0046		

表 10-2 PCE 错误

### 10.3 UserConstraintEditor

错误编号	错误说明	解决方法
UserConstraintEditor 0001	缺少设置当前配置项所依赖的配置项	
UserConstraintEditor 0002	指定了不支持的 option	
UserConstraintEditor 0003		
UserConstraintEditor 0004	command: %s not support !	
UserConstraintEditor 0005		
UserConstraintEditor 0006	没有给指定 option 指定对应的值	
UserConstraintEditor 0007	没有给定正确的命令行格式	
UserConstraintEditor 0008	指定对象不合法	
UserConstraintEditor 0009	指定的约束命令不正确	
UserConstraintEditor 0010	没有定义必须指定的对象	
UserConstraintEditor 0011		
UserConstraintEditor 0012	没有给定指定对象的类型用以区分	
UserConstraintEditor 0013		
UserConstraintEditor 0014		
UserConstraintEditor 0015		
UserConstraintEditor 0016		
UserConstraintEditor 0017		

表 10-3 UCE 错误

### 10.4 Flow error

#### 10.4.1 FabFlow

错误编号	错误说明	解决方法
FabFlow 0001	提示没有加入 design 文件	双击或右键 Pango Design Suite->Navigator 中 Designs, 添加 design 文件
FabFlow 0002	文件不存在或文件打开失败	
FabFlow 0003	当前 action 没有定义相应的 CanbeRun 处理函数	查看 action 的调用行为确定是否正解, 有必要时要重新实现该 action 的 CanbeRun 函数

FabFlow 0004	当前 action 执行了一个没有重写的函数	查看 action 的调用行为确定是否正确，有必要时要重新实现该 action 的特定函数
FabFlow 0005	当前 action 执行了一个没有重写的 executor 函数	查看 action 的调用行为是否正确，有必要时要重新实现该 action 的 executor 函数
FabFlow 0006	arch load 失败，背后的原因很多，一般伴随着其它详细错误信息	根据详细信息定位问题
FabFlow 0007	检测到当前没有 Design 文件	双击或右键 Pango Design Suite->Navigator 中 Designs，添加 design 文件
FabFlow 0008	综合工具没有设置路径	菜单栏中：Tools -> Preference -> 设置 Synplify pro 的路径
FabFlow 0009	集成版调用 Synplify 综合失败	
FabFlow 0010	集成版综合结束后删除 Synplify 产生的 synplify_impl 文件夹失败	找到该文件夹手动删除
FabFlow 0011	Flow 执行过程中读 DB 后返回了一个空 design	
FabFlow 0012	Flow 执行读 DB 时传入了一个空字符串	
FabFlow 0013	读 DB 失败	
FabFlow 0014	DB 文件与当前执行的 Action 不匹配	
FabFlow 0015		
FabFlow 0016	单步执行 action 与正常执行 action 不能同时进行	执行 clean -all 命令
FabFlow 0017	写 DB 文件失败	
FabFlow 0018	更新 Action 状态失败	
FabFlow 0019	当前软件不支持该工程文件	
FabFlow 0020	从 DB 中读 design name 失败	
FabFlow 0021	Timing library 加载失败	
FabFlow 0022		
FabFlow 0023		
FabFlow 0024		
FabFlow 0025		
FabFlow 0026		
FabFlow 0027		
FabFlow 0028	没有设置芯片器件型号	source 页面中双击图标设置芯片器件信号
FabFlow 0029	Synplify 没有设置 OEM 器件类型	source 页面中双击图标设置
FabFlow 0030	多个 design 文件汇总含有相同的 module 名，且该 module 被设置为 top module	删除 design 中含有同名 module 的文件
FabFlow 0031	综合失败或文件没有读写权限	重新执行综合
FabFlow 0032	文件指针为空	

FabFlow 0033		
FabFlow 0034		
FabFlow 0035	转换属性到物理约束文件失败	
FabFlow 0036		
FabFlow 0037		
FabFlow 0038		
FabFlow 0039		
FabFlow 0040		
FabFlow 0041		
FabFlow 0042		
FabFlow 0043		
FabFlow 0044		
FabFlow 0045		
FabFlow 0046		
FabFlow 0047		
FabFlow 0048		
FabFlow 0049		
FabFlow 0050		

表 10-4 FabFlow

## 10.4.2 DB

错误编号	错误说明	解决方法
DB 0001		
DB 0002		
DB 0003		
DB 0004		
DB 0005		
DB 0006		
DB 0007		
DB 0008		
DB 0009		
DB 0010		
DB 0011		
DB 0012		
DB 0013		
DB 0014		
DB 0015		
DB 0016		
DB 0017		
DB 0018		

DB 0019		
DB 0020		
DB 0021		
DB 0022		
DB 0023		
DB 0024		
DB 0025		
DB 0026		
DB 0027		
DB 0028		
DB 0029		
DB 0030		
DB 0031		
DB 0032		
DB 0033		
DB 0034		
DB 0035	Flow运行过程中读写DB时出现错误	一般情况下无法解决此类问题，可尝试重新运行Flow生成新的DB文件
DB 0036		
DB 0037		
DB 0038		
DB 0039		
DB 0040		
DB 0041		
DB 0042		
DB 0043		
DB 0044		
DB 0045		
DB 0046		
DB 0047		

表 10-5 DB

#### 10.4.3 UiWgt

错误编号	错误说明	解决方法
UiWgt 0001	获取 FAB_CFG_PATH 路径失败	添加 FAB_CFG_PATH 环境变量
UiWgt 0002	Fab 软件向前兼容，旧版本的 Fab 软件不兼容新版本的 Fab 工程文件	升级 Fab 软件
UiWgt 0003	没有权限读写工程文件或该工程文件已经被其他进程占用	
UiWgt 0004	集成版调用综合工具进行综合失败	
UiWgt 0005	action 中已经有该 option	

UiWgt 0006		
UiWgt 0007	插件执行操作失败，具体原因请参考调用的插件提示信息	
UiWgt 0008	Fab 软件找不到该插件	重新安装 Fab 套件
UiWgt 0009	调用插件失败	
UiWgt 0010		
UiWgt 0011		

表 10-6 UiWgt

## 10.5 DRC error

错误编号	错误说明	解决方法
DRC 0001	用户设计网表中没有 IO inst。可能是综合时没有插入相关 inst	修改综合相关选项中的设置
DRC 0002	用户设计的 Design 中， Port 连接多个 IO inst	修改设计用例
DRC 0003	由于芯片架构的限制，使得一些端口无法直接连接到芯片端口，而 design 使用了直接连接。该问题一般在直接例化 GTP 单元时出现。如 GTP_LUTADDSUB 作为第一级使用时，其 CIN 端口必须悬空	修改综合相关选项中的设置
DRC 0004	用户设计的 Design 中，某些 inst 的 Pin 必须和 GTP_LUT5CARRY 相连	在这些中间级中插入 GTP_LUT5CARRY
DRC 0005	由于芯片架构的限制，使得一些 GTP_LUT5CARRY 和 GTP_LUTADDSUB 端口 COUT 无法直接连接到除 CIN 外的端口，而 design 使用了直接连接。该问题一般在直接例化 GTP 单元时出现。如 GTP_LUTADDSUB 作为中间级使用时，其 CIN 端口必须只能和 GTP_LUT5CARRY 或者 GTP_LUTADDSUB 的 COUT 相连	在这些中间级中插入 GTP_LUT5CARRY 或者 GTP_LUTADDSUB，使得该级的 COUT 从插入级的 COUT 和 SUM 端口同时输出
DRC 0006	由于芯片架构的限制，使得某些 Inst 无法直接连接到除 CIN 外的端口，而 design 使用了直接连接。该问题一般在直接例化 GTP 单元时出现。	在这些中间级中插入 GTP_LUT5CARRY 或者 GTP_LUTADDSUB，使得该级的 COUT 从插入级的 COUT 和 SUM 端口同时输出
DRC 0007	由于芯片架构的限制，使得某些 Inst 无法直接连接到除 CIN 外的端口，而 design 使用了直接连接。该问题一般在直接例化 GTP 单元时出现。如 GTP_MUX2LUT7 的端口 I0 必须和 GTP_MUX2LUT6 的 Z 端口相连	按照提示插入相应的 GTP
DRC 0008	由于芯片架构的限制，某些 GTP 间的有固定连接。该问题一般在直接例化 GTP 单元时出现。如 GTP_CLKDELAY 必须和	按照提示修改相应的 GTP 连接

	GTP_INBUFG 相连	
DRC 0009	由于芯片架构的限制，某些 GTP 间的有固定连接。该问题一般在直接例化 GTP 单元时出现。如 GTP_DLL 的端口“IFIFO_RADDR[0]”必须和 GTP_DDC 相连	按照提示修改相应的 GTP 连接
DRC 0010	由于芯片架构的限制，某些 GTP 间的有固定连接。该问题一般在直接例化 GTP 单元时出现。如 GTP_OGDDR 和 GTP_DDC 的端口“PADT”必须相连，且不能和其他 inst 相连	按照提示修改相应的 GTP 连接
DRC 0011	该问题主要原因是存在不可以 Packing 的单元，如 GTP_IODELAY 和 GTP_IBUF 必须直接相连，如果不满足规则，则会导致 Packing 失败	按照提示修改相应的 GTP 连接
DRC 0012	该问题主要原因是 DDR 存在不可以 Packing 的单元，如 GTP_IGDES7 和 GTP_IBUF 相连不正确，如果不满足规则，则会导致 Packing 失败	按照提示修改相应的 GTP 连接
DRC 0013	该问题主要原因是软件内部问题，DDR 上端口 DESCLK 和 DCC 的 SAMPLE_CLK 或者 PROBE_CLK 相连时，其参数 DESCLK_SRC 需要做相应的修改	按照提示修改相应的 GTP 连接
DRC 0014	该问题主要原因是软件内部问题，DDR 上端口 DESCLK 和 DCC 的 SAMPLE_CLK 或者 PROBE_CLK 相连时，其参数 DESCLK_SRC 需要做相应的修改	
DRC 0015	该问题主要原因是错误的用户约束导致的。如同一个 Bank 中的两个 IO 其标准设置 LVCMS33、LVCMS15 或者设置了相冲突的参考电压选项	检查 IO 标准设置
DRC 0016	用户设计的 Design 或相应的 pcf 文件中，在某些 Bank 里设置了不支持的 IO 标准属性	检查 IO 标准设置
DRC 0017	软件处理过程中，出现了不可以布局的 GTP 资源	检查用户设计
DRC 0018	用户设计综合后的资源超过芯片所能提供的资源	修改设计用例或者换其他芯片
DRC 0019	用户设计的 Design 或相应的 pcf 文件中，设置了不支持的 IO 标准属性	检查 IO 标准设置
DRC 0020	用户加入的 pcf 文件中，同一个 IO Group 中的两个 IO 必须设置相同的标准	修改相应的 IO 标准
DRC 0021		
DRC 0022		
DRC 0023		
DRC 0024		

DRC 0025		
DRC 0026		
DRC 0027		
DRC 0028		
DRC 0029		
DRC 0030		
DRC 0031		

表 10-7 DRC error

## 10.6 Map

错误编号	错误说明	解决方法
DeviceMap 0001	按照 Fab 软件模型, 部分 Inst 必须是以 Group 形式存在	
DeviceMap 0002	用户设计的 Design 中, GTP 单元例化时, 它们之间的连接关系错误	按照提示修改相应的 GTP 连接
DeviceMap 0003	该问题主要原因是存在不可以 Packing 的单元, 如 GTP_IODELAY 和 GTP_IBUF 连接错误	按照提示修改相应的 GTP 连接
DeviceMap 0004	用户设计中 START 数目超过芯片所允许的数目	修改设计用例
DeviceMap 0005	用户设计中 GRS 数目超过芯片所允许的数目	修改设计用例
DeviceMap 0006		
DeviceMap 0007		
DeviceMap 0008		
DeviceMap 0009		

表 10-8 map

## 10.7 PLACE

错误编号	错误说明	解决方法
Place 0001	参与布局或约束的单元必须是 Log Inst	
Place 0002	用户设计资源超出芯片所允许的数目	修改测试用例以减少资源或者选用其他芯片
Place 0003	用户设计资源超出芯片所允许的数目	修改测试用例以减少资源或者选用其他芯片
Place 0004	用户将 DQS 相连的 IO 端口必须约束在同	修改约束文件

	一个 DQS 组内	
Place 0005	通过布局算法, Inst 布局失败,该原因有多种, 通常有如下原因: 1、临界资源情况。2、约束错误	提供约束信息, 帮助软件更合理的布局
Place 0006	通过布局算法, Inst 布局失败,该原因有多种, 通常有如下原因: 1、临界资源情况。2、约束错误	提供约束信息, 帮助软件更合理的布局
Place 0007	DLL 和 DQS 约束冲突, 如果按此约束,将会导致布线不成功	提供约束信息, 帮助软件更合理的布局
Place 0008	DLL 和 DQS 约束冲突, 如果按此约束,将会导致布线不成功	提供约束信息, 帮助软件更合理的布局
Place 0009	由于芯片架构的问题, 单个 USCM 只能驱动四分之一象限。软件在处理时, 将单元复制四分, 如果约束其中之一, 其他 USCM 单元必须约束	对 USCM 备份单元进行约束
Place 0010	USCM 编号错误, 导致 USCM 名称不匹配	
Place 0011	由于芯片架构的问题, 单个 USCM 只能驱动四分之一象限。软件在处理时, 将单元复制四分, 如果约束其中之一, 其他 USCM 单元必须约束	对 USCM 备份单元进行约束
Place 0012	用户设计中, GTP_CLKBUFG 、GTP_CLKBUFGCE 等需要 USCM 资源的 GTP 数目过多	修改设计用例, 减少相关 GTP
Place 0013	用户约束的 IO 标准在本器件中不支持或者标准名称错误	检查 pcf 中标准名称
Place 0014	IO Group 中的 IOB 和 IOL 相对位置错误	
Place 0015	IO Group 中不存在 IOB	
Place 0016		
Place 0017	在 FAB 软件中, Group 无法找到合适位置	通过手工约束指导布局
Place 0018	用户约束的 IO 标准属性错误	检查约束文件
Place 0019	用户约束的 IO 标准属性错误。如在同一 Bank 中设置了不同 VCCIO 的标准	检查约束文件
Place 0020	用户约束错误。在设置具有参考电压的标准属性时, 相应的用于设置参考电压的 IO 不可以约束 IO 端口	检查约束文件
Place 0021	Ref Pin 布局失败。在设置具有参考电压的标准属性时, 相应的用于设置参考电压的 IO 不可以约束 IO 端口	检查约束文件
Place 0022		
Place 0023	VCC/GND 模型错误, 不支持相应实现方法	更换其他系列

Place 0024	模型错误，不支持相应实现方法	更换其他系列
Place 0025	在 Floor Plan 对应的位置没有可布局资源	更换其他系列
Place 0026	模型错误，不支持相应实现方法	更换其他系列
Place 0027	布局时，inst 被布置在芯片之外,用户约束错误或者软件错误！	修改用户约束
Place 0028	布局结果中，Group 中的各个 Inst 布局失败	
Place 0029		
Place 0030	根据软件模型，部分单元根据不同资源具有多种实现方式。该问题是约束过程中使用了错误的实现方式。比如，在 CLMA 上，GTP_LUT 使用了 CLMS 的实现方法	检查约束文件
Place 0031	模型错误，不支持相应实现方法	更换其他系列
Place 0032	模型错误，不支持相应实现方法	更换其他系列
Place 0033	软件模型错误。Time 模型上端口在 instance 上找到	更换其他系列
Place 0034	设计用例中，直接例化 GTP 单元，这些 GTP 单元间连接错误。如 GTP_CLKDELAY 其 CLKIN 端口只能和 GTP_INBUFG 或者 GTP_INBUFGDS 相连	修改用户设计
Place 0035	CLK_DELAY 和相连的 CLK PAD 约束冲突	检查用户约束文件
Place 0036	在当前芯片上，CLK PAD 数目不够，导致布局失败	
Place 0037	CLK PAD 资源不够，或者相连的 Inst 约束有误	修改约束文件
Place 0038	CLK PAD 资源不够，或者相连的 Inst 约束有误	修改约束文件
Place 0039	器件资源不够	修改设计用例或更换其他器件
Place 0040		
Place 0041		
Place 0042		
Place 0043		
Place 0044		
Place 0045		
Place 0046		
Place 0047		
Place 0048		
Place 0049		
Place 0050		
Place 0051		

Place 0052		
Place 0053		
Place 0054		
Place 0055		
Place 0056		
Place 0057		
Place 0058		
Place 0059		
Place 0060		
Place 0061		
Place 0062		
Place 0063		
Place 0064		
Place 0065		
Place 0066		
Place 0067		
Place 0068		
Place 0069		
Place 0070		
Place 0071		
Place 0072		
Place 0073		
Place 0074		
Place 0075		
Place 0076		
Place 0077		

表 10-9 place

## 10.8 Route error

错误编号	错误说明	解决方法
Route 0001	检测到 net 中存在非法连接，即起点和终点之间不存在布线路径。	
Route 0002	在布线模块中检测到当前不是布线状态。	
Route 0003	检测到一个 net 中存在多个驱动。一个 net 只允许有一个驱动。	
Route 0004	手动布线的 loading 布线结果阶段，发现异常	
Route 0005	寻找某个 mux 中的一段 arc 失败。	
Route 0006	往某个电路单元上放置 VCC, GND 等应用，找不到合适的电路。	
Route 0007	某个 load 需要重新布线，但是在没有找到对	

	应的布线 pin。	
Route 0008	布线失败。	
Route 0009		
Route 0010		
Route 0011		
Route 0012		
Route 0013		
Route 0014		
Route 0015		
Route 0016		
Route 0017		

表 10-10 router

## 10.9 Design Editor error

错误编号	错误说明	解决方法
DesignEditor 0001	布线用到了 Routing Arc 之后，通过 Src 端口找不到对应的 Design Net。	
DesignEditor 0002	当布线完成之后，还存在有没有 driver 的 Design Net，而 Design Net 是一定需要有一个 driver 的。	
DesignEditor 0003	在画线的时候，当前 Design Net 经过的 anchor 为空。	
DesignEditor 0004	PnR 之后，载入 PnR DB 文件失败。	
DesignEditor 0005	PnR 之后，其 DB 文件没有导入进来。	

表 10-11 DE error

## 10.10 Timing error

错误编号	错误说明	解决方法
Timing 0001	当完成 PnR 后，还存在有 design port 没有到 arch pin 的映射，因为在用户设计中，每个 design port 都必须连到一个 IO buffer 元器件。	
Timing 0002	design net 到 arch pin 之间的映射失败	
Timing 0003	当 design port 类型为 INOUT 时，当 port 两端的 node 的连接关系不能保存时，就会报错。	
Timing 0004	对 grid device 进行例化时，无法创建有内部逻辑的 Instance，会导致无法创建该 device 的时序图。	
Timing 0005	对 routing 网络构建时序图失败	

Timing 0006	在建立 arch 中的 timing path 时，不能获得与之相连的 delay 值，导致此错误。	
Timing 0007	该 grid device 没有从 base device view 中获取到 timing view，从而不能构建时序网表。	
Timing 0008	原因有两种：第一种是将 timing graph 复制到顶层 architecture 时，app node 没有直接连到 IT node；第二种是在中途取消了构建时序表。	
Timing 0009	在时序图中，delay 保存在 timing view 中，类型为 port，该 timing view 在 delay 传送到时序图之前已经删除了	
Timing 0010	1、从时序表中没有得到时序图，导致此错误发生；2、在时序图中没有找到 timing node。	
Timing 0011	缺少某些文件，具体情况要根据报错信息分析。	
Timing 0012	可能是 routing arc 中的 src port 和 snk port 还有 arc 没有找到，在文件中查看这些 port。	
Timing 0013	没有获取到该 device 的 Instance。	
Timing 0014	硬件模型中不支持 delay model。	
Timing 0015	这个 device 在 arch 中没有找到	
Timing 0016	timing node 没有找到或者创建失败，导致无法在时序表中标注 timing info。	
Timing 0017		
Timing 0018	timing node 名字错误	
Timing 0019	grid device 中某管脚有多个输入驱动	
Timing 0020	在 tm 文件中没有找到 ctf 文件中定义的 timing arc	修改 tm 文件或者 ctf 文件
Timing 0021	latch 中 D 管脚必须连着 delay arc 的 starting，其他方式则会报错。	
Timing 0022	latch 中 D 管脚必须连着 check arc 的 ending，其他方式则会报错。	
Timing 0023	instance 如果已经创建了，再次创建就会报错。	
Timing 0024	timing cell 中没有找到端口	
Timing 0025	库中还没有加入这个单元，目前不支持。	
Timing 0026	在 device 中没有找到相应的 prim device	
Timing 0027	在 device 中没有找到相应的管脚	
Timing 0028	在 vop 中没有找到相应的 timing model	
Timing 0029	放置到 grid device 上的 instance 多于一个。	
Timing 0030	在将 structure netlist 中的 pin 映射到 timing netlist 时，在 timing netlist 中没有找到相对应的 pin。	
Timing 0031		

Timing 0032		
Timing 0033		
Timing 0034	在 grid device 中没有找到 timing view	
Timing 0035	在 grid device 中没有找到 delay model	
Timing 0036	在布局中产生一个管脚有多个驱动输入或者接受多个信号，导致不能构建时序图。	
Timing 0037	没有获取到 base netlist 中例化的 Instance，导致不能 map。	
Timing 0038	design 没有扁平化。时序图构建失败	
Timing 0039	构造时序网表失败，在驱动管脚中没有找到时序图端点	
Timing 0040	构造时序图失败，在接受管脚中没有找到时序网表端点	
Timing 0041	对于时钟传输，时钟沿类型必须是 01 或者 10，不允许是 XX。	
Timing 0042	某些功能函数在目前的版本中还没有实现。	
Timing 0043	domain 已经存在，不能再添加 domain	
Timing 0044	模板种类中不支持上述类型。	
Timing 0045	模板种类中不支持上述类型。	
Timing 0046	模板种类中不支持上述类型。	
Timing 0047	模板种类中不支持上述类型。	
Timing 0048	模板种类中不支持上述类型。	
Timing 0049	模板种类中不支持上述类型。	
Timing 0050	模板种类中不支持上述类型。	
Timing 0051	模板种类中不支持上述类型。	
Timing 0052	模板种类中不支持上述类型。	
Timing 0053	模板种类中不支持上述类型。	
Timing 0054	模板种类中不支持上述类型。	
Timing 0055	模板种类中不支持上述类型。	
Timing 0056	模板种类中不支持上述类型。	
Timing 0057	模板种类中不支持上述类型。	
Timing 0058	模板种类中不支持上述类型。	
Timing 0059	一般是名字冲突，修改库文件名字。	
Timing 0060	相应模板已经声明了	
Timing 0061	相应单元已经声明了	
Timing 0062	不能添加端口总线，该端口总线已经存在	
Timing 0063	不能添加端口，端口已经存在	
Timing 0064	在命令\" set_clock_latency \"中定义的管脚不是时钟管脚	
Timing 0065	不能确定 launch clock 和 capture clock 的触发沿	

Timing 0066	未知错误	
Timing 0067	\'set_input_delay\'失败, 在时序图中没有找到相应端口	修改要设置的端口
Timing 0068	\'set_output_delay\'失败, 在时序图中没有找到相应端口	修改要设置的端口
Timing 0069	起始数据不能保存	
Timing 0070	在整个 timing propagation 中某处计算失败, 沿着 timing propagation 从头开始分析。	
Timing 0071	不能确定 launch clock 和 capture clock 之间的关系	
Timing 0072	时钟已经存在	
Timing 0073	时钟组已经存在	
Timing 0074	在 Top-level 进行时序计算, netlist 不能构建时序图。	
Timing 0075	在线路中发现组合环路, 时序分析失败	检查电路, 去掉环路
Timing 0076	创建时序图失败, 同 timing 0074	
Timing 0077	未知信号跃迁的时钟不能被传输	重新定义时钟信号
Timing 0078	时序模型库没有找到	
Timing 0079	物理时序模型库没有找到	
Timing 0080	物理时序模型库没有找到	
Timing 0081	物理时序模型库没有找到	
Timing 0082		
Timing 0083		
Timing 0084		
Timing 0085		
Timing 0086		
Timing 0087		
Timing 0088		
Timing 0089		
Timing 0090		
Timing 0091		
Timing 0092		
Timing 0093		
Timing 0094		
Timing 0095		
Timing 0096		
Timing 0097		
Timing 0098		
Timing 0099		
Timing 0100		
Timing 0101		

Timing 0102		
Timing 0103		
Timing 0104		
Timing 0105		
Timing 0106		
Timing 0107		
Timing 0108		
Timing 0109		
Timing 0110		
Timing 0111		
Timing 0112		
Timing 0113		
Timing 0114		
Timing 0115		
Timing 0116		
Timing 0117		
Timing 0118		
Timing 0119		
Timing 0120		
Timing 0121		
Timing 0122		
Timing 0123		
Timing 0124		
Timing 0125		
Timing 0126		
Timing 0127		
Timing 0128		
Timing 0129		
Timing 0130		
Timing 0131		
Timing 0132		
Timing 0133		
Timing 0134		
Timing 0135		
Timing 0136		
Timing 0137		
Timing 0138		
Timing 0139		
Timing 0140		
Timing 0141		

Timing 0142		
Timing 0143		
Timing 0144		
Timing 0145		
Timing 0146		
Timing 0147		
Timing 0148		
Timing 0149		
Timing 0150		
Timing 0151		
Timing 0152		
Timing 0153		
Timing 0154		
Timing 0155		
Timing 0156		
Timing 0157		
Timing 0158		
Timing 0159		

表 10-12 timing error

## 10.11 Simulation error

错误编号	错误说明	解决方法
Simulation 0006	在指定路径下找不到仿真器的可执行文件	将路径指定为有仿真器可执行文件的文件夹下
Simulation 0007	不支持此类型仿真器	目前软件支持的仿真器包括 Modelsim 和 Questasim，选择其中的一种即可
Simulation 0008	编译仿真库不存在	先运行编译仿真库，其中 family 选项要与运行仿真时的芯片系列保持一致

表 10-13 Simulation error

## 免责声明

### 版权声明

本文档版权归深圳市紫光同创电子有限公司所有，并保留一切权利。未经书面许可，任何公司和个人不得将此文档中的任何部分公开、转载或以其他方式披露、散发给第三方。否则，公司必将追究其法律责任。

### 免责声明

1、本文档仅提供阶段性信息，所含内容可根据产品的实际情况随时更新，恕不另行通知。如因本文档使用不当造成的直接或间接损失，本公司不承担任何法律责任。

2、本文档按现状提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档在此未以禁止反言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。

3、公司保留任何时候在不事先声明的情况下对公司系列产品相关文档的修改权利。