



VIVADO EXPERT 系列 1 2017



高速设计收敛技术
Balachander Krishnamurthy

VIVADO[®]

HLx Editions

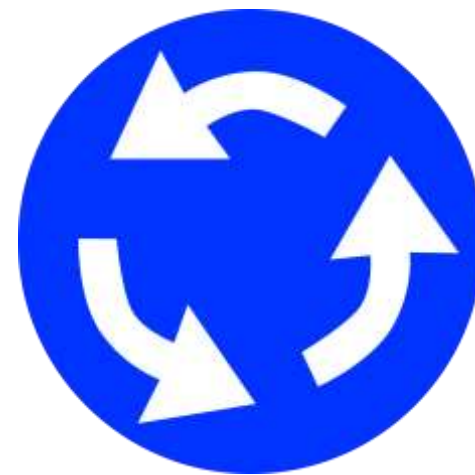
会议议程

- 高速设计挑战
- 设计分析
- 设计指南
- 复杂性与拥塞分析
- 总结

VIVADO EXPERT 系列 1 2017


高速设计挑战

- 高性能要求 (>400 MHz)
- 过晚发现问题
- RTL 编码风格
- 约束不彻底
- 逻辑层次过多
- 高扇出网络与控制集过多
- 次优时钟和大型块 (BRAM, DSP)
 - 功耗与性能之间的权衡
- 复杂性与拥塞
- 时序收敛迭代过多



UltraFast 设计方法 (UG949)



- 进行综合后迭代以实现更快速的分析与可预测的结果
 - RTL 编码风格与实例
- 在所有方面以检查表方式提供建议：
 - 单板和器件规划
 - 设计创建
 - 实现
 - 设计收敛
-  运行“Report Methodology”以尽快确定问题并解决问题
 - 综合、时序、XDC 及时钟方法检查



UG1231: UltraFast 设计方法快捷参考指南

UltraFast Design Methodology Quick Reference Guide (UG1231)

INTRODUCTION

The UltraFast™ Design Methodology is a set of best practices recommended by Xilinx to maximize productivity and reduce design iterations of complex systems, including embedded processor subsystems, analog and digital processing, high-speed connectivity, and network processing. See the *UltraFast Design Methodology Guide for the Vivado Design Suite* (UG949) for more information.

The UltraFast Design Methodology Checklist (XTP301) includes common questions that highlight typical areas where design decisions have downstream consequences and draws attention to potential problems that are often unknown or ignored. It provides easy access to related collateral. The checklist is available within the Xilinx Documentation Navigator tool (DocNav).

This quick reference guide highlights key design methodology steps to achieve quicker system integration and design implementation and to derive the greatest value from Xilinx® devices and tools. Pointers to related collateral are also provided. The main design tasks covered in this guide include:

- Board and Device Planning
- Design Entry and Implementation
- Top-Level Design Validation
- Design Analysis
- Design Closure

Refer to the UltraFast Design Methodology – System-Level Design Flow available within the Xilinx Documentation Navigator tool (DocNav) for pointers to all design hubs and specific collateral.



UG1231 (v2018.3) November 11, 2018

BOARD AND DEVICE PLANNING

PCB Designer

Examine Key Interfaces

- Validate part orientation and key interfaces

Examine the PCB Layout

- Perform the Memory Interface and Transceiver Checklists
- Follow PCB layout recommendations
- Ensure final FPGA pinout is signed off by FPGA designer

Review the Schematic

- Complete PCB Checklist review
- Check PDS, configuration, and power supplies
- Validate I/O state before, during, and after configuration

Manufacture and Test

- Verify the configuration sequence, power supplies, and I/O performance with the test I/O project

See Also:
[UG949: Board and Device Planning PCB Design Checklist](#)
[Memory Interface IP Design Checklists](#)
[Schematic Design Checklists](#)

FPGA Designer

Analyze Device for Pinout

- Examine transceiver and bonded I/O locations
- Examine SSI technology I/O planning
- Validate part orientation and key interfaces

Define I/O Pinouts for Key Interfaces

- Create I/O planning projects
- Define and validate memory controllers, GTs, and PCIe® technology locations
- Establish a clocking skeleton
- Minimize floorplan distance between connected IP

Define Final Pinout

- Merge interface projects into a final I/O project
- Validate DRCs and SSN analysis
- Implement design to check clocking and I/O rules
- Use the final I/O project for production test

Estimate Power

- Determine power budget and thermal margin using Xilinx Power Estimator (XPE)
- Apply toggle rates using knowledge of prior designs

See Also:
[UG949: Board and Device Planning Power Estimation and Optimization Design Hub](#)
[I/O and Clock Planning Design Hub](#)

DESIGN ENTRY AND IMPLEMENTATION

Logic Designer

Define a Good Design Hierarchy

- Define relevant hierarchies to help global placement and floorplanning
- Insert I/O and clock components near the top level
- Add registers at main hierarchical boundaries
- Generate IP and review target device utilization

Build and Validate RTL Submodules

- Ensure design adheres to RTL coding guidelines
- Add sufficient registers around DSP and memories
- Use control signals only when absolutely necessary
- Use synthesis attributes to control final logic mapping
- Create simple timing constraints to review estimated timing and address paths with too many logic levels
- Review synthesis log files, utilization report, and elaborated view to identify sub-optimal mapping
- Run Methodology and RTL checks and review issues
- Implement the submodule in out-of-context (OOC) mode to validate implemented performance
- Review utilization and power against original budget
- Simulate the design to validate functionality

Assemble and Validate Top-Level Design

- Synthesize the top-level RTL design and resolve all connectivity issues
- Review top-level utilization and clocking guidelines
- Create and validate top-level constraints
- Iterate the RTL and constraints to fix Methodology and DRC issues and meet timing
- Proceed to implementation

See Also:
[UG949: Design Creation and Implementation Designing with IP Design Hub](#)
[Using IP Integrator Design Hub](#)
[Logic Synthesis Design Hub](#)
[Applying Design Constraints Design Hub](#)
[Implementation Design Hub](#)

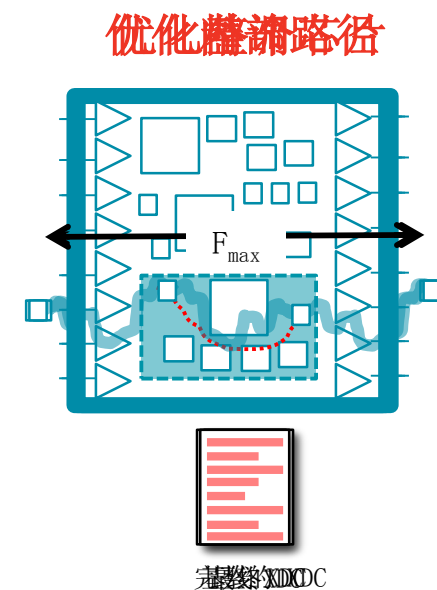
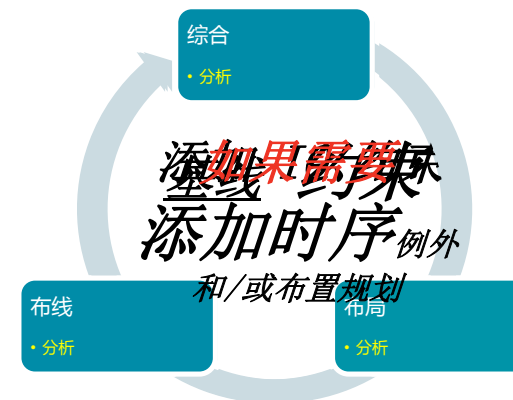
设计分析

约束、逻辑层次、控制集与高扇出网络

基线 - 彻底的约束

■ 基线：确保时序约束彻底

- 要定义所有时钟及时钟关系
- 要正确约束异步跨时钟域
 - 1 位 CDC：伪路径或时钟组约束
 - 多位 CDC：使用 `set_max_delay -datapath_only` 和/或 `set_bus_skew`
- IO 约束
- 其他点对点例外约束



逻辑层次

- 检查逻辑层次数量

- 指南：每个逻辑层次 500 ps（1 个 LUT + 1 个网络）
- 分析时使用：

```
report_design_analysis -logic_level_distribution -extend
```

- 建议：

- 修改 RTL 以减少逻辑层次的数量
- 使用基于实例的综合来优化模块
 - 综合选项：重定时、区域、性能、可布线性等。

速度等级	-1	-2	-3
7 系列	575ps	500ps	425ps
UltraScale	490ps	425ps	360ps
UltraScale+	350ps	300ps	250ps
UltraScale+ LV	490ps	425ps	360ps

基于实例的综合选项

■ 使用 XDC 实现基于实例的优化

```

set_property BLOCK_SYNTH.RETIMING 1 [get_cells U1]
set_property BLOCK_SYNTH.STRATEGY {ALTERNATE_ROUTABILITY} [get_cells U2]
set_property BLOCK_SYNTH.STRATEGY {AREA_OPTIMIZED} [get_cells U3]
set_property BLOCK_SYNTH.LUT_COMBINING [get_cells U3/inst1]
    
```

- 支持嵌套选项
- 以预置为策略
- 允许多个逻辑选项



基于实例的综合设置

策略
(捆绑预置)

逻辑选项

默认
占位面积
性能
可布线性
重定时
adder_threshold
comparator_threshold
shreg_min_size
fsm_extraction
lut_combining
flatten_inside_partition
extract_partition
flatten_hierarchy
control_set_threshold
max_lut_input
muxf_mapping
keep_equivalent_register

适用于实例的策略
和逻辑选项

控制集

使用 `report_utilization` 和 `report_control_sets -verbose` 进行分析

■ 控制集会影响布局和时序

■ 指南

- 如果唯一控制集的数量不足 slice 总数的 7.5%，则可以接受
- 如果唯一控制集的数量超过 slice 总数的 15%，则需要进行分析

■ 建议

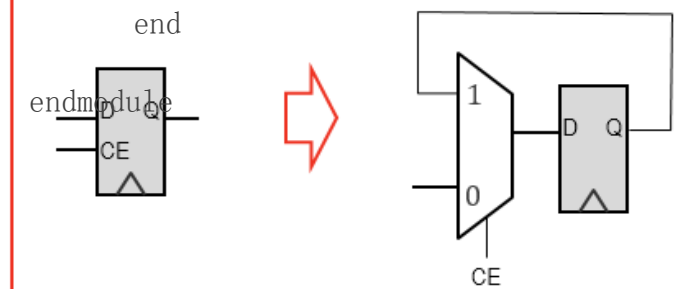
- 手动复制高扇出寄存器或使用 `PhysOpt`
- 利用全局综合选项更改默认值 (`-control_set_opt_threshold`)
 - 根据架构使用不同阈值
 - ✓ 7 系列使用的阈值为 4, UltraScale 和 UltraScale+ 使用的阈值为 2
- 在架构内实现时序关键低扇出 (<8) 控制信号
 - 在 RTL 中使用 `extract_enable` 或 `extract_reset`
 - 在 XDC 文件中:

```
set_property extract_enable/reset "no" [get_cells U1/tmp_reg]
```

```
module top (input clk,
            input [9:0] din, a, b
            (* extract_enable = "no" *)
            output reg [9:0] dout);

    reg [9:0] dinr, ar, br;
    wire en = ar = br;

    always @(posedge clk)
        begin
            ar <= a;
            br <= b;
            dinr <= din;
            if(en)
                dout <= dinr;
        end
endmodule
```



逻辑位于 D 输入侧的 CE 仿真

高扇出网络

运行 `report_high_fanout_nets` 进行分析

■ RTL 建议:

- 如果不是关键时序，而且高扇出网络直接馈送 FF
 - 将扇出 >25k 的网络推广到全局时钟

```
set_property CLOCK_BUFFER_TYPE BUFG [get_nets netName]
```

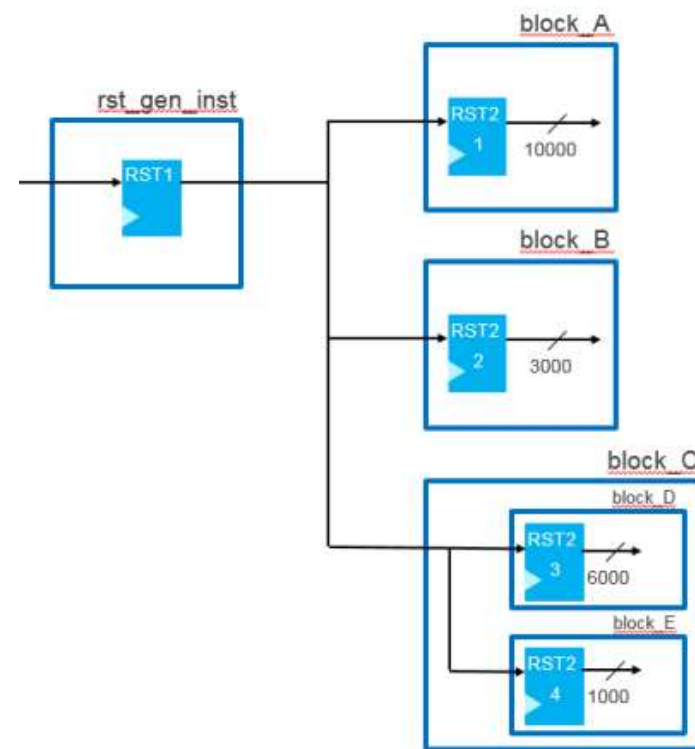
- 时序关键
 - 使用 KEEP 属性在 RTL 中复制寄存器
 - 使用 pblock 约束在每个 SLR 中进行复制
- 避免高扇出进出 DSP/BRAM

■ 综合建议:

- 避免将 MAX_FANOUT 用于全局控制信号
- 如果需要，只在小模块内使用 MAX_FANOUT

■ 使用 PhysOpt 复制高扇出网络驱动器

- `Phys_opt_design -force_replication_on_nets`
- `Phys_opt_design -directive AggressiveExplore`



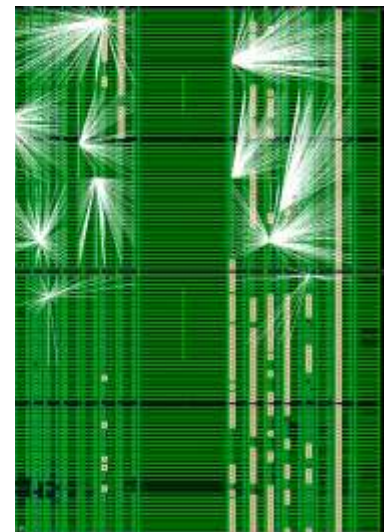
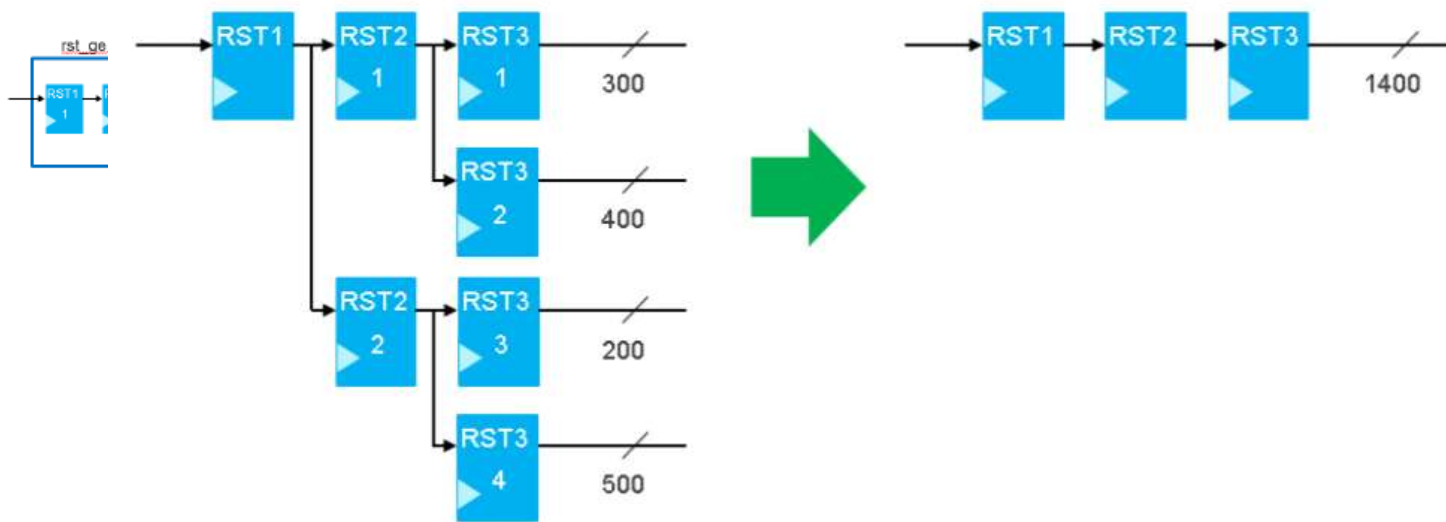
高扇出网络优化流程

- **综合：受限复制**
 - 避免将 MAX_FANOUT 用于全局控制信号
 - 如果需要，只在小模块内使用 MAX_FANOUT
 - 在手动复制单元上使用 KEEP
- **Opt Design：粗粒度复制**
 - 基于大设计层级
- **Place Design：中粒度复制**
 - 基于早期布局和时序信息
- **Phys Opt Design：精细粒度复制**
 - 基于精确时序信息

Opt Design 中的高扇出网络优化

■ Opt Design

- 在高扇出网络上插入、合并与分割 BUFG（默认执行）
- 使用 ‘-control_set_merge’ 合并 opt_design 中的低扇出控制信号
- 使用 ‘-merge_equivalent_drivers’ 合并所有 LUT 和 Flop 等效驱动器
- 使用 ‘-hier_fanout_limt <number>’ 控制基于模块的复制

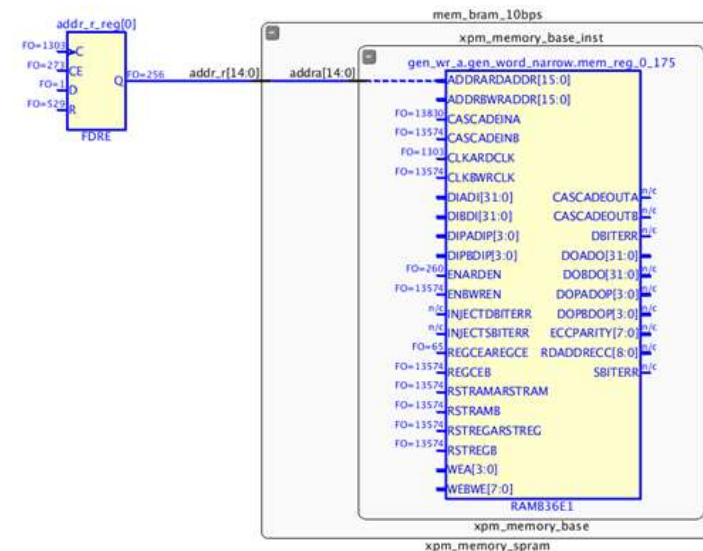


布局器中的高扇出网络优化

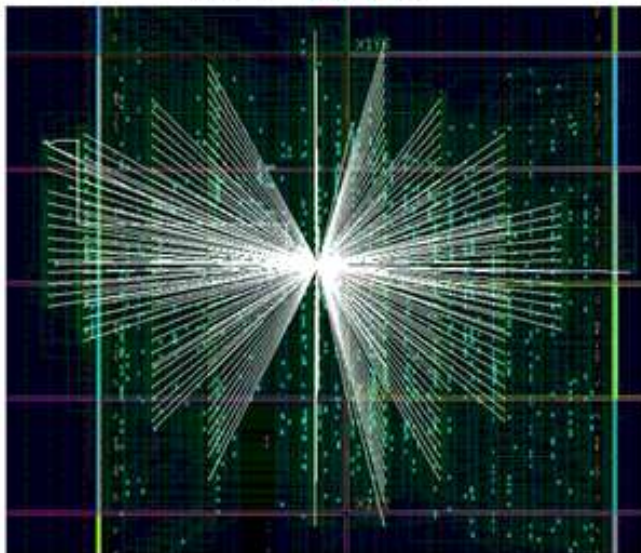
- 布局器中的物理综合 (PSIP)
 - 在全局布局过程中根据资源可用性自动插入 BUFG
 - 布局过程中使用 ‘-fanout_opt’ 复制高扇出网络
 - 复制连接到 DSP/BRAM 的低/中扇出网络
- 优势
 - 不需要在 RTL/综合层面进行猜测
 - 理想的控制集使用
 - 布局已知，而且复制基于驱动器和负载布局
 - 在布局器中较早解决时序关键的 HFN
 - 减少布局后的物理优化

PSIP 实例：控制信号

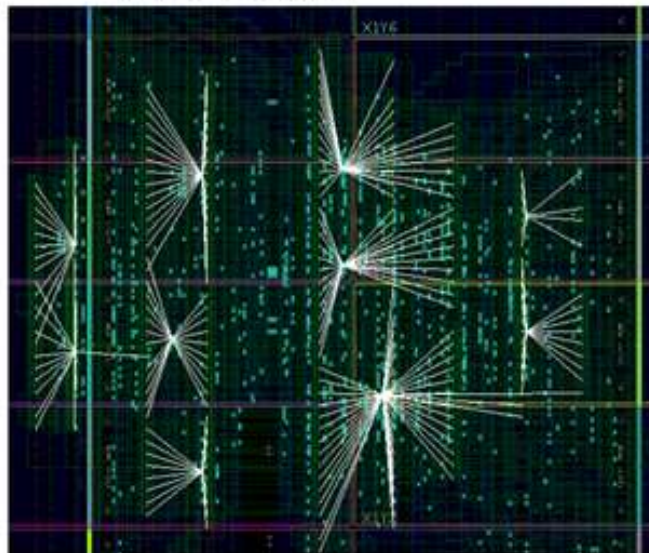
- 大型 BRAM 的 FF 驱动地址位



Without replication in placer
WNS = -0.312 ns



With replication in placer
WNS = 0.146 ns



设计指南

时钟、同步 CDC、BRAM 和 DSP

时钟指南

- 次优时钟损害时序
 - 对时钟方法检测进行检查并修复
- 尽可能减少时钟缓冲器的利用率
 - 通过组合相同的同步时钟来减少时钟数
 - 从 MMCM 反馈路径移除缓冲器（不支持延迟补偿）
 - 除非绝对需要，否则移除级联缓存
 - 当处在 INTERNAL 模式，从 MMCM 反馈路径移除缓冲器
- 布局规划要保留的时钟
 - 接近驱动器的低扇出时钟
 - 远离高利用率区域
- 避免使用在结构资源上布线的时钟
 - 不可预测的偏差，与其他网络在布线资源上的竞争
 - 负载超过 30 时，尽量插入时钟缓存器

同步 CDC 指南

■ 分析关键交叉

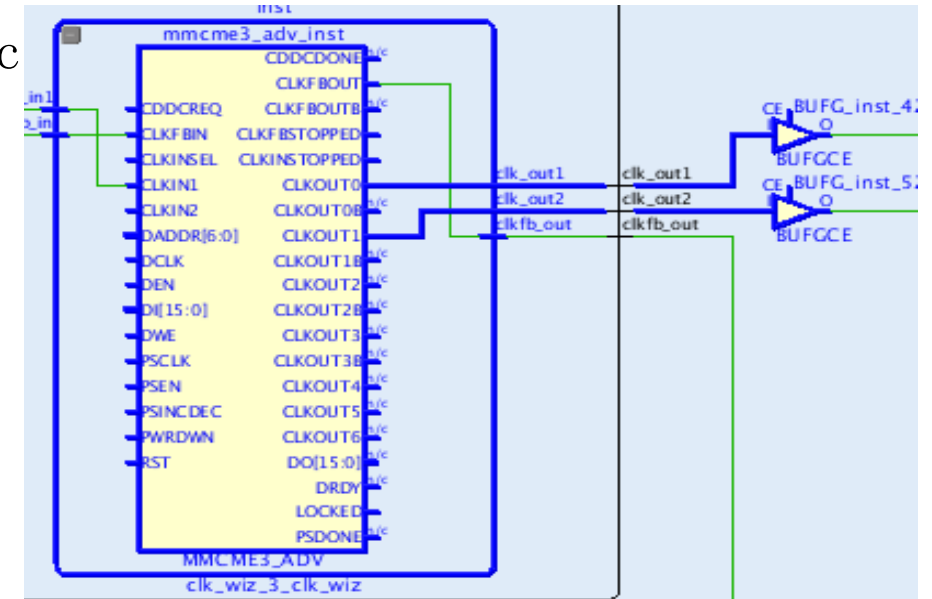
- 运行 `report_timing_summary` 或 `report_clock_interac`
- 严格的设置要求（对偏差更敏感）
- 最大数量的路径（对 TNS & THS 影响最大）

■ 检查并改善 CDC 时钟拓扑结构

- 配置 MMCM 的 VCO 以便在最高频率下运行
 - 降低时钟不确定性
- 在 MMCM 之前避免公共节点
- 避免级联缓存
- 避免跨越 IO/SLR 边界以最大限度降低偏差

■ 减少/放松 CDC 路径

- 作为异步处理（需要正确的同步电路）
- 使用多周期路径约束（放松设置）（需要 CE 逻辑）



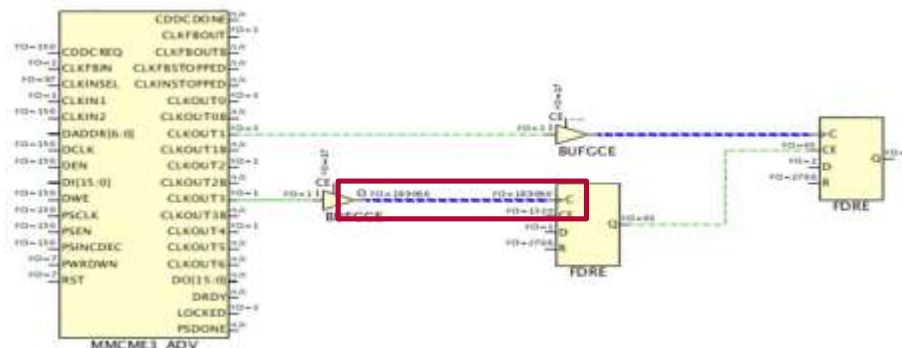
同步 CDC 中的时钟偏差

CLOCK_DELAY_GROUP 约束可限制同步时钟偏差

■ 匹配组内时钟网络延迟

- 用于约束同步 CDC 偏差
- 与使用 USER_CLOCK_ROOT 约束相比精度更高

■ 实例：



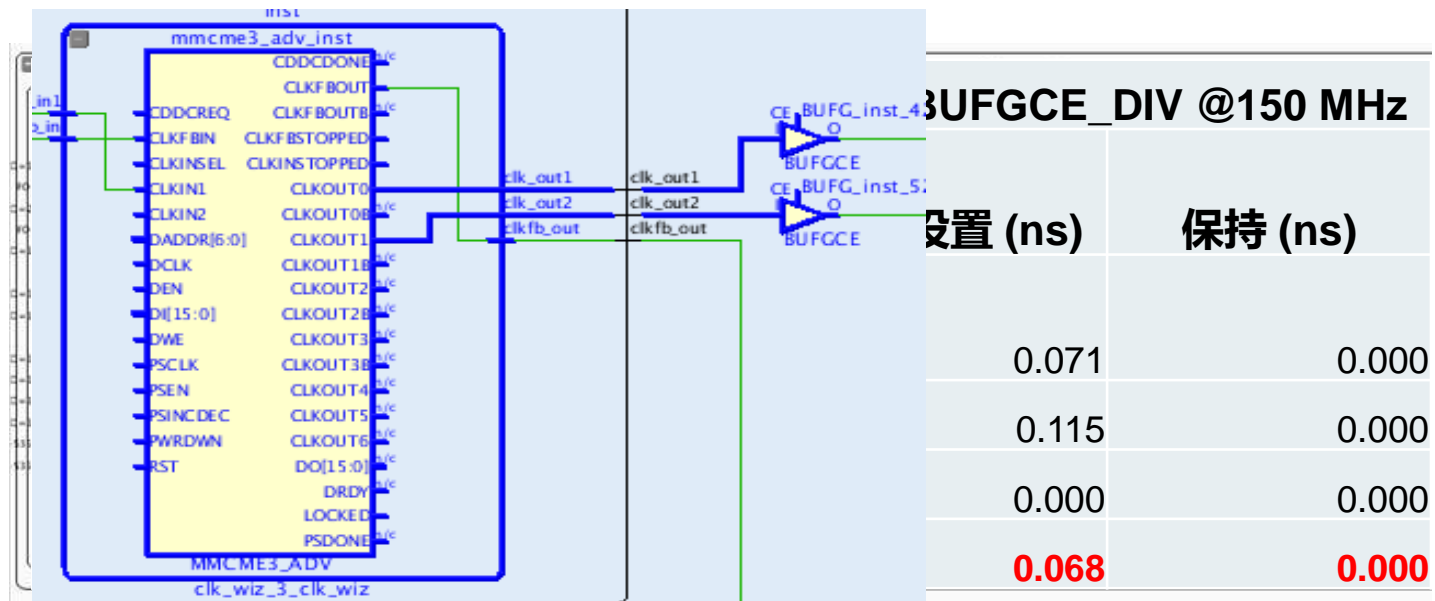
```
set_property CLOCK_DELAY_GROUP group_0 [get_nets {u1/clk_50M u1/clk_100M}]
```

■ 指南

- 适用于直接连接至缓冲器输出的网段
- 缓冲器必须具有相同的驱动单元 => 确保平衡的拓扑结构
 - 驱动器实例：MMCM、PLL、IBUFDS、GT_CHANNEL

利用 UltraScale 器件中的 BUFGCE_DIV

- 使用 BUFGCE_DIV 以执行分频
 - 移除相位误差 (~120ps) => 有助于设置和保持 CDC 路径
 - 具有简单周期比的时钟 (/1 /2 /4 /8)
 - 每个时钟区域仅有 4 个 BUFGCE_DIV
- 注意 BUFGCE 与 BUFGCE_DIV 之间的单元延迟差异
 - 两个时钟最好使用相同的缓存类型 (BUFGCE_DIV 可除以 1)



真实客户设计 - 案例研究

- 使用高速时钟及多个同步 CDC 进行设计
- 使用 report_clock_interaction 确定关键 CDC 路径（预布线）

Source Clock	Destination Clock	WNS (ns)	TNS (ns)	Failing Endpoints (TNS)	Total Endpoints (TNS)	Path Req (WNS)	WHS (ns)	THS (ns)	Failing Endpoints (THS)	Total Endpoints (THS)	Path Req (WHS)	Common Primary Clock	Inter-Clock Constraints
sys_clk_491	sys_clk_491	0.165	0.000	0	430480	2.034	-0.253	-289.459	10007	430480	0.000	Yes	Partial False Path
sys_clk_245	sys_clk_491	0.916	0.000	0	3292	2.034	-0.383	-271.798	2022	3292	0.000	Yes	Partial False Path
sys_clk_491	sys_clk_245	0.420	0.000	0	3751	2.034	-0.206	-254.735	2687	3751	0.000	Yes	Partial False Path
sys_clk_245	sys_clk_122	0.461	0.000	0	1125	4.069	-0.224	-73.057	619	1125	0.000	Yes	Partial False Path
sys_clk_245	sys_clk_61_44	10.444	0.000	0	3134	16.276	-0.234	-70.723	868	3134	0.000	Yes	Partial False Path
sys_clk_245	sys_clk_245	0.255	0.000	0	238146	4.069	-0.234	-46.686	1802	238146	0.000	Yes	Partial False Path
sys_clk_30_72	sys_clk_61_44	9.222	0.000	0	5348	16.276	-0.430	-41.222	545	5348	0.000	Yes	Partial False Path
sys_clk_30_72	sys_clk_122	27.186	0.000	0	12908	32.552	-0.356	-33.333	415	12908	0.000	Yes	Partial False Path
sys_clk_122	sys_clk_245	0.540	0.000	0	10781	4.069	-0.185	-21.341	580	10781	0.000	Yes	Timed
sys_clk_122	sys_clk_61_44	10.212	0.000	0	1083	16.276	-0.196	-14.464	172	1083	0.000	Yes	Timed

- 找出具有最严格设置要求和最高 THS 违规的 4 个时钟

- 改善时钟树并应用 CLOCK_DELAY_GROUP



块 RAM

进出块 RAM 的路径指南

- 在 LUTRAM (<8kb) 中实现更小的存储器
- 避免高扇出信号进出大型块
 - 使用 KEEP 属性根据层级在 RTL 中进行复制
 - 2016.3 提供整个流程的高级复制特性
- 使用 low-logic-level 路径以驱动控制信号
 - 根据需要复制地址总线或使用 PhysOpt
 - 针对性能/功耗, 建议使用 cascade_height 属性
- 针对高性能设计启用输出寄存器
 - 寄存器应位于内部或直接连接到输出引脚
- 使用 report_design_analysis 分析 块 RAM 路径

```
report_design_analysis -of_timing_paths [get_timing_paths \  
-from/to [get_cells -hier -filter {PRIMITIVE_GROUP==BLOCKRAM}] \  
-max 1000 -nworst 16 -unique_pins] -file <from><to>BLOCKRAM.rpt
```

```
...  
(* ram_style = "distributed" *)  
reg [31:0] mem [(2**8)-1:0];  
reg [7:0] addr_reg;  
...
```

```
...  
(* ram_style = "block",  
cascade_height = 1/8/32 *)  
reg [31:0] mem [(2**15)-1:0];  
reg [14:0] addr_reg;  
...
```


真实客户案例研究 - 未寄存的 RAMB 输出

- 来自相同 RAMB 的众多路径在 250MHz 时具有 4 个以上的逻辑级

The screenshot shows the 'Path Properties' window for a path with a slack of -0.604ns. The path is highlighted in red in the table below. The path is a setup path from a RAMB36E2 block to a DOA_REG primitive.

Slack	Logic Levels	Logical Path	End Point Pin Primitive	DOA_REG	DOB_REG
-0.024	0	RAMB36E2 FDRE	FDRE/D	1	0
-0.022	0	RAMB36E2 FDRE	FDRE/D	1	0
-0.009	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
-0.006	0	RAMB36E2 FDRE	FDRE/D	1	0
-0.006	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
-0.005	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
-0.002	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
0.004	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
0.007	0	RAMB36E2 FDRE	FDRE/D	1	0
0.007	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
0.007	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
0.009	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1
0.010	0	RAMB36E2 FDRE	FDRE/D	1	0
0.012	1	RAMB36E2 LUT5 FDRE	FDRE/D	1	1

Name	Path Type	Requirement	Clock Skew	Slack	Logic Levels	Logical Path	Start Point Pin Primitive	End Point Pin Primitive	DOA_REG	DOB_REG	BRAM Crossings	DSP Crossings	IO Crossings	Bounding Box Size	Clock Region Distance	#Blocks	High Fans
Path 1	SETUP	4.069	-0.387	-0.604	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.1% x 2%	(0, 0)	0	0
Path 2	SETUP	4.069	-0.387	-0.354	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.1% x 2%	(0, 0)	0	0
Path 3	SETUP	4.069	-0.390	-0.111	4	RAMB36E2 LUT6 LUT6 LUT6 MUXF7 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.1% x 1%	(0, 0)	0	0
Path 4	SETUP	4.069	-0.390	-0.523	4	RAMB36E2 LUT6 LUT6 LUT6 MUXF7 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.1% x 1%	(0, 0)	0	0
Path 5	SETUP	4.069	-0.383	-0.511	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 2%	(0, 0)	0	0
Path 6	SETUP	4.069	-0.383	-0.461	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 2%	(0, 0)	0	0
Path 7	SETUP	4.069	-0.390	-0.440	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 1%	(0, 0)	0	0
Path 9	SETUP	4.069	-0.390	-0.410	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 1%	(0, 0)	0	0
Path 10	SETUP	4.069	-0.390	-0.401	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 1%	(0, 0)	0	1
Path 11	SETUP	4.069	-0.390	-0.350	4	RAMB36E2 LUT6 LUT6 LUT6 MUXF7 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.1% x 1%	(0, 0)	0	1
Path 12	SETUP	4.069	-0.390	-0.371	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 1%	(0, 0)	0	0
Path 13	SETUP	4.069	-0.383	-0.365	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 2%	(0, 0)	0	0
Path 14	SETUP	4.069	-0.383	-0.316	4	RAMB36E2 LUT6 LUT6 LUT6 LUT6 FDCE	RAMB36E2/CLKARDCLK	FDCE/D	0	0	0	0	0	0.0% x 2%	(0, 0)	0	0

复杂性与拥塞

设计复杂性

使用 `report_design_analysis -complexity` 进行分析

- 复杂性衡量逻辑互连程度
- 复杂模块可能处于层级中靠下的几级
 - 更改 GUI 中的 ‘Hierarchical depth’

更大实例上的高 Rent、高平均扇出

高 LUT6%、高 MUXF* 利用率

Instance	Module	Rent	Average Fanout	Total Instances	LUT1	LUT2	LUT3	LUT4	LUT5	LUT6	Memory/LUT	DSP	RAMB	MUXF
utop	utop	0.51	2.73	38637	701	3507	2932	3016	3957	6128	1585	68	80	1025
mgtEngine (mgtTop)	mgtTop	0.27	1.80	1198	48	48	154	144	72					
cpuEngine (or1200_top)	or1200_top	0.52	3.78	10308	137	598	879	792	928	2461	416	4	22	312
fftEngine (fftTop)	fftTop	0.68	1.24	3315	34	1151	56	42	12	116	769	64	0	0
usbEngine0 (usb_top)	usb_top	0.70	4.12	11414	241	845	913	1015	1374	1717	200	0	29	331
usbEngine1 (usb_top_0)	usb_top_0	0.71	4.12	11414	241	845	913	1015	1374	1717	200	0	29	331

- 目的
 - 在布局之前找出具有高 rent (>0.65) 和/或高平均扇出 (>4.0) 的高难度模块 (>15k 个单元)
- 解决方案
 - 选项 1: 尝试不同综合选项, 采取从下到上的综合流程 (OOC)
 - 选项 2: 布局规划高难度模块

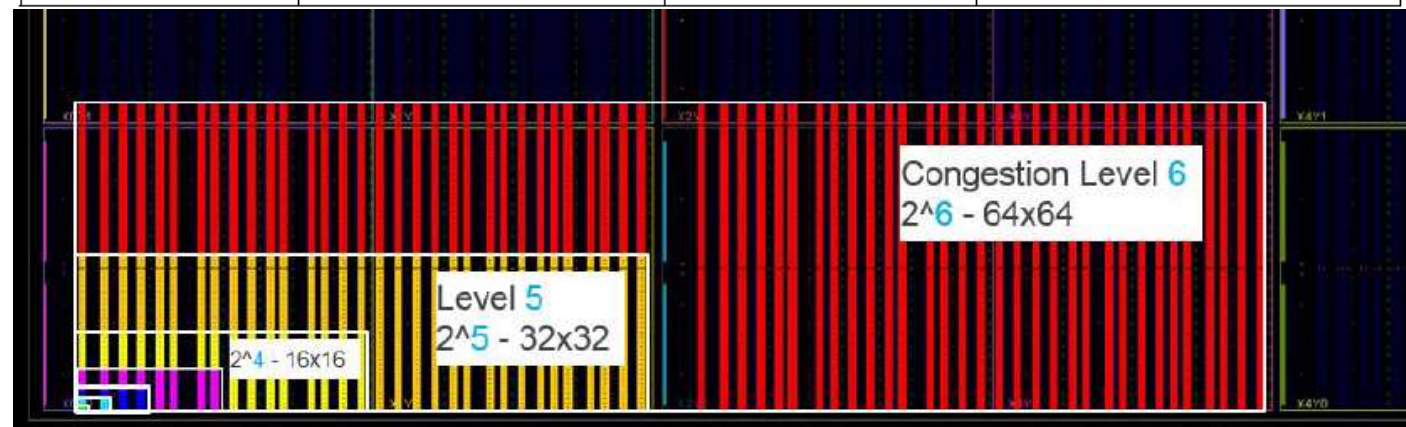
拥塞级别

■ 定义

- 拥塞级别相当于包含拥塞布线的正方形区域的大小
- 正方形大小基于互连 (INT_XnYm) 或 CLB (CLE_M_XnYm)

■ 拥塞级别范围

Level	Area	Congestion	QoR Impact
1, 2	2x2, 4x4	None	None
3, 4	8x8, 16x16	Mild	Possible QoR degradation
5	32x32	Moderate	Likely QoR degradation
6	64x64	High	Difficulty routing
7,8	128x128, 256x256	Impossible	Likely unroutable



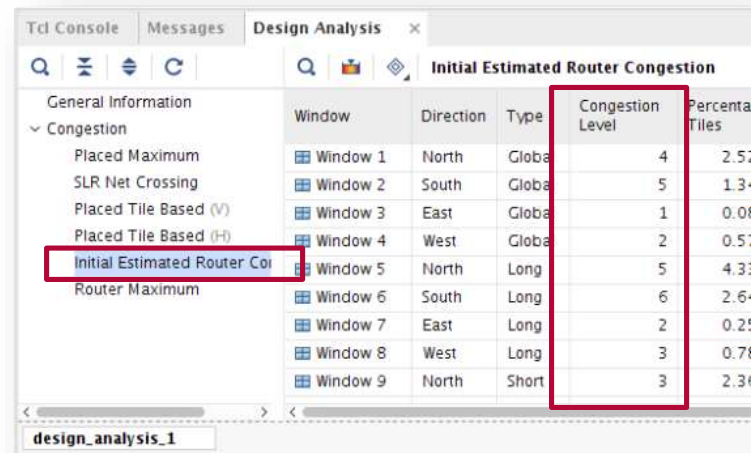
拥塞类型

- Global: 布线资源利用率高的区域
 - 与供应相比, 进出该区域的连接需求较高
 - 常见原因: 高度 LUT 合并、控制集过多、大量总线、布局规划较差
 - 更详细的模型通常有利于 UltraScale和UltraScale+ 的分析: [Short](#) 和 [Long](#)
- Short: 缺少短距离布线: SINGLE、DOUBLE、QUAD
 - 在 UltraScale 中更常见
 - 常见原因: MUXF 或 CARRY 原语的高密度集中
- Long: 缺少长距离布线: HLONG、VLONG
 - 在 UltraScale+ 中更常见
 - 常见原因: 模块的 Rent/平均扇出较高, 大型块的高利用率(强连接性), 过多 SLR 交叉网络

拥塞分析：建议方法

警告：[Route 35-447] 拥塞正阻止布线器对所有网络进行布线。

- 检查布线器警告消息
- 运行报告设计分析（IDE）或 `report_design_analysis -congestion`
 - 关注初始估算布线器拥塞
 - 最详细的报告：Global、Long 和 Short 拥塞
 - 在初始布线后布线器所见的拥塞图片
 - 纳入有效更改的最佳时机
 - 布局报告确认拥塞，但不够精确
 - Router Maximum 表明拥塞是否解决
- 检查初始估算布线器拥塞等级
 - 4 级及以下 - 通常没有问题
 - 5 级 - 有一定难度，但如果是隔离区域可以解决
 - 6 级 - 布线难度极大，影响编译时间和 Fmax
 - 7 级及以上 - 通常不可能



General Information	Window	Direction	Type	Congestion Level	Percentage of Tiles
Placed Maximum	Window 1	North	Global	4	2.5%
SLR Net Crossing	Window 2	South	Global	5	1.3%
Placed Tile Based (V)	Window 3	East	Global	1	0.0%
Placed Tile Based (H)	Window 4	West	Global	2	0.5%
Initial Estimated Router Congestion	Window 5	North	Long	5	4.3%
Router Maximum	Window 6	South	Long	6	2.6%
	Window 7	East	Long	2	0.2%
	Window 8	West	Long	3	0.7%
	Window 9	North	Short	3	2.3%

针对 UltraScale 和 UltraScale+ 的拥塞分析

- 找出造成拥塞区域的层级单元

Window	Direction	Type	Congestion Level	Percentage Tiles	Cell Names			Combined LUTs	LUT6	LUT5	Flop	MUXF
					Top Cell 1	Top Cell 2	Top Cell 3					
Window 1	North	Global	4	2.526%	sub_top_1_7 (95%)			54%	13%	0%	64%	0%
Window 2	South	Global	5	1.340%	sub_top_1_2 (79%)			50%	10%	0%	77%	0%
Window 3	East	Global	1	0.085%	sub_top_1_2 (78%)	sub_top_1_3 (14%)		70%	11%	1%	92%	0%
Window 4	West	Global	2	0.577%	sub_top_1_0 (55%)	sub_top_1_12 (42%)		94%	5%	0%	67%	0%
Window 5	North	Long	5	4.336%	sub_top_1_7 (51%)	sub_top_1_4 (25%)	sub_top_1_8 (18%)	49%	10%	0%	58%	0%
Window 6	South	Long	6	2.647%	sub_top_1_12 (45%)	sub_top_1_2 (38%)	sub_top_1_11 (8%)	48%	10%	0%	67%	0%
Window 7	East	Long	2	0.257%	sub_top_1_2 (80%)	sub_top_1_3 (15%)		78%	21%	0%	88%	0%
Window 8	West	Long	3	0.789%	sub_top_1_8 (19%)	sub_top_1_4 (18%)	sub_top_1_7 (17%)	0%	0%	0%	71%	0%
Window 9	North	Short	3	2.366%	sub_top_1_3 (86%)			61%	14%	0%	91%	0%

- 检查这些单元中的拥塞“症状”
 - `report_design_analysis -congestion`: 检查利用率列
 - `report_design_analysis -complexity`: 使用 `-hierarchical_depth` 报告单元复杂性标准
 - `report_qor_suggestions`: 评估分析结果并在问题区域应用建议方案
 - `report_utilization [-cells]`: 检查问题单元的利用率

导致拥塞的罪魁祸首

- 拥塞区域可能因网表结构形成
 - 大量使用 MUXF 会导致连接短缺
 - 长 CARRY 链引起密集逻辑和连接
 - 大量低扇出控制信号（复位、使能）消耗布线资源
 - 过量的复杂逻辑 cone（高 Rent 指数）
- 对造成拥塞的层级单元进行重新综合
 - 注意：全局使用策略和选项，可能会对时序造成更广泛的影响
 - 使用块级综合策略以生成更合适的网表
 - 无关联（OOC）流程可用于尚未由块级综合处理过的解决方案

拥塞解决方案 (1/2)

- 禁用问题实例上的 LUT 组合

- 在实现的最初阶段使用空值覆盖现有的 LUT 属性

```
set_property HLUTNM "" get_cells -hier -filter {ref_name =~ LUT* && name =~ */inst/*}
```

- MUXF 和 CARRY 原语的高利用率可能会导致拥塞

- 使用 `opt_design` 将 MUXF 和 CARRY 原语重新映射到 LUT
- 全局选项 `-muxf_remap` 和 `-carry_remap`: 可能会影响时序, 要谨慎使用
- 使用单元属性 `MUXF_REMAP` 和 `CARRY_REMAP` 关注拥塞区域内的单元

```
set property MUXF_REMAP 1 [get_cells inst_a]; # opt_design only remaps MUXF cells in inst_a
```

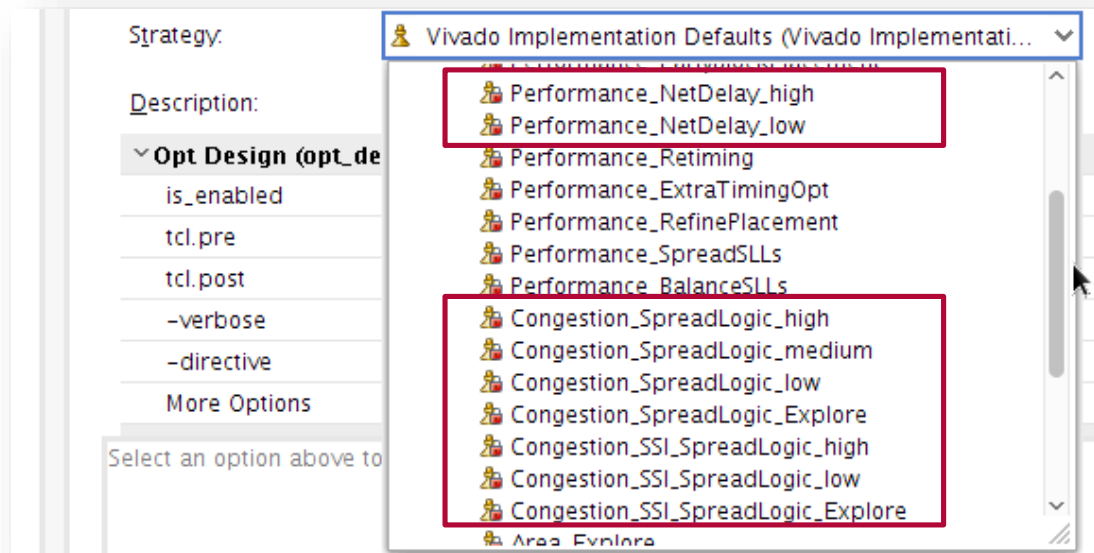
- 进一步降低逻辑层次以最大程度减小时序影响

- `opt_design -remap`: 影响整个设计
- 可以使用 `DONT_TOUCH` 进行精准优化

```
set orig_dont_touch_cells [get_cells -hier -filter DONT_TOUCH] ; # remember which cells have DONT_TOUCH
set DONT_TOUCH 1 <all cells except those to be remapped> ; # isolate the cells for optimization
opt_design -remap ; # only optimize the isolated cells
set DONT_TOUCH 0 [get_cells -hier] ; # removed DONT_TOUCH from all cells
set DONT_TOUCH 1 $orig_dont_touch_cells ; # restore original DONT_TOUCHed cells
```

拥塞解决方案 (2/2)

- 建议实现策略



UltraScale+: [NetDelay*](#) 是不错的选择
更少的 Short 拥塞，更多的 Long 拥塞

UltraScale: [SpreadLogic*](#) 是较好的选择

请参阅 UG904 附录 C，查看用于
每个策略的指令

report_qor_suggestions (RQS)

全新的
的

- 分析设计中的 QoR 问题并推荐解决方案
 - 可以在任意设计阶段运行
 - 报告有悖于不同 QoR 检查的设计问题
 - 针对每个设计问题推荐要采取的行动
 - 提供包含建议修复措施的交钥匙 XDC 和 Tcl 文件
- 下一个阶段朝自动时序收敛解决方案方向发展
 - 当前格式：包含详细描述和补充文件的文本报告
 - 未来版本：交互式 GUI 报告
 - 最终目标：设计收敛向导

总结

减少时序收敛迭代

- 高速设计提出很多挑战
 - 分析 → 根本原因 → 解决方案
- 使用 Vivado 强大的分析功能
 - 尽早分析并修复设计问题
 - 运行、检查并修复所有方法检测
 - 检查逻辑层次、控制集和高扇出网络
 - 依照指南缓解拥塞
- 强大的优化技术
 - 使用高级综合与实现指令及策略
 - PhysOpt - 布局后、布线后

- report_timing_summary
- check_report_clocks (Note: report)
- report_clock_networks (IC1-only)
- report_clock_interaction
- report_cdc
- report_methodology
- report_design_analysis
- report_control_sets
- report_high_fanout_nets

附加资源

➤ UG 949

➤ UG 903

➤ UG 906

➤ UG 1231

现在均可在资源工具集中找到!

➤ [Vivado Design Suite 的 UltraFast 设计方法 - 简介与概览](#)

➤ [Quick Take 视频 - 约束爆炸](#)

➤ [时序收敛的 Vivado UltraFast 设计方法](#)

➤ china.xilinx.com/vivado

Chinese



English

