UltraFast 设计方法
快捷参考指南 (UG1231)

引言

UltraFast 设计方法是由赛灵思推荐的一套最佳实践，旨在最大限度提升生产力，并减少复杂系统的设计迭代，面向系统包括嵌入式处理器、模拟与数字化处理、高速连接、以及网络处理。如欲了解详情，请参阅《UltraFast 设计方法指南（适用于 Vivado Design Suite）》 UG949)。

UltraFast 设计方法检查表 (XTP301) 包含了所有常见问题，重点关注设计在下游能够产生影响的方方面面，并介绍了通常被忽略或未知的潜在问题。它能够帮助您轻松访问相关辅助材料。该检查表已经纳入 Xilinx Documentation Navigator 工具 (DocNav) 中。

本快速参考指南重点介绍了关键的设计方法步骤，有助于更快地实现系统集成与设计实现，从而利用赛灵思器件与工具创造最大限度的价值。同时也提供了相关辅助材料的提示。本指南中涵盖的主要设计任务包括：

- 主板和器件规划
- 设计输入和设计实现
- 顶层设计验证
- 设计分析
- 设计收敛

请访问 Xilinx Documentation Navigator 工具 (DocNav)，参阅《UltraFast 设计方法》中的系统级设计流程部分，了解所有设计中心与特定辅助材料的相关提示。

主板和器件规划

RCB 设计人员

审核关键接口
- 验证部件安置与关键接口

审核 PCB 布局
- 进行存储器接口与收发器检查表工作
- 遵循 PCB 布局建议
- 确保最终 FPGA 管脚由 FPGA 设计人员签发

审核原理图
- 完成 PCB 检查表审核
- 检查 PDS、配置，以及电源
- 在配置之前、开展配置时、配置后验证 I/O 状态

制造并测试
- 确认测试 I/O 项目的配置顺序、电源与 I/O 性能

另请参阅：
UG949: 主板和器件规划
PCB 设计检查表
存储器接口 IP 设计检查表
原理图设计检查表

FPGA 设计人员

针对管脚分析器件
- 检查收发器与结合 I/O 的位置
- 检查 SSI 技术 I/O 管脚分配
- 验证部件安置与关键接口

定义关键接口的 I/O 管脚
- 创建 I/O 管脚分配项目
- 定义并验证存储器控制器、GT 以及 PCIe 技术位置
- 确立时钟框架
- 最大限度地减少连接 IP 间的布局规划距离

定义最终管脚
- 将接口项目融合为一个最终 I/O 项目
- 验证 DRC 与 SSN 分析
- 实现设计以检查时钟与 I/O 规则
- 采用最终 I/O 项目开展生产测试

估算功耗
- 采用 Xilinx Power Estimator (XPE) 确立功耗预算与热裕量
- 采用此前设计知识应用反转率

另请参阅：
UG949: 主板和器件规划
功率估算和优化设计中心
I/O 和时钟规划设计中心

设计输入和设计实现

逻辑设计人员

定义良好的设计层
- 定义相关层次结构，帮助开展全局布局与布局规划
- 在上层附近插入 I/O 与时钟组件
- 在主要层次结构边界上加入寄存器
- 生成 IP 并审核目标器件利用率

构建并验证 RTL 子模块
- 确保设计符合 RTL 编程指导方针
- 在 DSP 与存储器周围添加足够的寄存器
- 只在必要时使用控制信号
- 采用综合属性控制最终逻辑映射
- 创建简单时序约束，审核估算时序与逻辑层过多的地址路径
- 审核综合日志文件、利用率报告，以及细化视图，找出非理想的映射
- 运行方法与 RTL 检查，审核问题
- 在 out-of-context (OOC) 模式下实现子模块，验证实现性能
- 对比原预算审核利用率与功耗
- 开展设计仿真，验证功能性

组装并验证顶层设计
- 综合顶层 RTL 设计并解决所有连接问题
- 审核顶层利用率与时钟指导方针
- 创建并验证顶层约束
- 迭代 RTL 与约束，解决方法与 DRC 问题并符合时序
- 开展实现

另请参阅：
UG949: 设计创建和设计实现
使用 Vivado IP 设计中心进行设计
使用 Vivado IP 集成器设计中心
逻辑综合设计中心
应用设计约束设计中心
设计实现设计中心

XILINX

UG1231 (v2018.2) 2018 年 7 月 27 日

UltraFast 设计方法
快捷参考指南 (UG1231)

顶层约束验证

开展设计基线

- 在设计流程早期，大多数块与关键 IP 可用的情况下，验证时序收敛可行性
- 指定必要约束仅：
  - 使用全部 IP 约束
  - 定义符合现实情况的主时钟与生成时钟
  - 定义全部跨时钟域约束
  - 在需要的位置添加多周期路径
  - 在该阶段切勿使用 I/O 约束
- 确保路径要求的合理性
- 在流程每个阶段使用 report_timing_summary 验证 WNS ≈ 0.0 ns：
  - 综合后
  - 布局前
  - 布线前后
- 在流程早期解决时序违规
- 在 RTL 和综合中解决 QoR 问题，实现最大成效

验证时序约束

- 运行 report_timing_summary 或 check_timing 以确保全部时钟得到定义以及全部寄存器、输入端口、输出端口得到约束
- 运行 report_methodology 并解决全部 TIMING* 与 XDC* 问题
- 运行 report_clock_interaction 以确保每个时钟对采用合理的路径要求进行了安全的定时
- 运行 report_cdc 以确认所有异步跨时钟域路径得到妥善约束并使用了安全的同步电路
- 运行 report_exceptions 以找出交覆的时序例外已忽略，或不足的现象
- 确保设计加载与约束应用时全部关键告警得到了解决

另请参阅：

UG949: 设计收敛
- 检查设计是否正确约束
- 设计基准 (baseline)

应用设计约束设计中心

时序收敛与设计分析设计中心

设计分析和收敛

找出时序违规问题根源

- 用 report_qor_suggestions 进行自动分析和时序收敛建议
- 采用 report_timing_summary 或 report_design_analysis 找出问题根源
- 针对设置路径，检查下列原因造成的高数据路径延迟：
  - 大型单元延迟 (7 系列 > 25%, UltraScale 器件 > 50%)
  - 大型网络延迟 (7 系列 > 75%, UltraScale 器件 > 50%)
- 针对保持路径，检查正保持要求大于 0 ns
- 检查高度时钟偏差 (> 500 ps)、高度时钟不确定性 (> 200 ps)，或二者皆有的情况

减少逻辑延迟

- 修改 RTL，使用并行运算符或高效运算符
- 添加流水线寄存器并使用综合重定时
- 为 DSP 或模块 RAM 输出添加寄存器
- 用LUT_REMAP把最长（时序）路径中的小 LUT 合并起来。
- 针对 SRL 输入、输出，或输入与输出将寄存器移出 SRL
- 去除 KEEP/DONT_TOUCH/MARK_DEBUG

减少网络延迟

- 审核并调整布局规划约束
- 优化高扇出网络
- 如下列报告中出现 level > 4，解决拥塞问题：
  - report_design_analysis 布局器拥塞表
  - 日志文件中的初始估算路由拥塞

减少时钟偏差

- 使用并行缓存，而不是级联缓存
- 在来自同一输入或 PLL 的同步时钟间使用 CLOCK_DELAY_GROUP
- 在异步时钟间添加时序例外

减少时钟不确定性

- 优化 MMCM 设置
- 采用 BUFGCE_DIV 在 UltraScale 器件中分开时钟

另请参阅：

UG949: 设计收敛
- 了解时序报告
- 确定时序违规的根源

时序收敛和设计分析设计中心

减少控制集

- 在控制信号中避免出现 MAX_FANOUT
- 提高综合控制集阈值
- 使用 opt_design 融合对应信号

优化高度扇出网络

- 在 RTL 中使用基于层次结构的寄存器复制
- 用 opt_design -merge_equivalent_drivers -hier_fanout_limit 提升复制效率
- 如非关键，则提升至全局时钟
- 通过 phys_opt_design 强制复制

解决拥塞

- 降低期间利用率并平衡 SLR 利用率
- 尝试使用布局器指令 AltSpreadLogic* 或 SSI_Spread*
- 通过 report_design_analysis -complexity -congestion 找出拥塞模块
- 针对拥塞模块，尝试使用 AlternateRoutability 块级综合策略或使用 opt_design 减少 MUXF*/CARRY*、或使用 CELL_BLOAT_FACTOR 属性
- 拥塞区域中用全局时钟提供高扇出网络
- 重复使用此前低拥塞实现中的 DSP 和块 RAM 布局

调整编译流程

- 尝试使用多个 place_design 指令
- 针对最优网表文件使用模块级综合策略
- 在布局与物理优化阶段通过 set_clock_uncertainty 对关键时钟进行过度约束
- 在细小设计修改后采用增量编译来维持 QoR 并改善运行时间

分析并优化功耗

- 约束活动、环境与进程
- 尝试通过 power_opt 减少功耗
- 最大限度提高块 RAM 级联的使用

另请参阅：

UG949: 实现与设计收敛
- 分析并解决时序违规
- 应用通用时序收敛技术

实现设计中心

时序收敛与设计分析设计中心