



NPC 的 FPGA 部署经验

ysyx_23060015 叶永檬

NPC-FPGA

- 为什么要将 NPC 部署到 FPGA
 - FPGA 部署流程和 IC 的设计流程有共通之处
 - 比 ASIC 更简化，成本更低
 - 硬件设计中各种要素的折中
 - FPGA 的工具链是最容易接触到的工业级 EDA
 - 利用工业级的 EDA 工具学习更好的 RTL 设计
 - 毕设

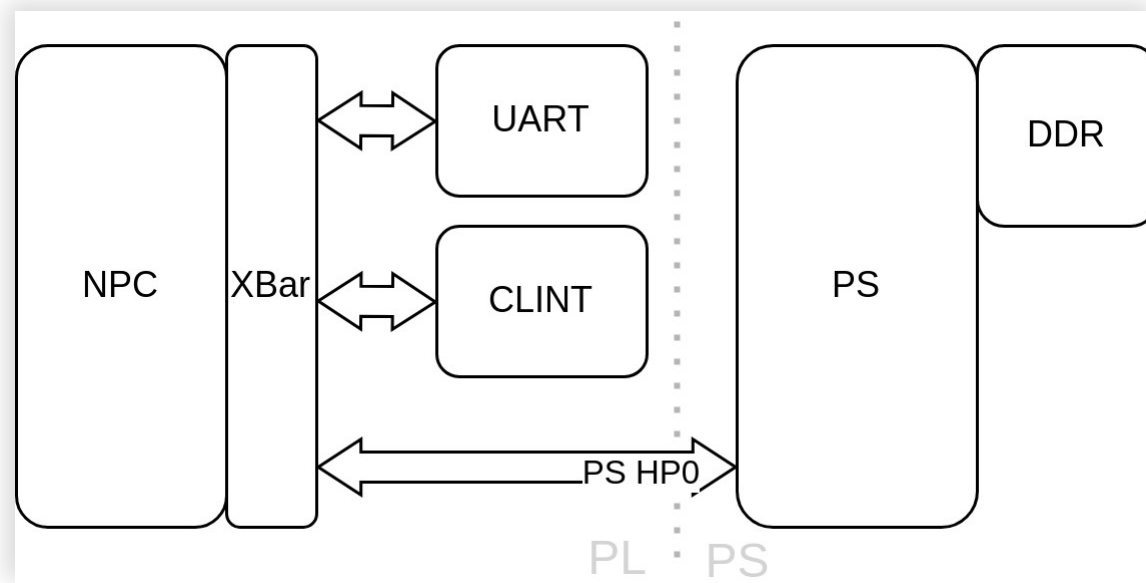
FPGA 流程

ASIC 流程



NPC-FPGA

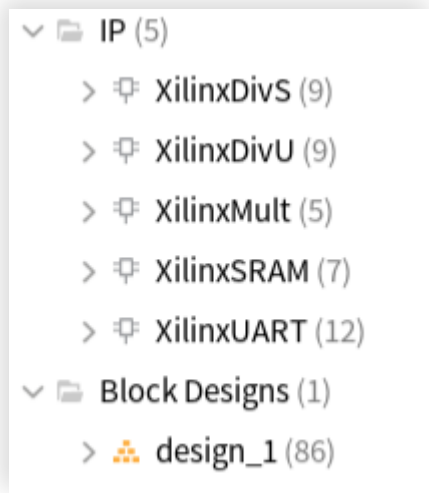
- 设计目标
 - 目标平台： xc7z020clg400-2
 - 工具链版本： 2023.2
 - 最基本的 SoC： Mem， CLINT， UART
 - 50MHz 以上频率
 - 只关心时序性能



ZYNQ 的板载 DDR 连接到 PS 上
PL 访问 DDR 需要由 HP 接口经过 PS
系统上电时先由 PS 将程序装载到 DDR
预告： HP 接口存在 bug

从前仿到可综合 RTL

- 基于 Verilator 的前仿包含一些无法综合的部分
 - DPI-C 访存
 - Verilog 系统函数
 - SRAM 模型
 -
- 你可能还想利用一下 FPGA 的 IP
 - DSP、乘除法器
 -



	前仿	可综合 RTL	一致性目标
内存	AXI+DPI-C	AXI(HP)+DDR	功能一致
Cache 颗粒	SRAM 模型	单端口 BRAM	功能 + 时序一致
串口	AXI+\$dispay	AXI UARTLite IP	功能一致
乘除法器	仿真模型	IP	功能 + 时序一致

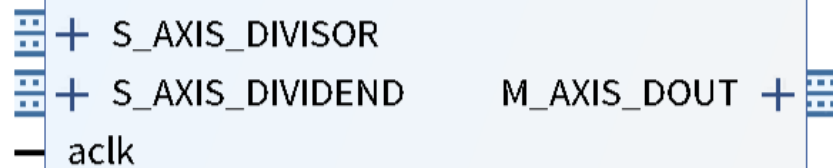
从前仿到可综合 RTL

- 一致性的验证
 - 如何验证?
 - Vivado 仿真
 - 运行哪些程序?
 - cpu-tests
 - riscv-arch-tests
 - 需要验证哪些被替换的模块?
 - AXI 接口的模块:
 - 如果 AXI 协议实现是正确的，那么基本不会有问题
 - Cache 颗粒：
 - 经过设置的 BRAM 的操作时序和 SRAM 是一致的
 - 还需要注意连线关系
 - 乘法器：
 - 固定的延迟周期
 - 除法器：
 - 固定的延迟周期 +AXI-Stream 接口

Information

Memory Type: Single Port Memory
Block RAM resource(s) (18K BRAMs): 0
Block RAM resource(s) (36K BRAMs): 2
Total Port A Read Latency : 1 Clock Cycle(s)
Address Width A: 6

BRAM 的延迟应为 1 周期，与 SRAM 一致
预告：由于 BRAM 的延迟设为 1 周期，
后续会遇到重大问题



除法器的接口是 AXI-Stream
具体请 RTFM

从前仿到可综合 RTL

- Vivado 上仿真没有 Dfftest
- 而且测试程序 HIT BAD TRAP 了!
- 「异步 Dfftest」!
 - 每执行一条指令，就在 Vivado 仿真器中导出日志
 - Rf、仿真时间
 - npc (verilator) 读取导出的日志，执行相同的程序
 - 只不过，这一次 npc 成为了 ref

导出日志

```
reg en = 1;
initial begin
    fp = $fopen("./reg_log", "w");
    #50000 en = 0;
end

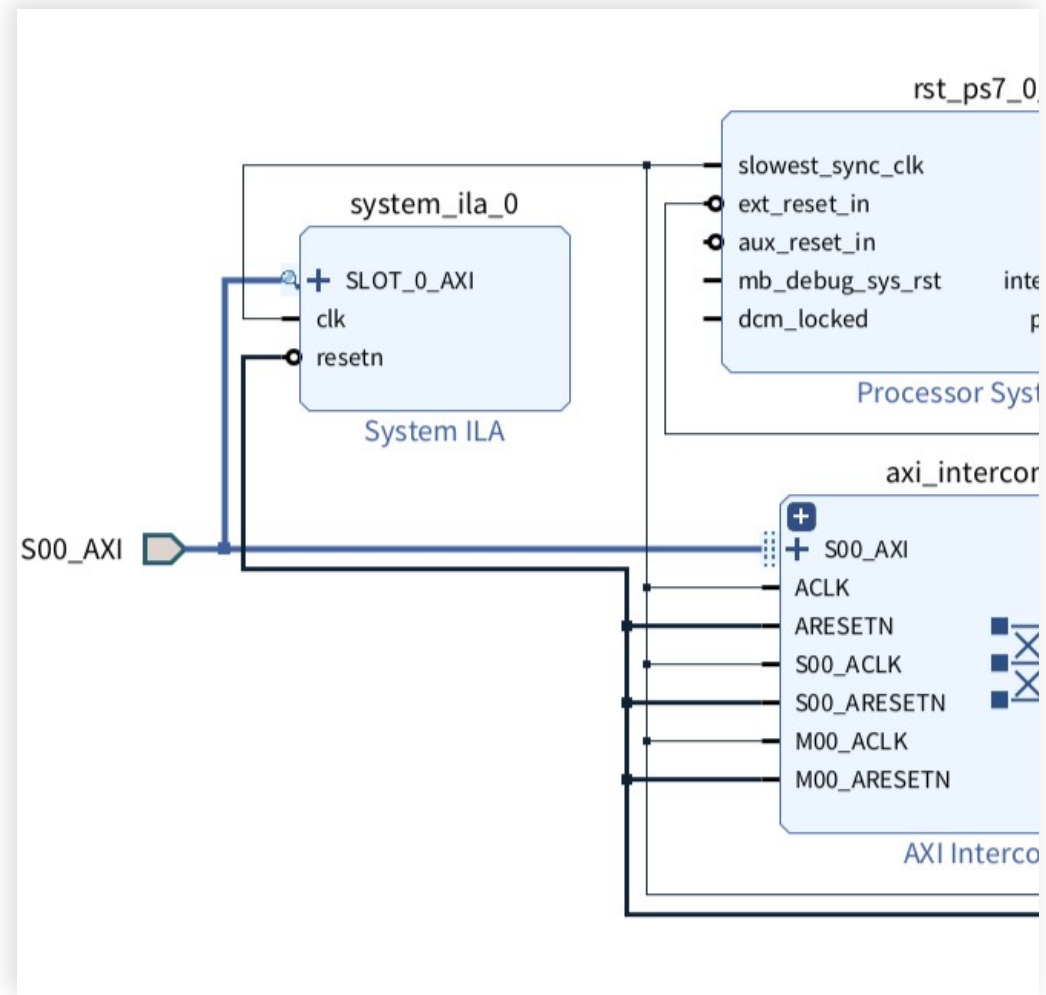
always @(posedge clk) begin
    if (!rst) begin
        if (InstReady & en) begin
            $fwrite(fp, "%d\n", $time);
            #1;
            for (i = 0; i < 32; i = i + 1)
                $fwrite(fp, "%08x\n", Reg[32*i+:32]);
        end
        if (Done | ~en) begin
            $fclose(fp);
        end
    end
end
```

读入日志
异步 Dfftest

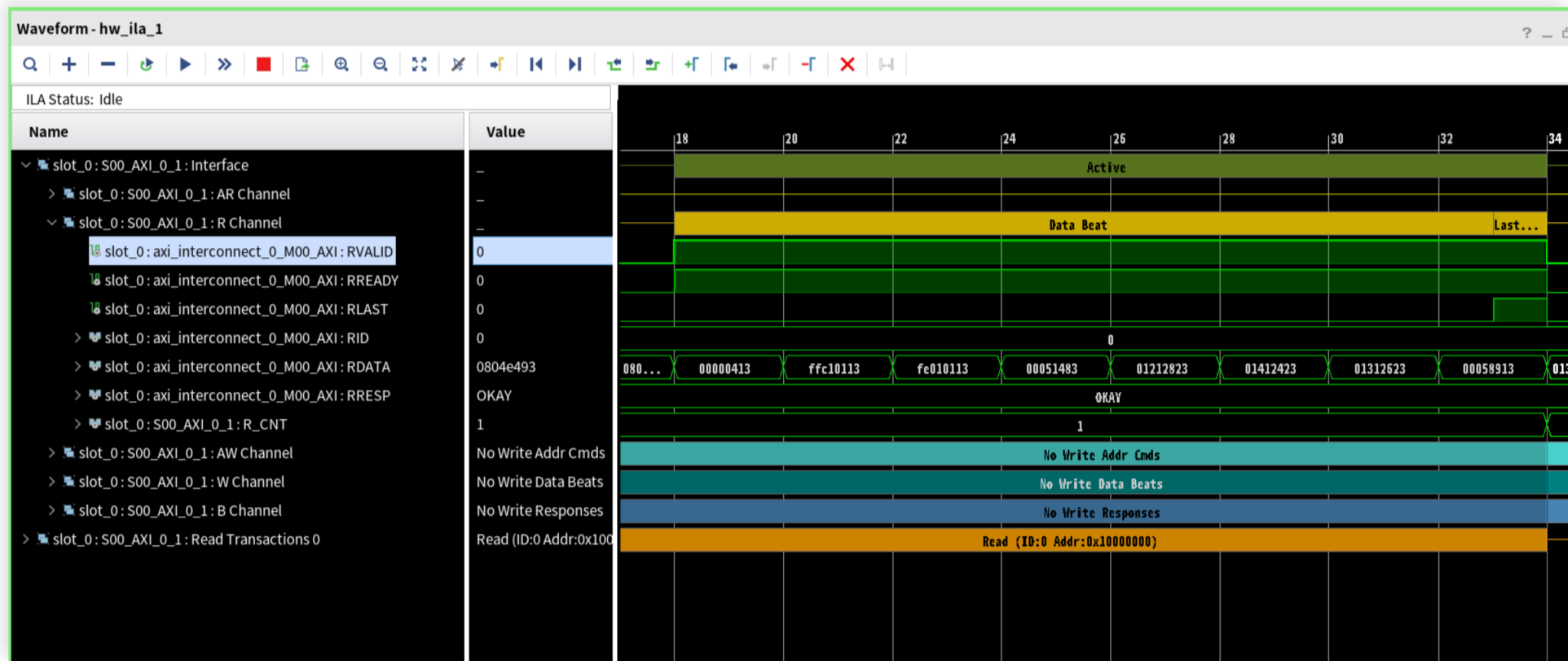
```
#ifdef CONFIG_DIFFTEST_X
    if (diff_x) {
        word_t reg;
        int x_sim_time;
        bool diff_x_reg = true;
        fscanf(diff_x_log, "%d", &x_sim_time);
        for (int j = 0; j < REG_NUM; j++) {
            fscanf(diff_x_log, "%x", &reg);
            if (reg != REG_FILE(j)) {
                diff_x_reg = false;
                printf("x%d(%s) REF(verilator): 0x%08x, DUT(xilinx): 0x%08x\n",
                    j, regs[j], REG_FILE(j), reg);
            }
        }
        if (!diff_x_reg) {
            printf("\033[47;31mDIFFTEST_X FAILED\033[0m\n");
            printf("at xilinx sim time %d\n", x_sim_time);
            ASSERT(diff_x_reg);
        }
    }
#endif
```

案例 1：HP 接口工作不正常

- 案例 1：上板无法工作
- 其他模块都已经通过仿真测试
- HP 接口属于 PS，无法通过仿真验证
- PS 调试发现 DDR 被正确写入
- → 推测是 HP 接口的问题
- 加入 ILA，观察 HP 接口行为

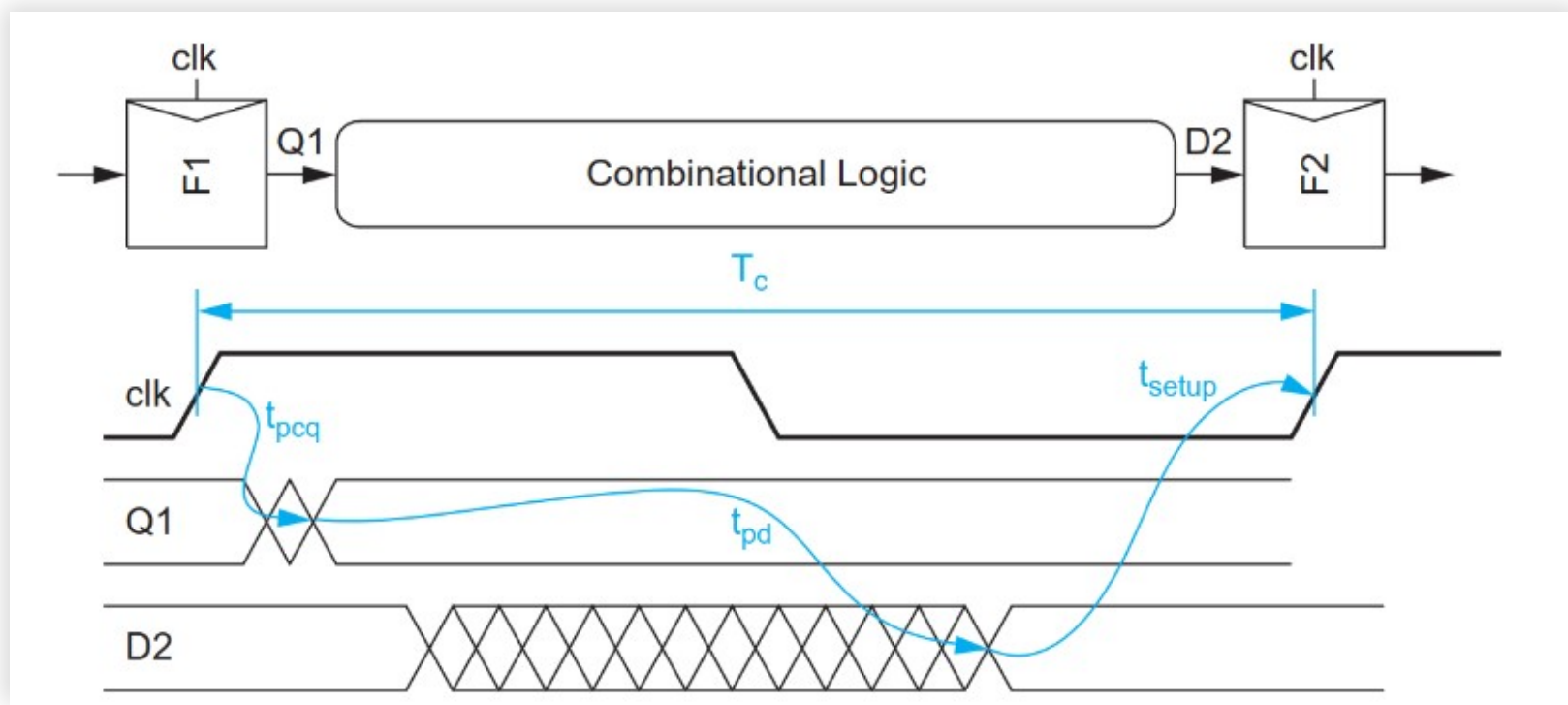


案例 1：HP 接口工作不正常



- ILA 发现 burst 传输的数据是 8 字节对齐的
- 但是 HP0 已经配置成 32 位了，不能理解
- 解决方法：HP0 配置成 64 位，AXI Interconnect 自动进行位宽转换

FPGA 上的时序优化



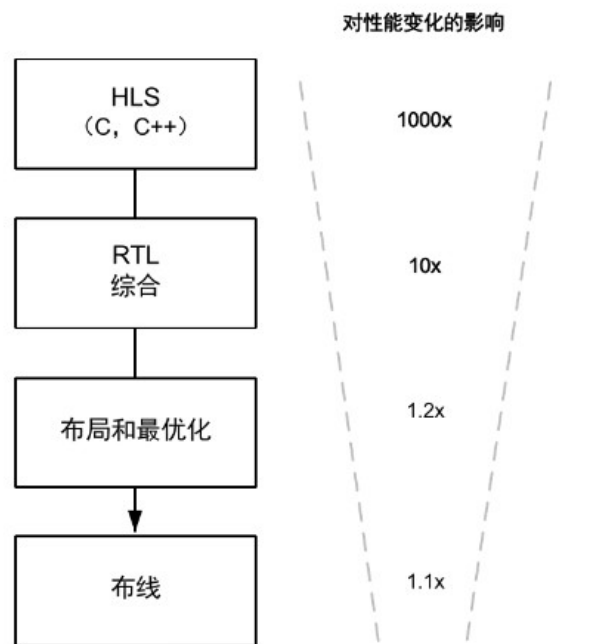
- CMOS VLSI Design A Circuits and Systems Perspective (4th Edition) - FIGURE 10.5
- 最简单的时序约束（假设时钟是理想的）：
- $T_c \geq t_{pcq} + t_{pd} + t_{setup}$
- $Slack = T_c - (t_{pcq} + t_{pd} + t_{setup})$
- 触发器的传播延迟 t_{pcq} 和建立时间 t_{setup} 是已知的
- 设置电路的目标周期 T_c （频率）
- EDA 计算组合逻辑的最大延迟 t_{pd} ，并以此为目标优化

FPGA 上的时序优化

- FPGA 上组合逻辑延迟的两大来源
 - 长组合逻辑链
 - ASIC 设计中可以在关键路径使用更快的器件 -> FPGA 上没有
 - 高扇出
 - ASIC 设计中可以插入缓冲器加强驱动 -> FPGA 上没有
 - ASIC 设计中可以复制逻辑来减少扇出 -> FPGA 上只能复制触发器
- 根本解决方案：改设计

- FPGA 的性能优化，推荐学习 Xilinx UG949
- 包含大量 FPGA 设计优化的工程性建议
 - 虽然是面向 Xilinx FPGA，但有些经验在 ASIC 也是共通的

图 2：整个流程中设计变更的影响

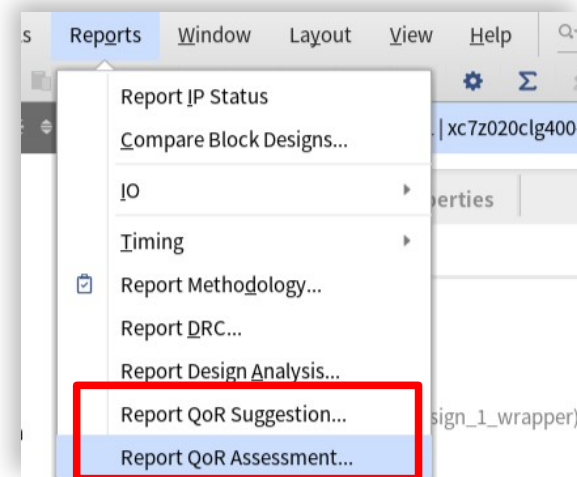


X13423-052822

适用于 FPGA 和 SoC 的
UltraFast 设计方法指南

FPGA 上的时序优化

- 能不能让 EDA 工具评估我的设计？
- 设计质量报告 QoR+ 自动优化建议 RQS

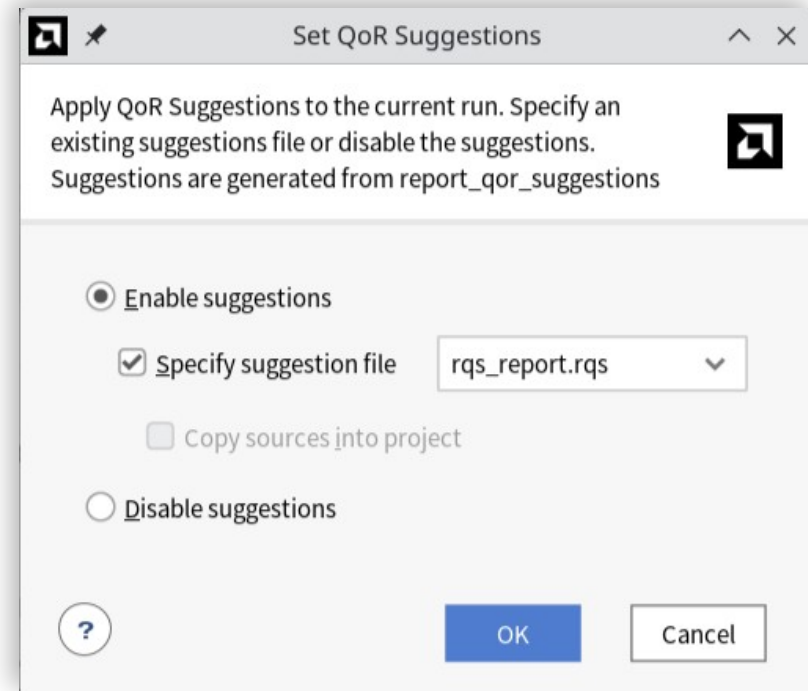


General	QoR Assessment Score	Flow Guidance
QoR Assessment	3 - Design runs have a small chance of success	Run report_qor_suggestions to generate suggestions. They may indirectly improve REVIEW items.
RQA Summary		To see critical timing paths examine the CSV file containing timing paths.
Assessment Details		
ML Strategy Availability		
Challenging Timing Paths		
Clock Skew		
Netlist Objects with DONT_TOL		
Netlist Objects with DONT_		

General	Name	Threshold	Actual	Used	Available	Score	Status
QoR Assessment	Utilization					5.0	
RQA Summary	LUT Combined	<2.00	0.00	0	14239		REVIEW
Assessment Details	Netlist						
ML Strategy Availability	*DONT_TOUCH (cells/nets)	0	96	-	-		REVIEW
Challenging Timing Paths	Clocking					5.0	OK
Clock Skew	Congestion					3.0	
Netlist Objects with DONT_TOL	Predicted Congestion Score	5	3			3.0	REVIEW
Netlist Objects with DONT_	Timing					5.0	OK

FPGA 上的时序优化

- QoR 评分低于 5 时，就可以尝试一下 RQS
- QoR 和 RQS 具体可以参考 UG906 的相关内容
- RQS 可能有惊喜
 - 具体能提升多少因设计而异

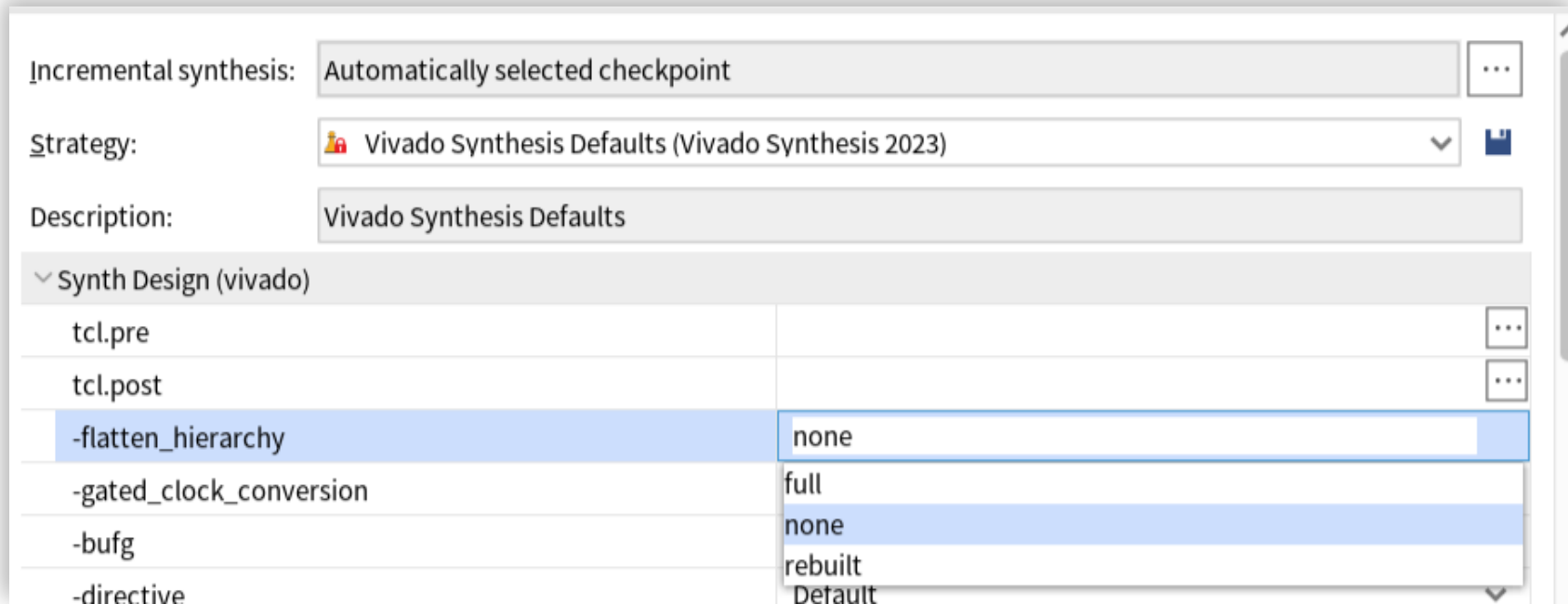


Vivado Design Suite 用户指南

设计分析与收敛技巧

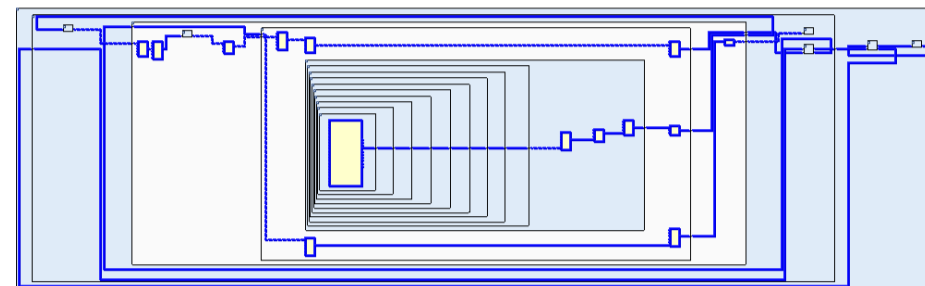
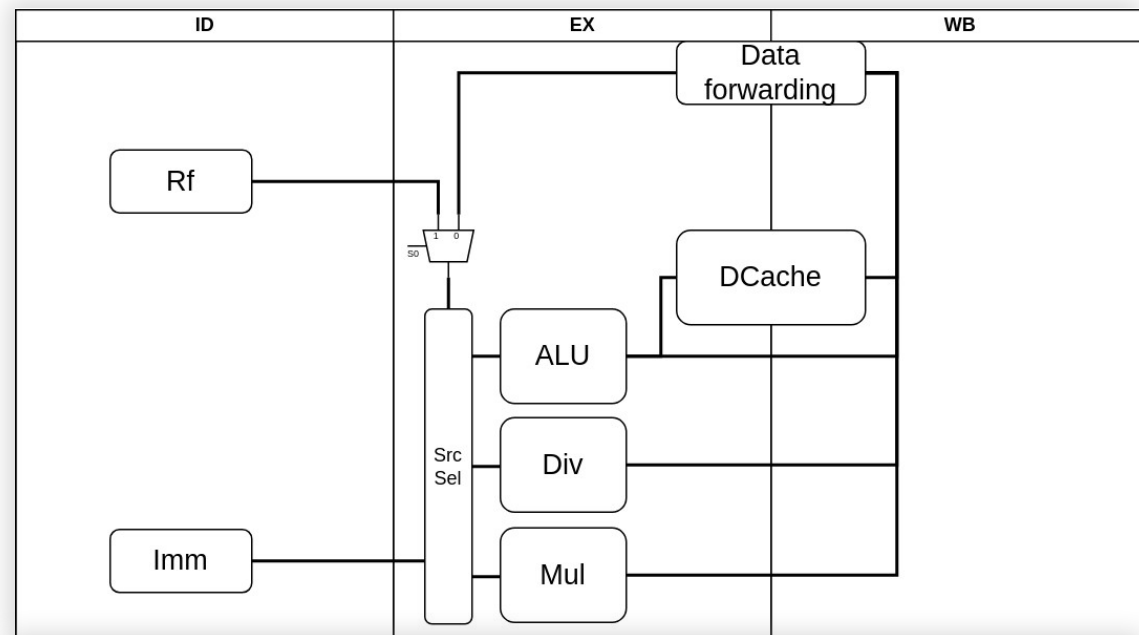
FPGA 上的时序优化

- 分析关键路径前建议关闭综合器的层次展开选项



案例 2：流水线优化

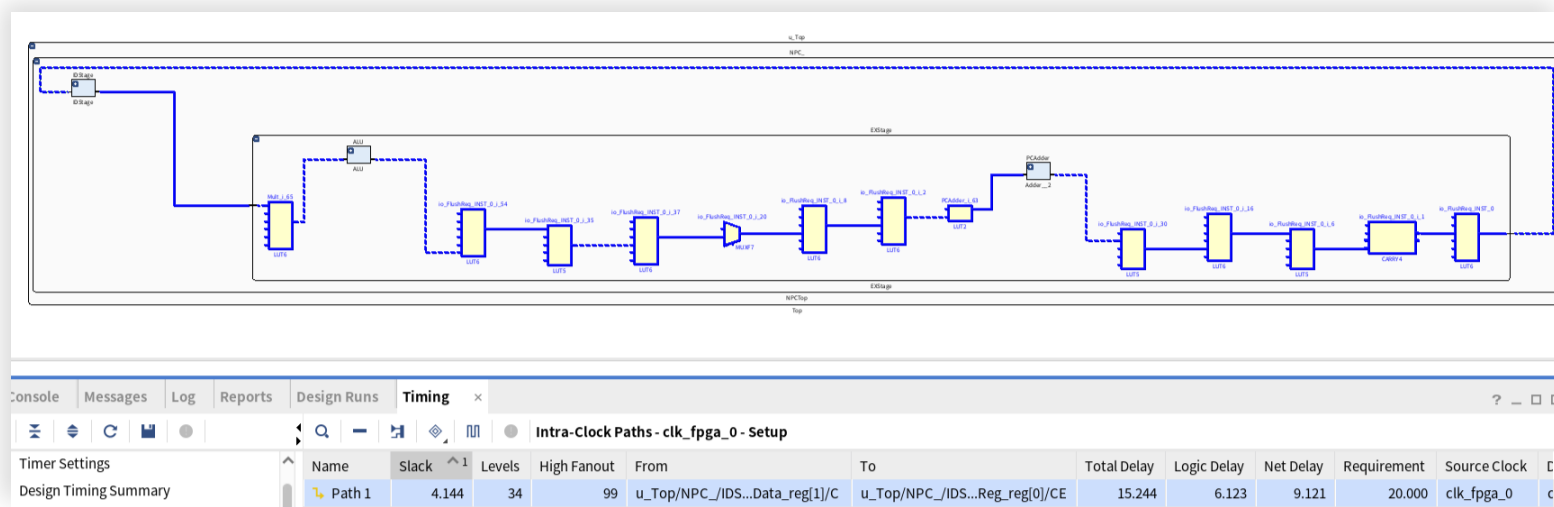
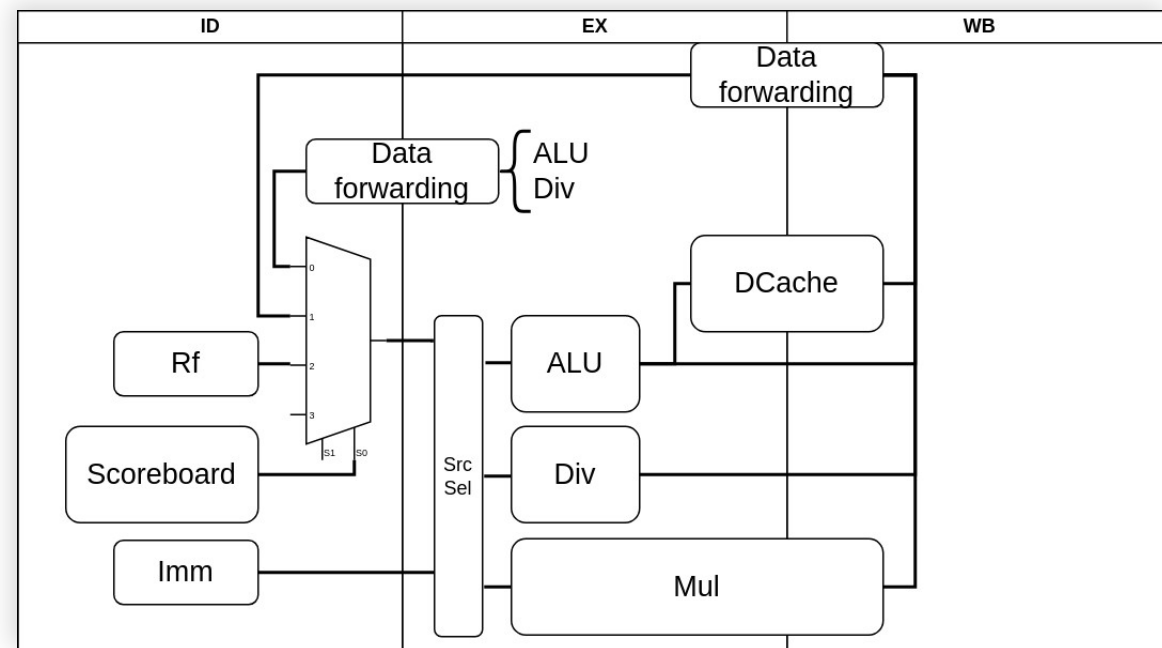
- 某个版本的流水线
- 相当于将 DCache 的 BRAM 作为流水线触发器
- 数据转发：WB 到 EX
- 优点：完全没有 RAW 造成的阻塞
- 时序问题：WNS -1.456
 - BRAM 较长的传播延迟 + EX 的长路径
 - 总线互联桥设计不佳



Design Runs Timing x										
Intra-Clock Paths - clk_fpga_0 - Setup										
Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirer	
Path 1	-1.456	39	302	u_Top/NPC_/E...m/CLKARDCLK	u_Top/NPC_/IDS...Reg_reg[0]/CE	20.844	6.861	13.983	2	

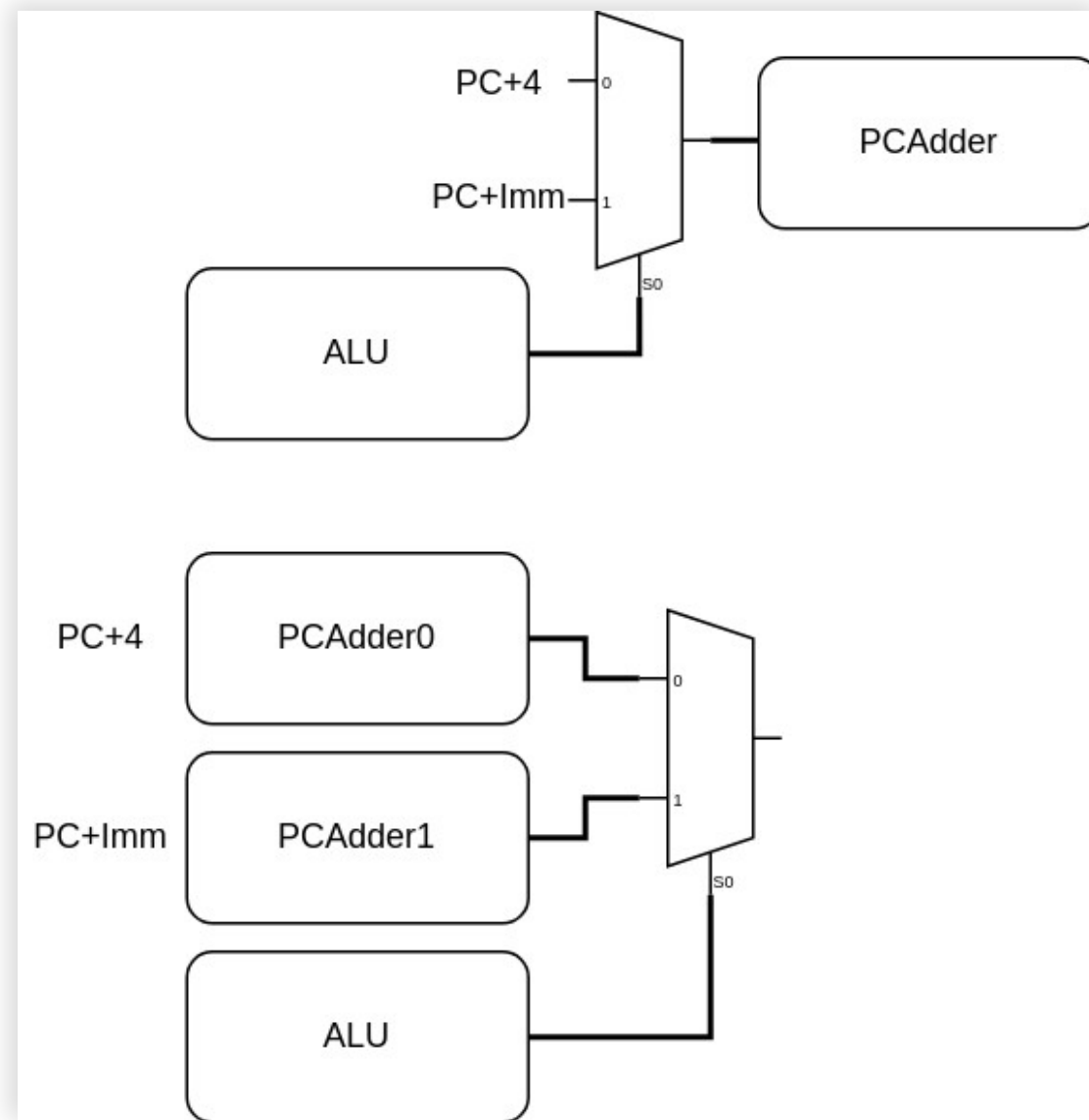
案例 2：流水线优化

- 改进的流水线
 - 数据统一转发到 ID，由 Scoreboard 选择
 - load 指令如果遇到 RAW 会发生阻塞
 - 流水化乘法器
 - 估计 IPC 下降 5%，但频率可以提升更多
- 新流水线的关键路径十分清晰
 - ID->EX-> 反压回 ID
- WNS 4.144



案例 2：流水线优化

- 还能不能更快？
- EX 的关键路径：ALU->PCAdder
 - 根据 ALU 的结果，选择 PCAdder 的输入
- 优化后的关键路径
 - 根据 ALU 的结果，选择 PCAdder 的输出
- 并行逻辑 + 面积换时序



案例 2：流水线优化

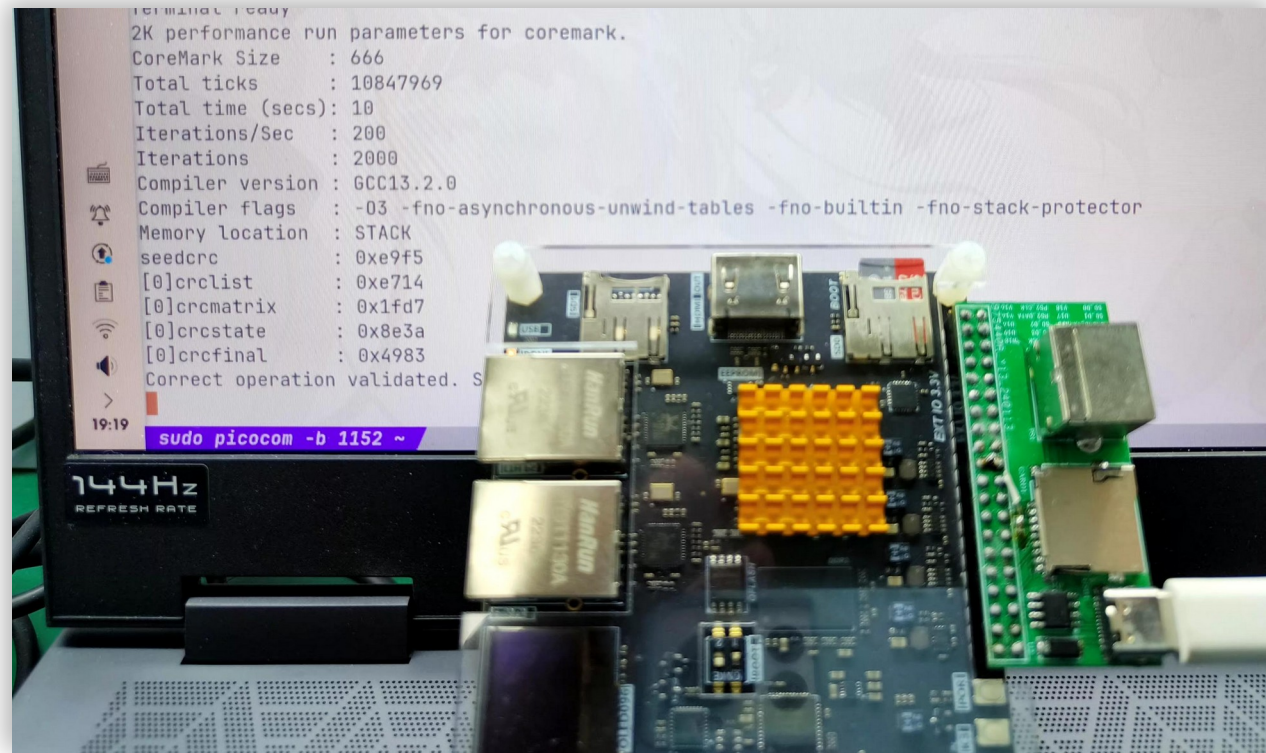
- 还能不能更快?
 - 继续优化设计
 - 缩短关键路径、优化反压……
 - EDA 优化的潜力还没有榨干
 - 在综合设置重新打开展开层次的选项
 - 适当提高约束要求
 - QoR 报告 + 自动建议
 - 布局布线优化
 - ……
- 64MHz !

1. Setup Path Characteristics 1-1

Characteristics	
Requirement	15.625
Path Delay	14.944
Logic Delay	4.441(30%)
Net Delay	10.503(70%)
Clock Skew	-0.141
Slack	0.335
Clock Uncertainty	0.237
Clock Relationship	Safely Timed
Clock Delay Group	Same Clock

总结

- 极简 SoC+ 上板运行 Coremark
- 184 Coremark
- 2.88 Coremark/MHz
- ysyxSoC 上板部署?
- 其他开源 SoC? (比如 LiteX)



从零开始
创造属于你的
RISC-V®处理器 (低配版)



感谢聆听