

## 一 工程项目：

- 1 项目名称：fpga\_training
- 2 项目使用的资源与软件：
  - (1) 正点原子开拓者 FPGA 开发板
  - (2) 示波器（推荐）
  - (3) Quartus II 13.1
  - (4) ModelSim 10.4

## 二 工程目标：

- 1 实验现象：
  - (1) 使用按键控制 led 灯的通断。
  - (2) 使用按键控制 ram 和 fifo 的读写并验证读写是否正确。
- 2 实验内容：
  - (1) 前仿环境下分别使用 Altera IP 核和纯 rtl 代码实现两路频率不同的时钟并进行 ram 读写测试、fifo 读写测试，其中需要验证 pll 失锁时的情况和运行中强制修改 ram 的初始化内容的情况。
  - (2) 在 fpga 平台只需使用 ram 和 fifo 的 Altera IP 核进行实验，对工程进行综合编译，下载并固化 fpga 程序，观察实验现象。
  - (3) 使用 Quartus II 和 ModelSim 进行联合后仿验证。
  - (4) 对比 fpga 综合编译时选择不同编译策略或 seed 的结果，包括资源使用量、时序结果（例如 Fmax）等。
- 3 具体要求：
  - (1) 按键控制 led 时，要求使用三段式状态机。
  - (2) 整个工程分为 5 个大模块，分别为时钟、按键、led、ram 读写、fifo 读写，每个模块要求使用增量编译，且 ram 读写和 fifo 读写两个模块要求使用 lock region 并在 chip planner 指定一块足够大的区域。
  - (3) 对所有出现的时钟进行约束，找出相互独立的时钟设置 group，对复位信号设置 false path。
  - (4) 使用 stp 查看 ram 读写和 fifo 读写的内部信号是否正确。
  - (5) 使用 SingalProbe 将时钟信号和按键信号连接至 fpga 管脚上进行测量。
  - (6) 在 chip planner 里面进行 eco 操作，改变 pll 器件输出的占空比并将其中的某些信号（例如时钟信号）连接至 fpga 管脚上进行测量。

## 三 学习目标（下述内容非先后顺序）：

- 1 学习 verilog 编程规则：
  - (1) verilog 基本语法规则以及相关高级的用法，包括 genvar、generate、for、function、define、parameter、localparam 等关键词用法。
  - (2) 三段式状态机的结构与用法。
  - (3) 模块化编程的思想。
- 2 学习常见 ip 核与基本逻辑功能模块的 rtl 实现：
  - (1) Altera 的 pll IP 核的使用以及失锁处理，以及时钟分频的简单 rtl 实现。
  - (2) Altera 的 fifo IP 核的使用，以及 fifo 的简单 rtl 实现。
  - (3) Altera 的 ram IP 核的使用以及初始化文件的使用，以及 ram 的简单 rtl 实现。

- (4) 边沿检测的 rtl 实现。
- (5) 信号同步的 rtl 实现。
- (6) 二进制码和格雷码转换的 rtl 实现。

### 3 学习 verilog 仿真方法:

- (1) verilog 仿真语法规则以及相关高级的用法, 包括 force、release、wait、readmemh、finish 等关键词用法。
- (2) Testbench 的编写。
- (3) 激励的编写, 包括仿真时钟信号的产生、复位信号的产生、mif 文件的产生。
- (4) 异常情形仿真的处理思路, 例如 pll 失锁状态仿真 (使用 force)、中途修改存储器内部初始化内容 (使用 readmemh) 等。

### 4 学习 ModelSim 软件使用方法:

- (1) 单独使用 ModelSim 对工程进行编译、前仿。
- (2) 分别对 Altera IP 核和 rtl 逻辑实现进行仿真。
- (3) 查看指定信号的波形。
- (4) 根据波形追踪信号。

### 5 学习 Quartus II 软件使用方法:

- (1) 工程的建立与配置, 引脚分配。
- (2) 工程的编译流程, 包括分析综合、布局布线、生成 sof 文件、时序分析等。
- (3) 使用不同策略、seed 进行编译综合, 对比不同编译策略下的情况。
- (4) 与 ModelSim 软件进行联合后仿。
- (5) 生成的 sof 文件转化成 jic 文件进行 fpga 程序固化。
- (6) 增量编译用法。
- (7) lock region 用法。
- (8) 使用 technology map 查看网表追踪信号。
- (9) 使用 chip planner 查看器件内部使用情况, 了解 fpga 内部器件结构与排布, 并进行 eco 操作将时钟信号手动拉线至管脚。
- (10) 使用 STP 查看内部信号, 设置相关触发。
- (11) 使用 SingalProbe 引出内部信号进行 eco 操作。
- (12) 生成和使用 TCL 脚本。

### 6 学习静态时序分析方法:

- (1) 使用 TimeQuest 软件进行时序分析, 并阅读时序报告和查看最差路径。
- (2) 编写 sdc 时序约束文件, 使用 clock、generate\_clock、false\_path、group 等对时序进行约束。

## 四 可以重复利用的资源:

- 1 工程项目目录下 Library 下面的内容, 包括 Altera 仿真库、基本功能逻辑 RTL 的实现。
- 2 工程项目目录下 User\Verilog\clock\_tree.v, 这里处理了 PLL 失锁时的必要操作保证整体系统能够正常稳定地运行。
- 3 其他内容无实际项目参考意义, 仅供学习而设计。

## 五 注意事项:

- 1 ModelSim 软件存在 define 的 bug, 每一次运行前仿的时候请检查 define 是否正确。方法是在控制台输入 vlog, 弹出的对话框内粘贴对应的 define 信息, 具体在

仿真文件夹里面的 **Script** 文件夹里面找对应的.f 文件。

2 小编时间仓促，因此项目中有不完善的地方敬请大家提出建议。

3 学习交流群 1126635164。