Vivado调试工具原理及实现

主 讲：何宾

Email：hebin@mail.buct.edu.cn

2018.08

设计调试的原理和方法

- 对FPGA的调试，是一个反复迭代，直到满足设计功能和设计时序的过程。
  - 对于FPGA这样比较复杂数字系统的调试，就是将其分解成一个个很小的部分。
  - 然后，通过仿真或者调试，对设计中的每个很小部分进行验证。

这样，要比在一个复杂设计完成后，再进行仿真或者调试效率要高得多。

设计调试原理和方法

可以通过下面的方法保证设计的正确性

- RTL级的设计仿真；
- 实现后的设计仿真；

前两个在课程前面已经做过详细的介绍

- 系统内调试；

设计调试原理和方法
--系统内调试

Vivado集成开发环境中包含逻辑分析特性，使得设计者可以对实现后的FPGA器件进行系统内调试。

- 系统内调试可以在真正的系统环境下，以系统要求的速度，调试设计的时序准确性和实现后的设计。
- 与使用仿真模型相比，系统内调试降低了调试信号的可视性，潜在延长了设计/实现/调试迭代的时间。

注: (1)迭代时间取决于设计的规模和复杂度；

（2）Vivado工具提供了不同的调试设计；

设计调试原理和方法
--串行I/O设计调试

为了实现系统内对串行I/O的验证和调试，Vivado集成开发环境包含一个串行的I/O分析特性。

- 设计者可以在基于FPGA的系统中，测量并优化高速串行I/O连接。
- 与使用外部测量仪器相比，使用Vivado内提供的串行I/O分析仪使得设计者可以测量接收器对接收信号进行均衡后的信号质量。

设计调试原理和方法
--系统内调试

包括三个重要的阶段

- 探测阶段
  - ✓标识设计中需要探测的信号，以及探测方法；
- 实现阶段
  - ✓实现设计，包括将额外的调试IP连接到被探测的网络；
- 分析阶段
  - ✓通过与设计中的调试IP交互，调试和验证功能问题；

设计调试原理和方法
--调试策略

调试目标	推荐的调试编程流程
在HDL源代码中识别调试信号，同时保留灵活性，用于流程后面使能或者禁止调试	（1）在HDL中，使用mark_debug属性标记需要调试的信号； （2）使用Set up Debug向导来引导设计者通过网表插入探测流程
在综合后的设计网表中识别调试网络，不需要修改HDL源代码	（1）使用Mark Debug，通过右键单击菜单选项，选择在综合设计的网表中需要调试的网络。 （2）使用Set up Debug向导来引导设计者使用网表插入探测流程。
使用Tcl命令，自动调试探测流程	（1）使用set_property Tcl命令，在调试网络上设置mark_debug属性。 （2）使用网表插入探测流程Tcl命令，创建调试核，并将其连接到调试网络。
在HDL语言中，显式将信号添加到ILA调试核中	（1）识别用于调试的HDL信号。 （2）使用HDL例化探测流程产生和例化一个集成逻辑分析仪（ILA）核，并且将它连接到设计中的调试信号。