Vivado从此开始（To Learn Vivado From Here）

Vivado
从此开始
To Learn Vivado From Here
高亚军 编著
·内容翔实全面：涵盖Vivado所有基本功能
·讲解深入浅出：结合大量案例，帮助读者加强对基本概念的理解
·描述图文并茂：给出具体操作步骤，易于快速动手实践
中国工信出版集团
电子工业出版社

本书围绕Vivado四大主题

- 设计流程
- 时序约束
- 时序分析
- Tcl脚本的使用

作者：高亚军（Xilinx战略应用高级工程师）

- 2012年2月，出版《基于FPGA的数字信号处理（第1版）》
- 2012年9月，发布网络视频课程《Vivado入门与提高》
- 2015年7月，出版《基于FPGA的数字信号处理（第2版）》
- 2016年7月，发布网络视频课程《跟Xilinx SAE学HLS》

◆ 内容翔实全面：涵盖Vivado所有基本功能
◆ 讲解深入浅出：结合大量案例，帮助读者加强对基本概念的理解
◆ 描述图文并茂：给出具体操作步骤，易于快速动手实践

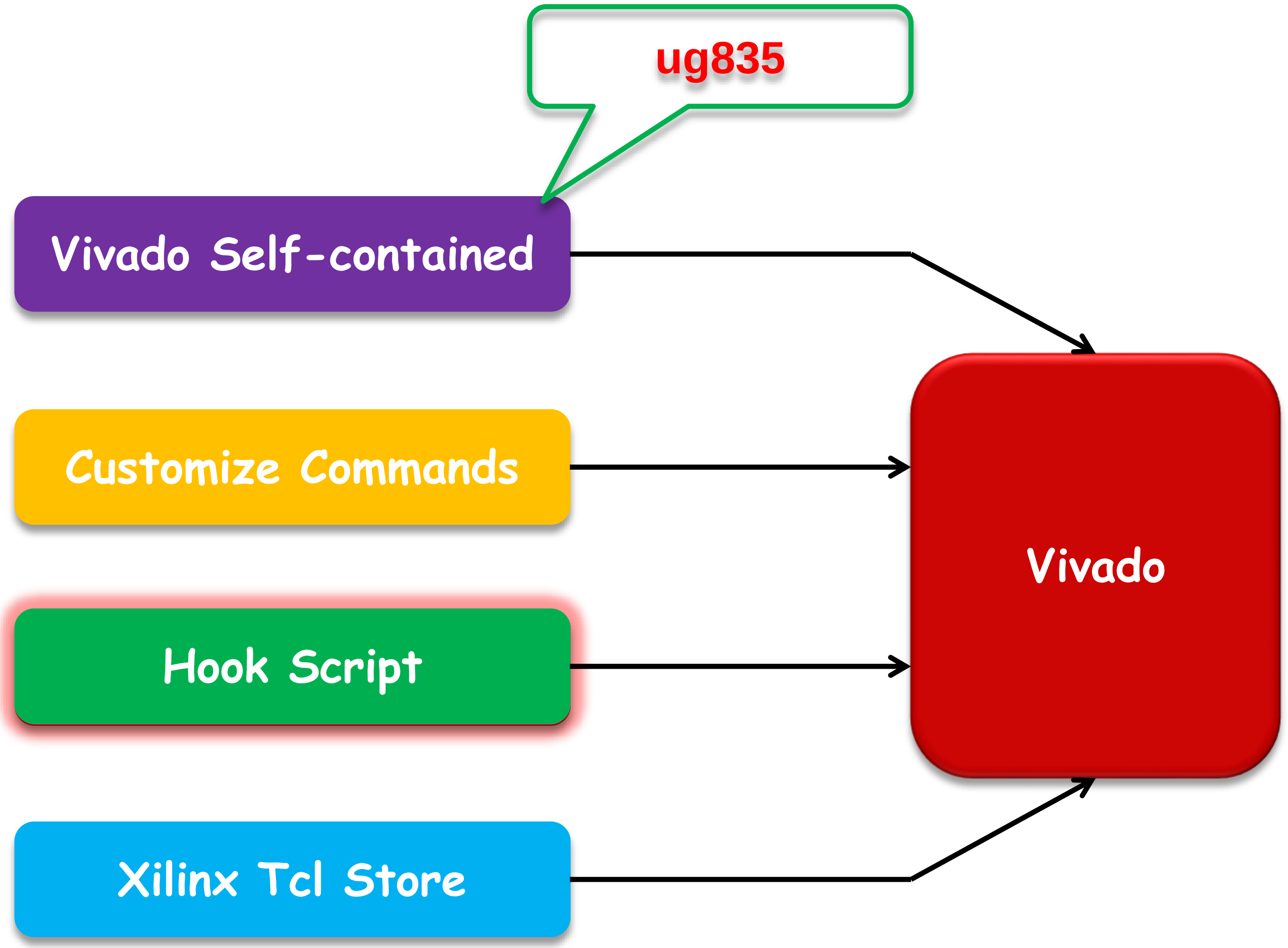
XILINX
ALL PROGRAMMABLE™

TCL, Vivado One World

Part 3: Hook Scripts

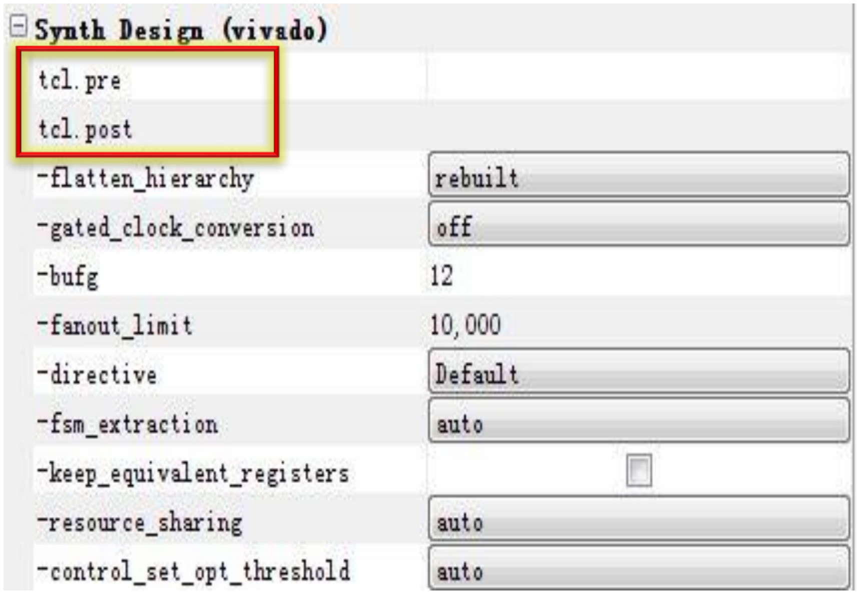
Lauren Gao

Tcl Sources in Vivado

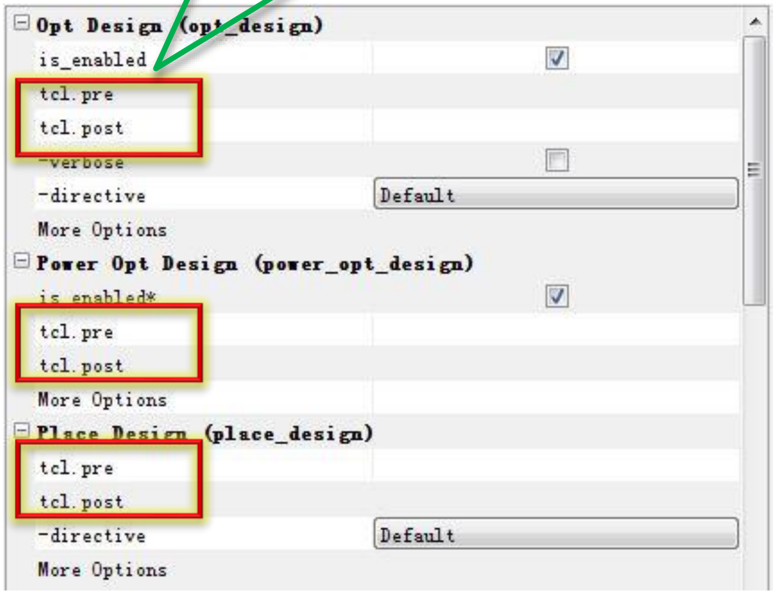


Hook Script

- What is hook script?
  - It is TCL pre/post capability for a Vivado process
- All the process in Vivado contains this tcl.pre/.post option
  - Synthesis and Implementation including each sub-step
  - tcl.pre: prior to synthesis and implementation
  - tcl.post: after synthesis and implementation



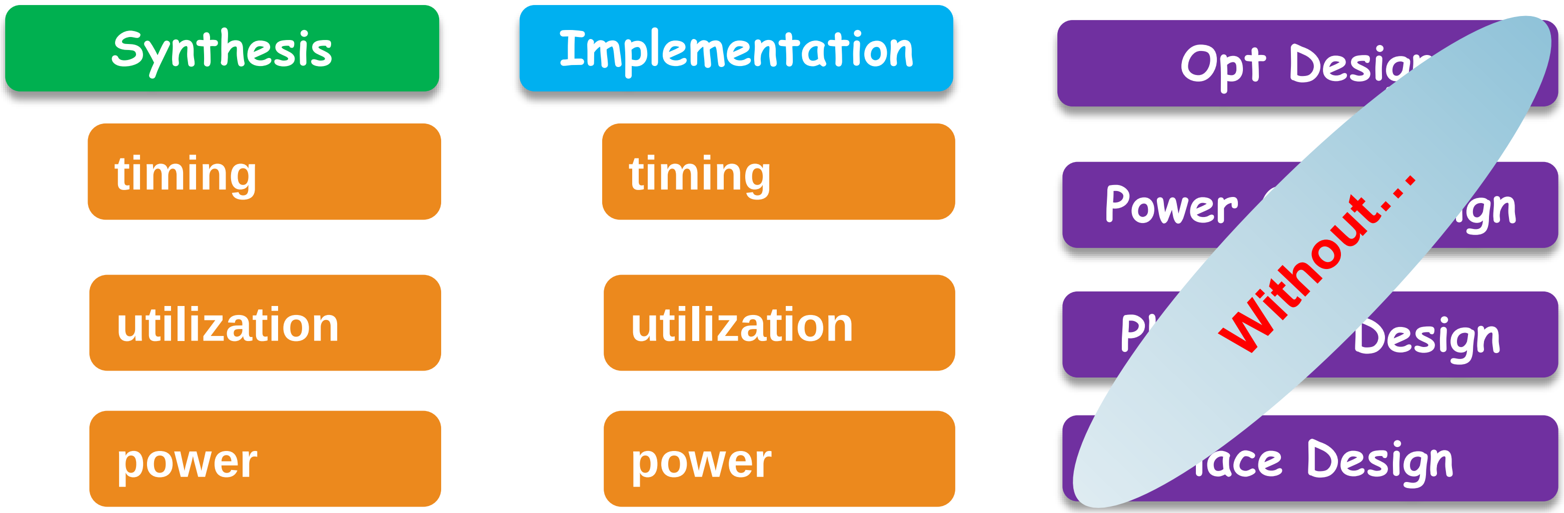
Synth Design (vivado)	
tcl.pre	
tcl.post	
-flatten_hierarchy	rebuilt
-gated_clock_conversion	off
-bufg	12
-fanout_limit	10,000
-directive	Default
-fsm_extraction	auto
-keep_equivalent_registers	☐
-resource_sharing	auto
-control_set_opt_threshold	auto



Opt Design (opt_design)	
is_enabled	✓
tcl.pre	
tcl.post	
-verbose	☐
-directive	Default
More Options	
Power Opt Design (power_opt_design)	
is_enabled*	✓
tcl.pre	
tcl.post	
More Options	
Place Design (place_design)	
tcl.pre	
tcl.post	
-directive	Default
More Options	

Common Uses of Hook Scripts

- Custom reports
  - timing, power, utilization, or any user-defined tcl report
- Modifying the timing constraints for portions of the flow only
- Modifications to netlist, constraint, or device programming



Specify a Hook Script

- GUI
  - Both in [Synthesis Settings]() and in [Implementation Settings]()
  - Tcl script
- Specify a hook script with Tcl script
  - The properties to set on a synthesis run
    - STEPS.SYNTH_DESIGN.TCL.PRE
    - STEPS.SYNTH_DESIGN.TCL.POST

Example

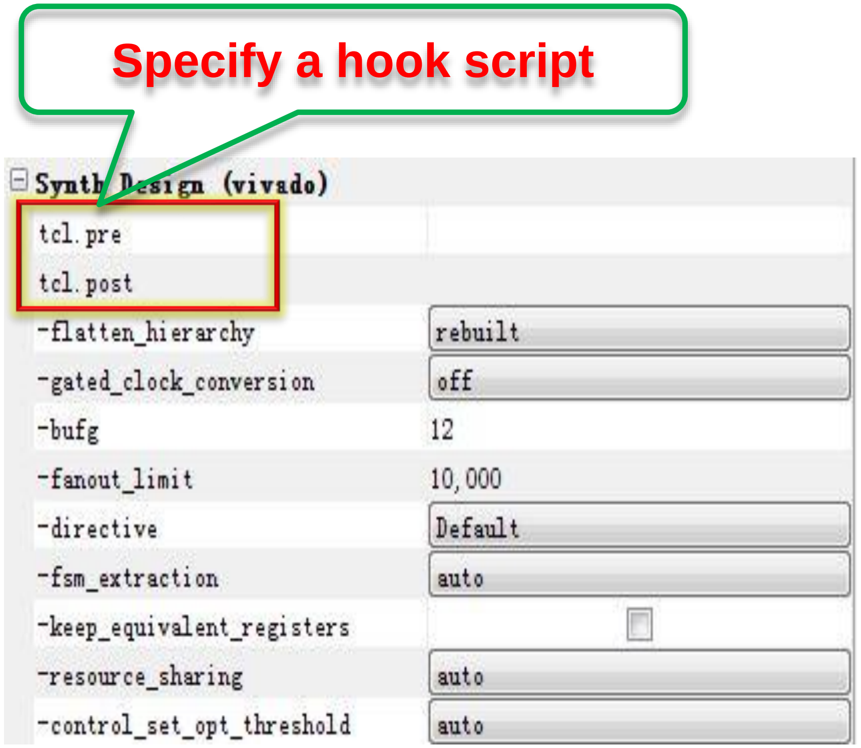
```
set_property STEPS.SYNTH_DESIGN.TCL.PRE \
 {C:/Data/report.tcl} [get_runs synth_1]
```

Specify a Hook Script

- You can define Tcl scripts before and after each step of the implementation process
  - Opt Design
  - Power Opt Design
  - Place Design,
  - Post-Place Power Opt Design
  - Phys Opt Design
  - Route Design
  - Bitstream generation

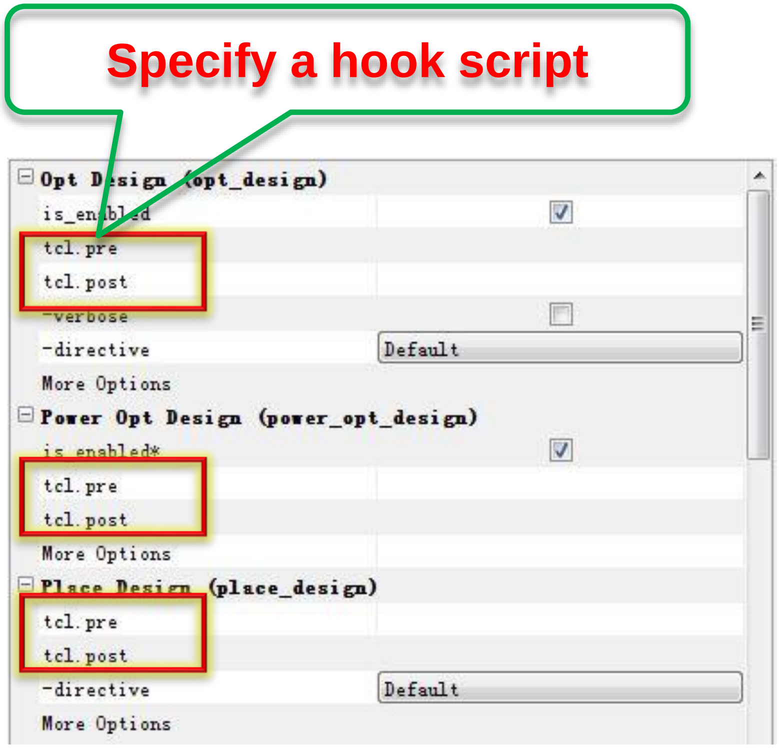
```
STEPS.OPT_DESIGN.TCL.PRE
STEPS.OPT_DESIGN.TCL.POST
STEPS.POWER_OPT_DESIGN.TCL.PRE
STEPS.POWER_OPT_DESIGN.TCL.POST
STEPS.PLACE_DESIGN.TCL.PRE
STEPS.PLACE_DESIGN.TCL.POST
STEPS.POST_PLACE_POWER_OPT_DESIGN.TCL.PRE
STEPS.POST_PLACE_POWER_OPT_DESIGN.TCL.POST
STEPS.PHYS_OPT_DESIGN.TCL.PRE
STEPS.PHYS_OPT_DESIGN.TCL.POST
STEPS.ROUTE_DESIGN.TCL.PRE
STEPS.ROUTE_DESIGN.TCL.POST
STEPS.WRITE_BITSTREAM.TCL.PRE
STEPS.WRITE_BITSTREAM.TCL.POST
```

It's Simple to Specify a Hook Script



Synth Design (vivado)

tcl.pre	
tcl.post	
-flatten_hierarchy	rebuilt
-gated_clock_conversion	off
-bufg	12
-fanout_limit	10,000
-directive	Default
-fsm_extraction	auto
-keep_equivalent_registers	☐
-resource_sharing	auto
-control_set_opt_threshold	auto



Opt Design (opt_design)

is_enabled	✓
tcl.pre	
tcl.post	
-verbose	☐
-directive	Default
More Options	

Power Opt Design (power_opt_design)

is_enabled*	✓
tcl.pre	
tcl.post	
More Options	

Place Design (place_design)

tcl.pre	
tcl.post	
-directive	Default
More Options	

```
set_property STEPS.<STEP_NAME>.TCL.PRE <Tcl File>\
[get_runs synth_1]

set_property STEPS.<STEP_NAME>.TCL.POST <Tcl File>\
[get_runs impl_1]
```

Relative Paths in Hook Script

- Relative paths within the tcl.pre and tcl.post scripts are relative to the appropriate run directory of the project they are applied to:
  - `<project>/<project.runs>/<run_name>`
- You can use the DIRECTORY property of the current project or current run to define the relative paths in your Tcl hook scripts:
  - `get_property DIRECTORY [current_project]`
  - `get_property DIRECTORY [current_run]`

DEMO