Using Tcl Commands in the Vivado Design Suite Project Flow Demo Script

Introduction

This demonstration introduces the execution of a Tcl script in the Vivado® Design Suite project-based flow. In project-based flow you can access the integrated design environment (IDE) to configure, launch, and manage the entire design process, integrate IP configuration and implementation, visually analyze the design and apply constraints at any stage, cross-probe back to RTL source files, and full Tcl and IDE use model support.

Preparation:

- Required files: `$TRAINING_PATH/Tcl_ProjectFlow/demo/KCU105/verilog`
- Required hardware: None
- Required software: Vivado Design Suite, Tcl editor such as Notepad++ (Windows) or gedit (Linux)

Note: If you intend to build the script by cutting-and-pasting commands from this document you may need to re-enter some commands manually.

Using Tcl Commands in the Vivado Design Suite Project Flow

Action with Description	Point of Emphasis and Key Takeaway

• Open a Tcl editor, such as Notepad++ (Windows) or gedit (Linux), to create a Tcl script file.
  • Select **File** > **New** to create a blank file.
  • Name the file **`projectBased.tcl`** and save as a *.tcl* file in the following directory location:
  `$TRAINING_PATH/Tcl_ProjectFlow/demo/KCU105/verilog`
  Note: Manually edit the file extension to *.tcl*.
  Use `$Tcl_ProjectFlow = /home/xilinx/training/Tcl_ProjectFlow` if there are any issues.
	The Tcl file is created to store project-based flow Tcl commands that will be sourced in later steps.

• Create the Vivado Design Suite project by using the `create_project` Tcl command:
`create_project wave_gen /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog -part xcku040-ffva1156-2-e`
• Set the created project to target a specific board part:
`set_property board_part xilinx.com:kcu105:part0:1.7 [current_project]`
	The Vivado Design Suite project is created with the name *wave_gen* and targeted to the Kintex® UltraScale™ device part `xcku040-ffva1156-2-e`.

Action with Description	Point of Emphasis and Key Takeaway
• Add the provided Verilog source files to the project created. `add_files /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/sources/hdls/verilog` This will make it possible to see the added files in the source file hierarchy.	The objective is to add the HDL source files in the project.
• Enable the Vivado Design Suite to find the filesets available in the project: `get_filesets` This will show the list of filesets *sources_1*, *constrs_1*, and *sim_1*.	Gets a list of filesets in the current project that match a specified search pattern. The default command gets a list of all filesets in the project.

• Import the existing clock and FIFO IP cores into the project. The IP is available in the `../verilog/sources/hdls/ip` directory:
`import_ip /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/sources/hdls/ip/char_fifo/char_fifo.xci`
`import_ip /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/sources/hdls/ip/clk_core/clk_core.xci`
	These commands import the existing IP core files into the project.

Action with Description	Point of Emphasis and Key Takeaway
• Add the two constraint files from the `../verilog/sources/constraints` directory: `add_files -fileset constrs_1 /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/sources/constraints` The fileset `constrs_1` folder is created with the project by default. Users can always create the fileset of choice by using the `create_fileset` Tcl command (e.g., `create_fileset -constrset constrs_2`).	This command adds a list of files to the specified fileset in the current project that matches a specified search pattern.
• Add the simulation testbenches to the project from the `../verilog/sources/testbenches/verilog` directory: `add_files -fileset sim_1 /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/sources/testbenches/verilog` The fileset `sim_1` folder is created with the project by default. Users can always create the fileset of choice by using the `create_fileset` Tcl command (e.g., `create_fileset -simset sim_2`).	This command adds the existing testbench files to the project.

Action with Description	Point of Emphasis and Key Takeaway

• Update the compile order of the design sources in the current project, or in the specified fileset:
`update_compile_order -fileset sim_1`
`update_compile_order -fileset sources_1`
	These commands update the fileset compile order and the hierarchy update settings.

• Synthesize the design using the command `launch_runs` in project mode:
`launch_runs synth_1`
`wait_on_run synth_1`
	A synthesis run called synth_1 is generated during project creation.
	Note: `wait_on_runs` blocks the execution of Tcl commands until the specified run completes, or until the specified amount of time has elapsed. This command is used for running the tool in batch mode or from Tcl scripts. It is ignored when running interactively from the GUI.

• Open the specified synthesis run's netlist: `open_run synth_1 -name netlist_1`	The run properties define the target part and constraint set that is used with the synthesis result to create a netlist version of the design.
• Generate a timing summary of a design and store in an RPT (Report) file on an open synthesized design: `report_timing_summary -file /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/timing_summary_post_synth.rpt`	Generate a timing summary to help in understanding if the design has met timing requirements and where timing problems may exist.

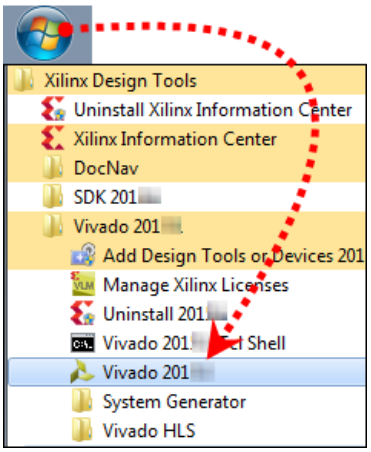
Action with Description	Point of Emphasis and Key Takeaway

• Implement the design by using the command `launch_runs` tools in project mode:
`launch_runs impl_1`
`wait_on_run impl_1`
	An implementation run called impl_1 is created by this command.

• Write a bitstream for the implemented design:
`launch_runs impl_1 -to_step write_bitstream`
`wait_on_run impl_1`
	This command writes a bitstream file for the current project. This command must be run on an implemented design.

• Open the implemented design: `open_run impl_1`	The run properties define the target part and constraint set that is combined with the implementation results to create the design view in the Vivado IDE.
• Compute and report the device utilization of resources on the target part by the current synthesized or implemented design: `report_utilization -file /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/report_utilization.txt` • Open the **`report_utilization.txt`** created in the project directory after the command has been executed.	Though resource utilization can be reported early in the design process, the report will be more accurate as the design progresses from synthesis through implementation.
• Save the **`projectBased.tcl`** script.	The script file will be sourced from the Vivado Tcl Console.

Action with Description	Point of Emphasis and Key Takeaway

• Launch the Vivado Design Suite.
  • **[Windows users]:** Select **Start** > **All Programs** > **Xilinx Design Tools** > **Vivado 2021.2** > **Vivado 2021.2**.
 
 	
Figure 1: Launching the Vivado Design Suite from the Start Menu
 	
-- OR --
 	
Double-click the **Vivado Design Suite** shortcut icon () on the desktop.
  • **[Linux users]:** Press <**Ctrl** + **Alt** + **T**> to open a new terminal window, and source the **`settings.sh`** file located at `/opt/Xilinx/Vivado/2021.2`.
  • Type **`vivado`** and press <**Enter**> to launch the Vivado Design Suite environment.
	The Vivado Design Suite opens to the Welcome window. From the Welcome window you can create a new project, open an existing project, or enter Tcl commands directly into the Vivado Design Suite as well as access documentation and example projects.

Action with Description	Point of Emphasis and Key Takeaway
• Source and run the **`projectBased.tcl`** file in the project-based flow in the Vivado Tcl Console: `source /home/xilinx/training/Tcl_ProjectFlow/demo/KCU105/verilog/projectBased.tcl`	Execute the Tcl script to verify that the script worked properly. This will finally generate the project and compile the processes. **If you obtain the following error**: "ERROR: [Common 17-165] Too many positional options when parsing..." Locate the line number listed and manually re-type the command. This error is typically caused by the "-" character when typed in a Windows editor like notepad. Then continue executing the Tcl commands. **If you do not want to execute all commands in the file:** You can add the # in front of the command to comment the command out. This can be used if you make edits to the file and do not want to re-execute all the commands.
• Open the **`timing_summary_post_synth.rpt`** and **`report_utilization.txt`** files created in the project directory to review the post-synthesis timing summary and post-implementation utilization report respectively.	Besides creating reports at each stage of the design flow, Tcl can be used in the Vivado IDE to modify and customize such reports.

Summary

This demonstration showed how to build and execute a project in the project-based design flow by using Xilinx Tcl commands.

References:

- Supporting materials
  - *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894)