Digitronix Nepal
Innovation from Engineers...

Embedded System Design with Xilinx VIVADO & Zynq FPGA

Course Prepared by

Digitronix Nepal
www.digitronixnepal.com

LogicTronix
www.LogicTronix.com

logictronix
FPGA, ASIC & VLSI Design:Verification

Section 8. Tcl Scripting with VIVADO

Lecture 1 : Tool Command Language (Tcl) Introduction

Lecture 2 : Basics Tcl commands

Lecture 3 : Lab81: Creating a basic Project on VIVADO IPI with Tcl

Lecture 4 : Lab82: Importing Tcl Scripts from Github or tcl file

Objective of the Section

After Completing this section you will be able to:

- Describe and explain about Tool Command Language(Tcl) Scripting
- Explain about Tcl commands for creating, opening, adding , exporting sources on Vivado project
- Basic project design with tcl on Vivado.

Lecture 1 : Tcl Introduction

- The Tool Command Language (Tcl) is the scripting language integrated in the Vivado tool environment.
- Tcl is a standard language in the semiconductor industry for application programming interfaces, and is used by Synopsys® Design Constraints (SDC).
- SDC is the mechanism for communicating timing constraints for FPGA synthesis tools from Synopsys Synplify as well as other vendors, and is a timing constraint industry standard; consequently, the Tcl infrastructure is a “Best Practice” for scripting language.
- Tcl lets you perform interactive queries to design tools in addition to executing automated scripts.

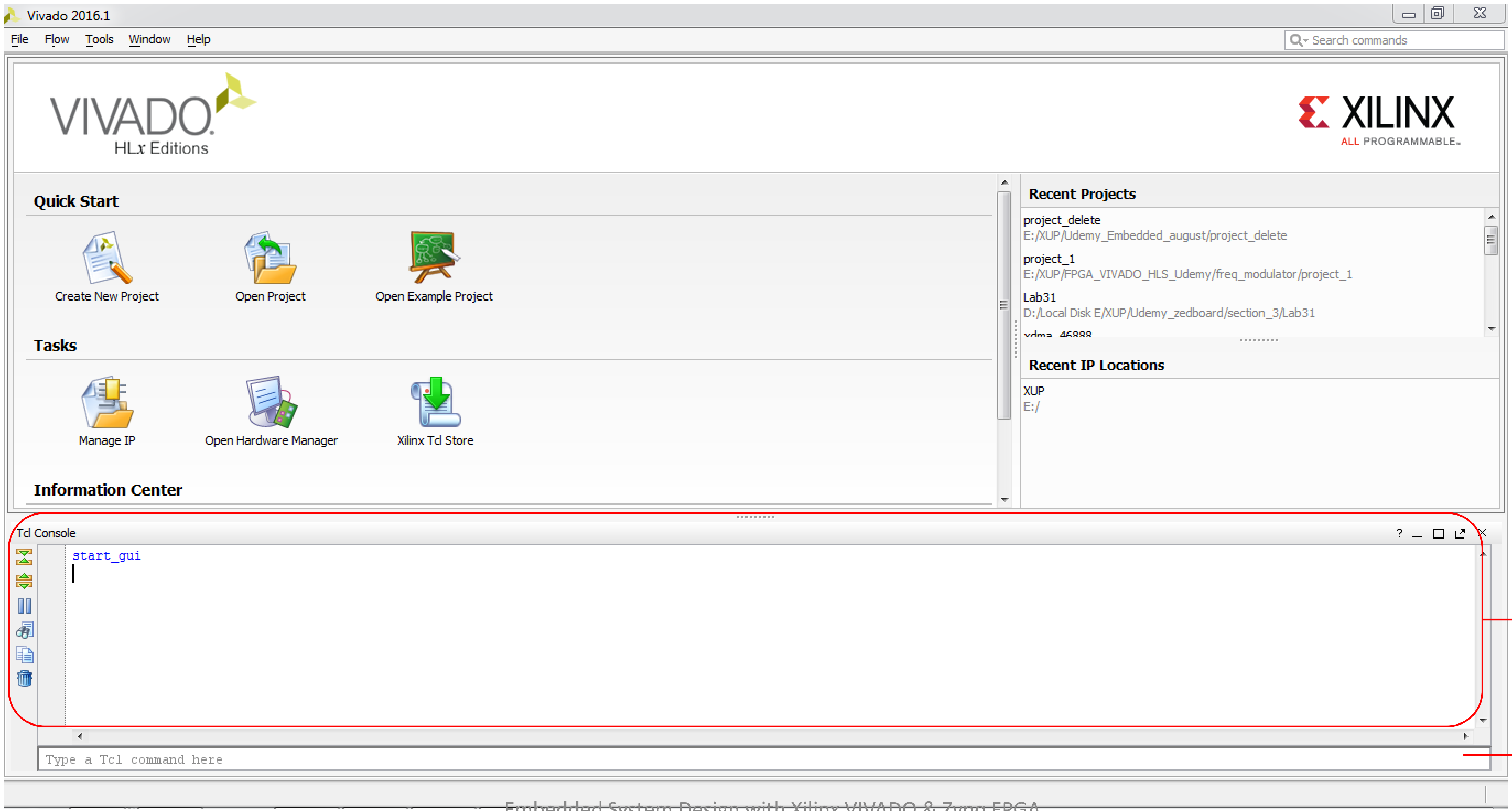
Need for Tcl Commands

- Optimize the design and design methodology
- Scripting methodology on FPGA Design methodology
- Creating Board, Timing, Debugging Constraints on VIVDO with Scripting.
- Speeding the Design Flow
- Tcl automation is one of the most powerful features integrated into the Vivado and Xilinx SDK tools and should be fully exploited to maximize your productivity as an FPGA developer.
- Tcl allow to querying specific timing analysis reporting commands live, applying incremental constraints, and performing queries immediately after to verify expected behavior without re-running any tool steps.

VIVADO Tcl is highly preferred for:

- Creating new project:
- Adding sources
- Creating block design
- Inserting IP on the block design
- Creating board constraints
- Creating debugging and timing constraints
- The Tcl file (of few KB) can be used to create project on similar version of VIVADO or Older version of VIVADO.
- While transferring Project it will take 100’s MB or if you archive project it will take 10’s MB memory which is complicated to transfer and launch project.
- While Creating the Project with GUI VIVADO automatically generates a Tcl commands on the Tcl Console which can be copied and edit for further use.

VIVADO Tcl Console and UI: Starting VIVADO

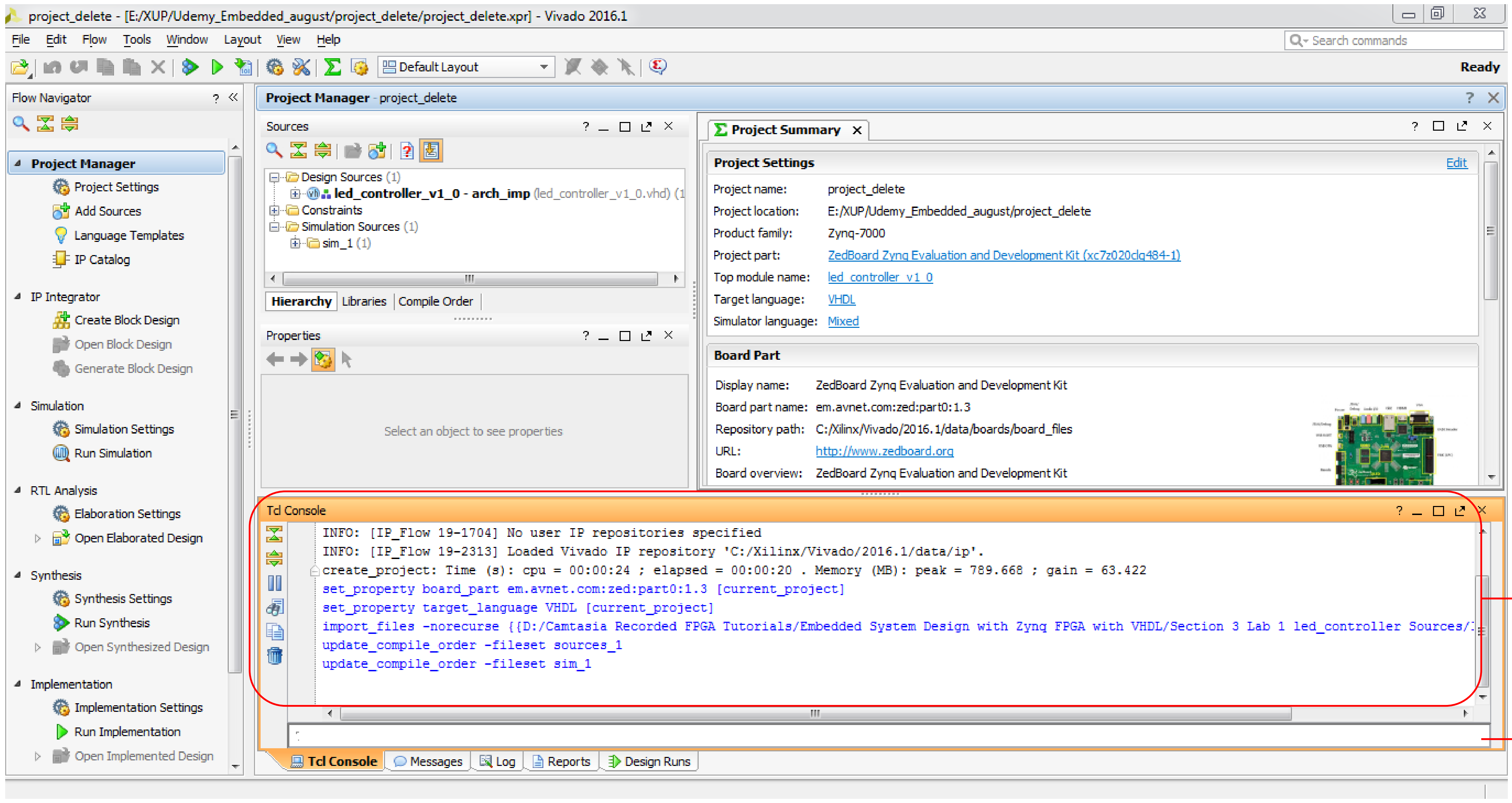


```
start_gui
```

VIVADO Tcl Console

Console for inserting commands

VIVADO Tcl Console and UI



Project Settings

Project name: project_delete
Project location: E:/XUP/Udemy_Embedded_august/project_delete
Product family: Zynq-7000
Project part: ZedBoard Zynq Evaluation and Development Kit (xc7z020clg484-1)
Top module name: led_controller_v1_0
Target language: VHDL
Simulator language: Mixed

Board Part

Display name: ZedBoard Zynq Evaluation and Development Kit
Board part name: em.avnet.com:zed:part0:1.3
Repository path: C:/Xilinx/Vivado/2016.1/data/boards/board_files
URL: http://www.zedboard.org
Board overview: ZedBoard Zynq Evaluation and Development Kit

```
INFO: [IP_Flow 19-1704] No user IP repositories specified
INFO: [IP_Flow 19-2313] Loaded Vivado IP repository 'C:/Xilinx/Vivado/2016.1/data/ip'.
create_project: Time (s): cpu = 00:00:24 ; elapsed = 00:00:20 . Memory (MB): peak = 789.668 ; gain = 63.422
set_property board_part em.avnet.com:zed:part0:1.3 [current_project]
set_property target_language VHDL [current_project]
import_files -norecurse {{D:/Camtasia Recorded FPGA Tutorials/Embedded System Design with Zynq FPGA with VHDL/Section 3 Lab 1 led_controller Sources/
update_compile_order -fileset sources_1
update_compile_order -fileset sim_1
```

VIVADO Tcl Console

Console for inserting commands

Lecture 2 : Basic Tcl Commands

1. Change directory of Vivado: `cd e:/xup/Udemy_Embedded/`
2. Creating New project with specified directory and board/part: `create_project project_delete E:/XUP/Udemy_Embedded/project_delete -part xc7z020clg484-1`
3. Adding Design Sources and Constraints: `add_files E:/XUP/Udemy_Embedded/project_delete/project_delete.srcs/sources_1/new/new.v`
4. Crating block design: `create_bd_design design_name`
5. Adding IP on block design: `startgroup`

```
   create_bd_cell -type ip -vlnv xilinx.com:ip:axi_gpio:2.0 axi_gpio_0
   Endgroup
```

1. Updating Fileset and Compile Order: `update_compile_order -fileset sources_1`
2. Close block design: `close_bd_design`
3. Closing project: `close_project`
4. Exit fro Vivado: `exit`

See more tcl commands at
VIVADO Tcl Commands: UG835

Contd..

10. Synthesize a VIVADO project:

```
# Synthesize project
launch_runs synth_1
wait_on_run synth_1
```

11. Implement the Project

```
# Implement project
launch_runs impl_1 -to_step write_bitstream
wait_on_run impl_1
```

12. Open VIVADO Project:

```
# Open project
open_project $origin_dir/$proj_name/$proj_name.xpr
```

Contd..

13. Export VIVADO Project to SDK

```
# Export project to SDK
set bit_filename [lindex [glob -dir
"$origin_dir/$proj_name/${proj_name}.runs/impl_1" *.bit] 0]
set bit_filename_only [lindex [split $bit_filename /] end]
set top_module_name [lindex [split $bit_filename_only .] 0]
set export_dir "$origin_dir/$proj_name/$proj_name.sdk"
file mkdir $export_dir
write_sysdef -force \
  -hwdef "$origin_dir/$proj_name/${proj_name}.runs/impl_1/$top_module_name.hwdef" \
  -bitfile "$origin_dir/$proj_name/${proj_name}.runs/impl_1/$top_module_name.bit" \
  -meminfo "$origin_dir/$proj_name/${proj_name}.runs/impl_1/$top_module_name.mmi" \
$export_dir/$top_module_name.hdf
```

Lecture 3 : Lab81: Creating a basic Project on VIVADO IPI with Tcl

Design Steps:

1. Open VIVADO from GUI or Console (**Vivado 201x.x Tcl Shell)** : `start_gui`
2. A. Run the Following Command : Creating New project with specified directory and board/part:

```
create_project project_delete E:/XUP/project_delete -part xc7z020clg484-1
```

3. Adding Design Sources and Constraints:

```
file mkdir E:/XUP/project_delete/project_delete.srcs/sources_1/new
close [ open E:/XUP/project_delete/project_delete.srcs/sources_1/new/new.vhd w ]
add_files E:/XUP/project_delete/project_delete.srcs/sources_1/new/new.vhd
```

Adding Constraint: `file mkdir E:/XUP/project_delete/project_delete.srcs/constrs_1/new`

```
close [open E:/XUP/project_delete/project_delete.srcs/constrs_1/new/constraint.xdc w
]
add_files -fileset constrs_1
E:/XUP/project_delete/project_delete.srcs/constrs_1/constraint.xdc
```

4. Now you can run the synthesis: `launch_runs synth_1`

5. Now run the implementation: `launch_runs impl_1`

6. Run Generate Bitstream: `write_bitstream file_name.bit`

Lecture 4 :Lab82: Importing Tcl Scripts from Github or tcl file

<u>Design Steps:</u>

1. Open VIVADO from GUI or Console: `start_gui`
2. A. Change the directory of VIVADO where your Tcl file located as follows and run source command

```
eg: cd E:/xup/
Source project_name.tcl
```

If you don't change the directory and run the tcl file then project will created on `C:/user/user/AppData/Roaming/Xilinx/Vivado`

Or

B. For creating project with VIVADO GUI; Go to Tools and click on run tcl commands→ Locate your tcl source directory→click open.

3. Now you can run the synthesis: `launch_runs synth_1`

4. Now run the implementation (before it you might need to have constraint in the current directory): `launch_runs impl_1`

5. Run Generate Bitstream: `write_bitstream file_name.bit`

Some Necessary Tcl Commands

- Creating/generating Tcl file of the Block Design: Example

```
write_bd_tcl E:/XUP/Udemy_Embedded_august/project_name.tcl
```

- Creating/generating tcl file of the project: Example

```
write_project_tcl E:/xup/Udemy_Embedded_august/project_new.tcl
```

How to Run Tcl Command on VIVADO 2016.1 which is generated by VIVADO 2016.4 or Later

1. Open the Tcl file on Notepad++ (actually it support Tcl for Editing)
2. Update the VIVADO Synthesis Version according to your VIVADO Version
3. Some IP version might have updated on the tcl file so initially run the tcl file updated after step 2 and see if any errors occurs on Vivado.
4. Point out which ip is not supported, then goto tcl file and update/edit the ip version to older one or previous one.
5. Run the tcl file again on Vivado tcl console.
6. Cheers!!! It might work.
7. Or you can run the tcl command on same Vivado version and run that project on older version, actually you can open Vivado project on older version to.

For Version Control of VIAVDO Projects with Tcl: Customizing Tcl Files

- You can visit Blog of FPGA Developer by Jeff Johnson

http://www.fpgadeveloper.com/2014/08/version-control-for-vivado-projects.html

Tcl Scripting with VIVADO Design References:

Key Documents

Name	Description
UG835 VIVADO Tcl Commands	Tcl commands and Support on VIVADO
UG894 VIVADO Tcl Scripting	VIVADO Tcl Scripting

Let's Go to VIVADO for the project

Thank You!