

Introduction to Zynq Hardware

Lab 4

Using TCL in Vivado Embedded Designs



June 2017
Version 10

Lab 4 Overview

This lab will give users a brief introduction to various Tool Command Language (TCL) commands that can be run to modify the project as well as archive the project. It's wise to understand TCL as all Xilinx tools run TCL under the hood. This can be powerful for scripting commands for design flows that require several steps. Additionally, we'll explore some of the files that have been created so far.

Lab 4 Objectives

When you have completed Lab 4, you will know how to do the following:

- Open and close block designs using TCL
- Execute simple TCL commands to manipulate IP Integrator block designs
- Export a block design to a TCL file on disk

Experiment 1: Explore Project Files

This experiment explores some of the files created thus far.

Experiment 1 General Instruction:

Open Windows Explorer and browse the files created by Vivado to this point.

Experiment 1 Step-by-Step Instructions:

1. <Optional> If you did not complete Lab 3 or wish to start with a clean copy, delete the ZynqDesign folder in the ZynqHW/2017_1 folder. Then unzip **Solutions\ZynqHW_Lab3_Solution.zip** to the 2017_1 folder. If you have 7-Zip installed, you can do this by right-clicking and dragging **ZynqHW_Lab3_Solution.zip** to the 2017_1 folder. Select **7-Zip → Extract Here**.
2. Launch **Windows Explorer** (Windows Key + E).
3. **Navigate** to the Speedway Project folder:

C:\Speedway\ZynqHW\2017_1\ZynqDesign










Name	Date modified	Type	Size
 ZynqDesign.cache	8/28/2017 11:44 AM	File folder	
 ZynqDesign.hw	8/28/2017 11:44 AM	File folder	
 ZynqDesign.ip_user_files	8/28/2017 11:44 AM	File folder	
 ZynqDesign.lab3	8/28/2017 11:44 AM	File folder	
 ZynqDesign.runs	8/28/2017 11:44 AM	File folder	
 ZynqDesign.sdk	8/28/2017 11:44 AM	File folder	
 ZynqDesign.sim	6/16/2017 10:34 AM	File folder	
 ZynqDesign.srscs	8/28/2017 11:44 AM	File folder	
 ZynqDesign.xpr	6/16/2017 10:56 AM	Vivado Project File	9 KB

Figure 1 - Project Directory Contents

4. There is only one Vivado source file here, **ZynqDesign.xpr**. This is our Vivado project.
5. Our current design is very basic thus far, so there are not many interesting files here. Though the files we have created exist in .srscs folder. **Navigate** to the following folder:

ZynqDesign.srscs\sources_1\bd\Z_system

Name	Date modified	Type	Size
hdl	8/28/2017 11:44 AM	File folder	
hw_handoff	8/28/2017 11:44 AM	File folder	
ip	8/28/2017 11:44 AM	File folder	
ipshared	8/28/2017 11:44 AM	File folder	
ui	8/28/2017 11:44 AM	File folder	
Z_system.bd	6/16/2017 10:54 AM	BD File	12 KB
Z_system.bxml	6/16/2017 10:55 AM	BXML File	3 KB
Z_system_ooc.xdc	6/16/2017 10:54 AM	XDC File	1 KB

Figure 2 - Project Sources

The only project source we have so far is our block diagram and it can be found in this directory.

6. **Browse** into the HDL folder, you will see our top-level system wrapper as well as our block diagram wrapper.

<< 2015_2 > ZynqDesign > ZynqDesign.srcs > sources_1 > bd > Z_system > hdl			
Include in library ▼ Share with ▼ Burn New folder			
Name	Date modified	Type	
Z_system.hwdef	06/10/2015 10:52	HWDEF File	
Z_system.vhd	06/10/2015 10:52	VHD File	
Z_system_wrapper.vhd	06/10/2015 10:52	VHD File	

Figure 3 - HDL Wrapper Files

As we progress through the labs, keep checking back into these folders to see what files are created.

Experiment 2: TCL Overview

This experiment will illustrate how to use TCL commands to navigate through Vivado and manipulate our project.

Experiment 2 General Instruction:

Open a Vivado project using TCL commands as well as open block diagram, add remove ports, query design and export block design to a TCL file.

Experiment 2 Step-by-Step Instructions:

1. Launch **Vivado**.
2. Take note from the previous Experiment 1 where the Vivado project is located on the PC.
3. Notice the TCL console is available immediately and commands can be entered without opening a project. Click the **Tcl Console** tab at the bottom of the Vivado initial window and change the working directory to the Vivado Project directory by entering the following command: (Note: slashes for the directories are forward slashes, not backslashes as Windows uses.)

```
cd C:/Speedway/ZynqHW/2017_1/ZynqDesign
```



Figure 4 - Change Working Directory

4. Type **pwd** to print the current working directory

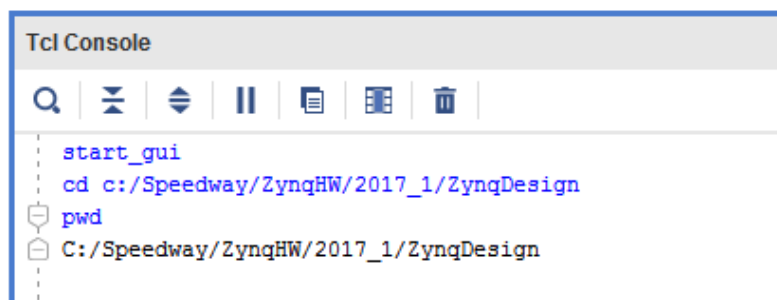


Figure 5 - Print Working Directory

5. In the TCL console, open the Vivado project by typing “**open_p**”. As TCL commands are entered possible options appear. When **open_project** appears hit tab to auto-complete the string. Then enter the relative path to our Vivado project name. By starting the path with “./” this will activate the auto-complete function for the path string.

open_project ./ZynqDesign.xpr

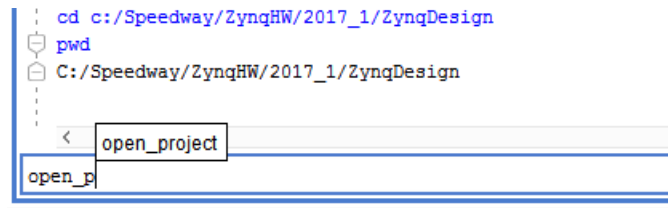


Figure 6 - Vivado TCL Console

6. Our project will open. Open the block design using the following TCL command:

open_bd_design ./ZynqDesign.srscs/sources_1/bd/Z_system/Z_system.bd

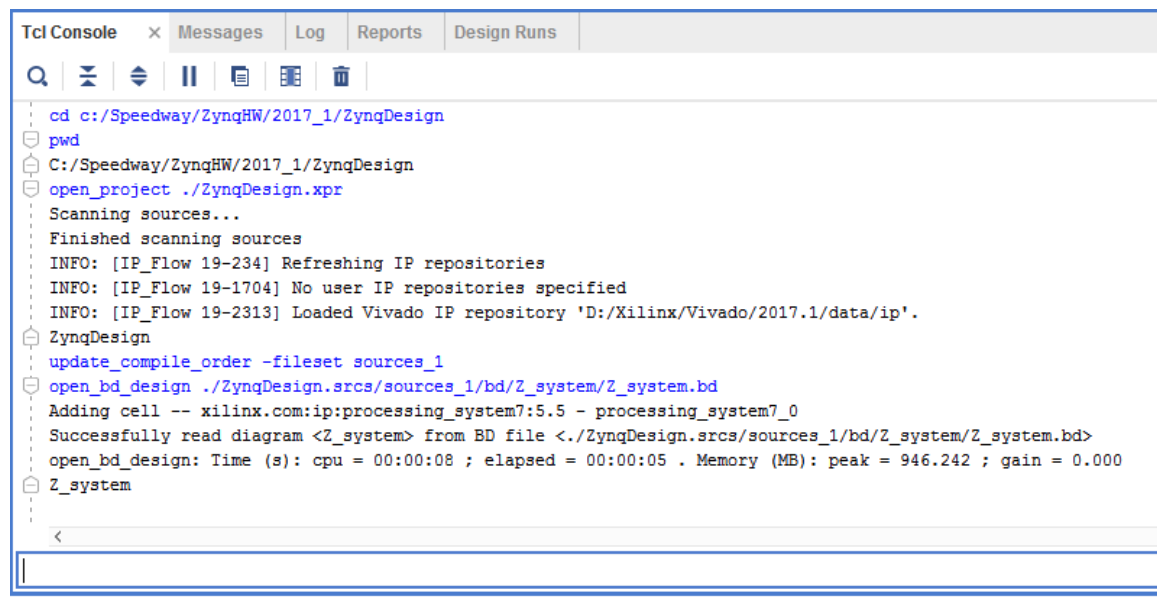


Figure 7 - Open Block Diagram

To recap, we’ve opened our project all the way down to the block design with only using TCL commands. One of the powerful features of TCL is scripting. And these commands can be stored and run from a single script.

With the block design open there are many things we can do with TCL. We can query a list of ports, create interface ports and nets, and even get a list of address spaces.

7. Start by querying a list of interfaces ports, type “**get_bd_intf_ports**”. Of course our design is simple and only two ports are reported; DDR and FIXED_IO.



Figure 8 - Get Interface Ports

- Ports can be created right from the TCL console. Create an output port for reset, type **"create_bd_port -dir O -type rst Reset"**. Note: all created ports require a direction type [I | O | IO] and optionally a type [CLK | RST | CE | INTR | DATA | UNDEF].

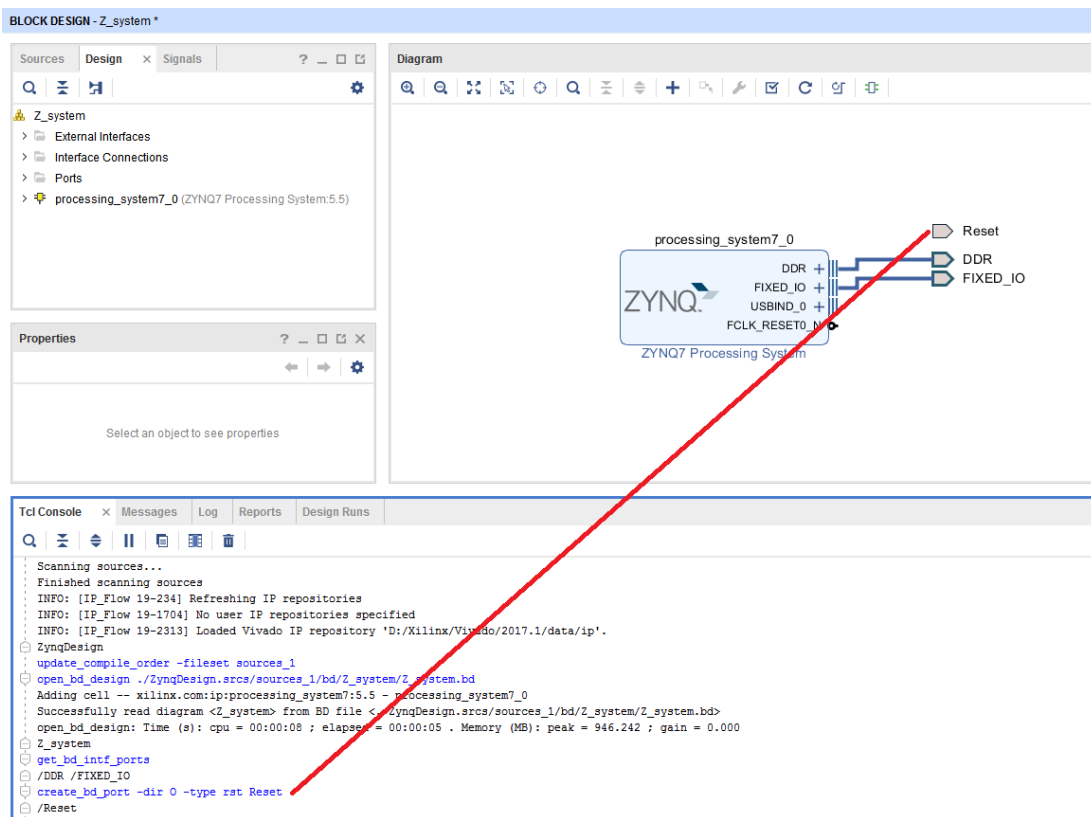


Figure 9 - Create Ports

9. Ports can be deleted as well. And if you are not sure what TCL command is being used to perform a task, Vivado echoes all GUI commands via the TCL Console. Right-click on the newly created Reset port and select **Delete**.

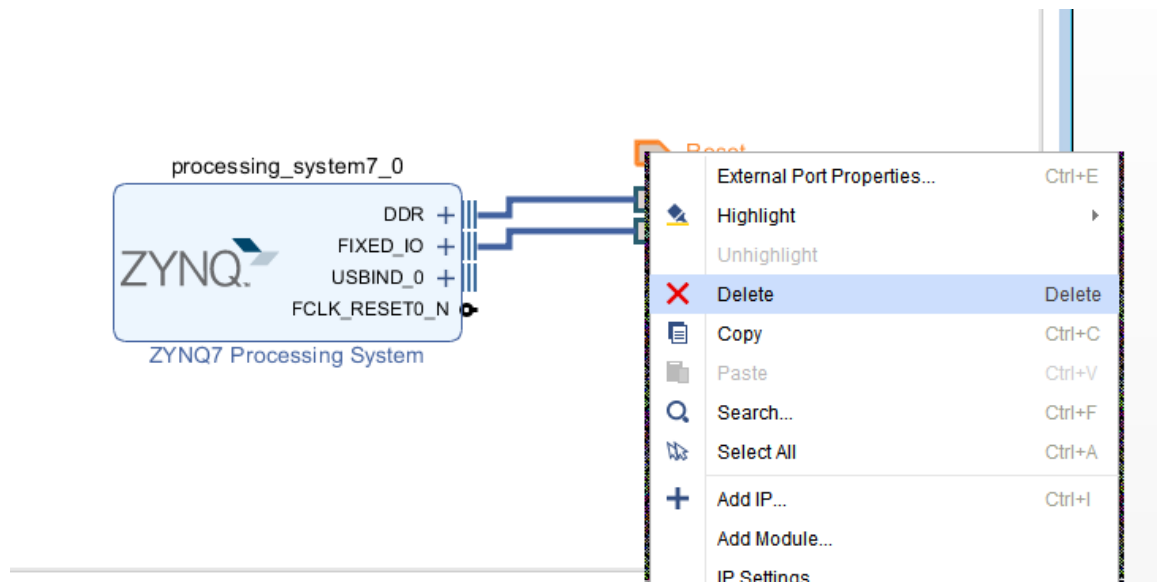


Figure 10 - Delete Ports

Notice in the TCL console, the GUI TCL command was echoed:

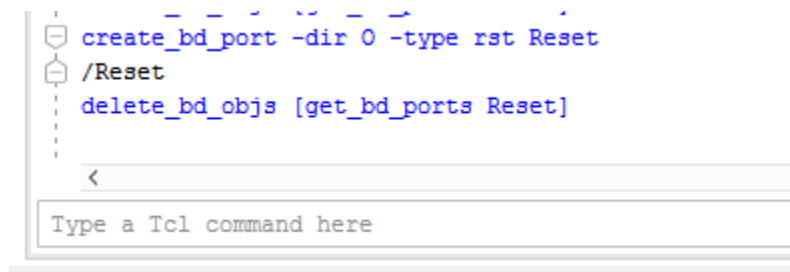
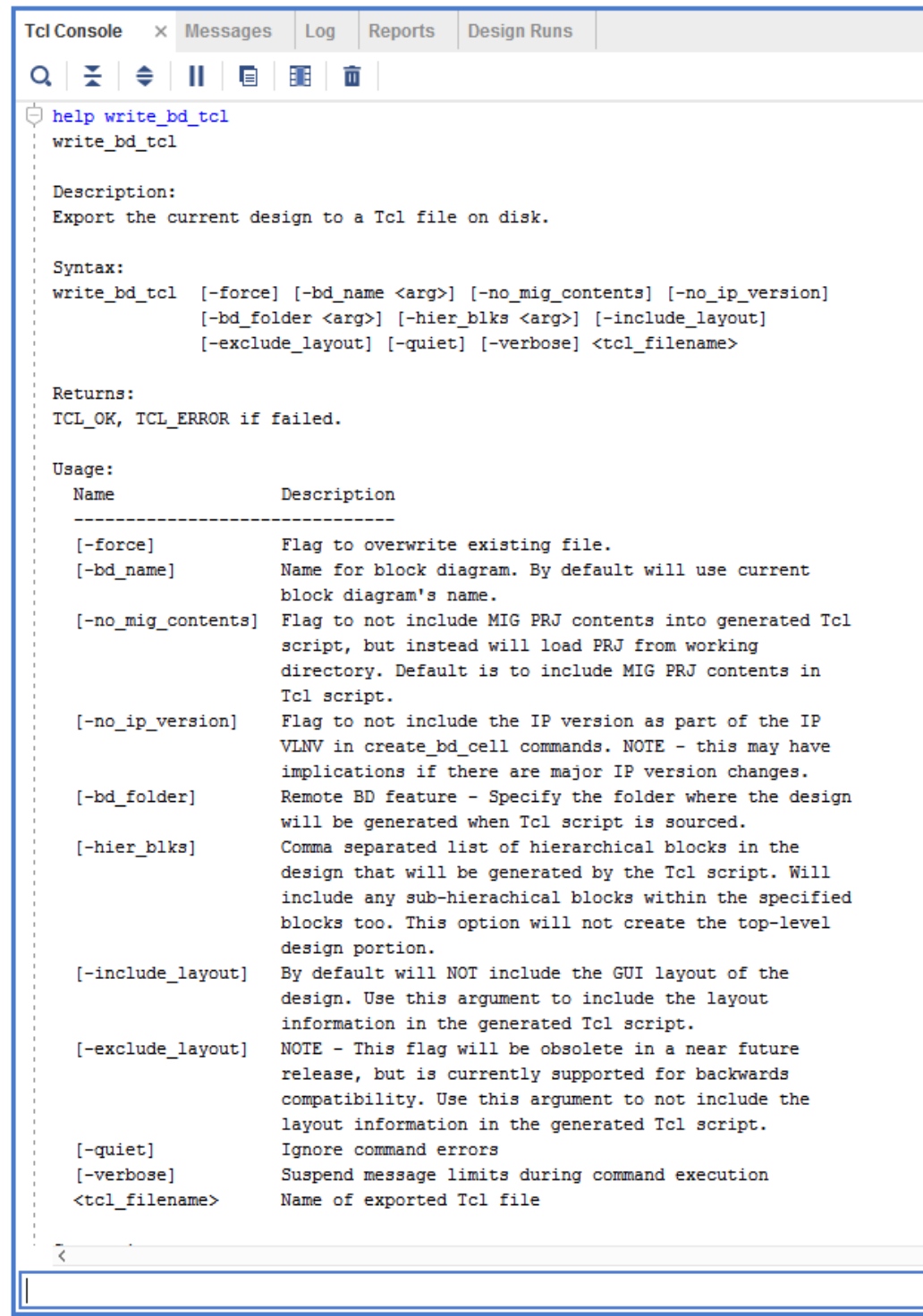


Figure 11 - TCL Console echoes GUI commands

Perhaps the most useful TCL command with regard to an IP Integrator based design is **write_bd_tcl**. This TCL command exports the current block design. Moreover, script files lets you recreate, reuse, and customize IP Integrator subsystem designs without having to archive the original subsystem design.

10. Run help on write_bd_tcl: **help write_bd_tcl**



The screenshot shows the TCL Console window with the following content:

```
Tcl Console x Messages Log Reports Design Runs
help write_bd_tcl
write_bd_tcl

Description:
Export the current design to a Tcl file on disk.

Syntax:
write_bd_tcl [-force] [-bd_name <arg>] [-no_mig_contents] [-no_ip_version]
             [-bd_folder <arg>] [-hier_blks <arg>] [-include_layout]
             [-exclude_layout] [-quiet] [-verbose] <tcl_filename>

Returns:
TCL_OK, TCL_ERROR if failed.

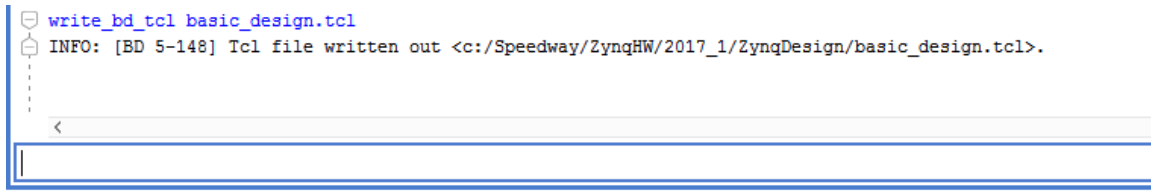
Usage:
  Name                Description
  -----
  [-force]             Flag to overwrite existing file.
  [-bd_name]           Name for block diagram. By default will use current
                      block diagram's name.
  [-no_mig_contents]   Flag to not include MIG PRJ contents into generated Tcl
                      script, but instead will load PRJ from working
                      directory. Default is to include MIG PRJ contents in
                      Tcl script.
  [-no_ip_version]     Flag to not include the IP version as part of the IP
                      VLNv in create_bd_cell commands. NOTE - this may have
                      implications if there are major IP version changes.
  [-bd_folder]         Remote BD feature - Specify the folder where the design
                      will be generated when Tcl script is sourced.
  [-hier_blks]         Comma separated list of hierarchical blocks in the
                      design that will be generated by the Tcl script. Will
                      include any sub-hierarchical blocks within the specified
                      blocks too. This option will not create the top-level
                      design portion.
  [-include_layout]    By default will NOT include the GUI layout of the
                      design. Use this argument to include the layout
                      information in the generated Tcl script.
  [-exclude_layout]    NOTE - This flag will be obsolete in a near future
                      release, but is currently supported for backwards
                      compatibility. Use this argument to not include the
                      layout information in the generated Tcl script.
  [-quiet]             Ignore command errors
  [-verbose]           Suspend message limits during command execution
  <tcl_filename>       Name of exported Tcl file
```

Figure 12 - TCL Console: help write_bd_tcl

Next, we'll export the block design using this command. Note, the block design not only encompasses all connections and IP in the design, but also the IP configurations. In this case,

the only IP is our Zynq Processing System. But at this point it is a good idea to save this as it could be the starting point for any new designs.

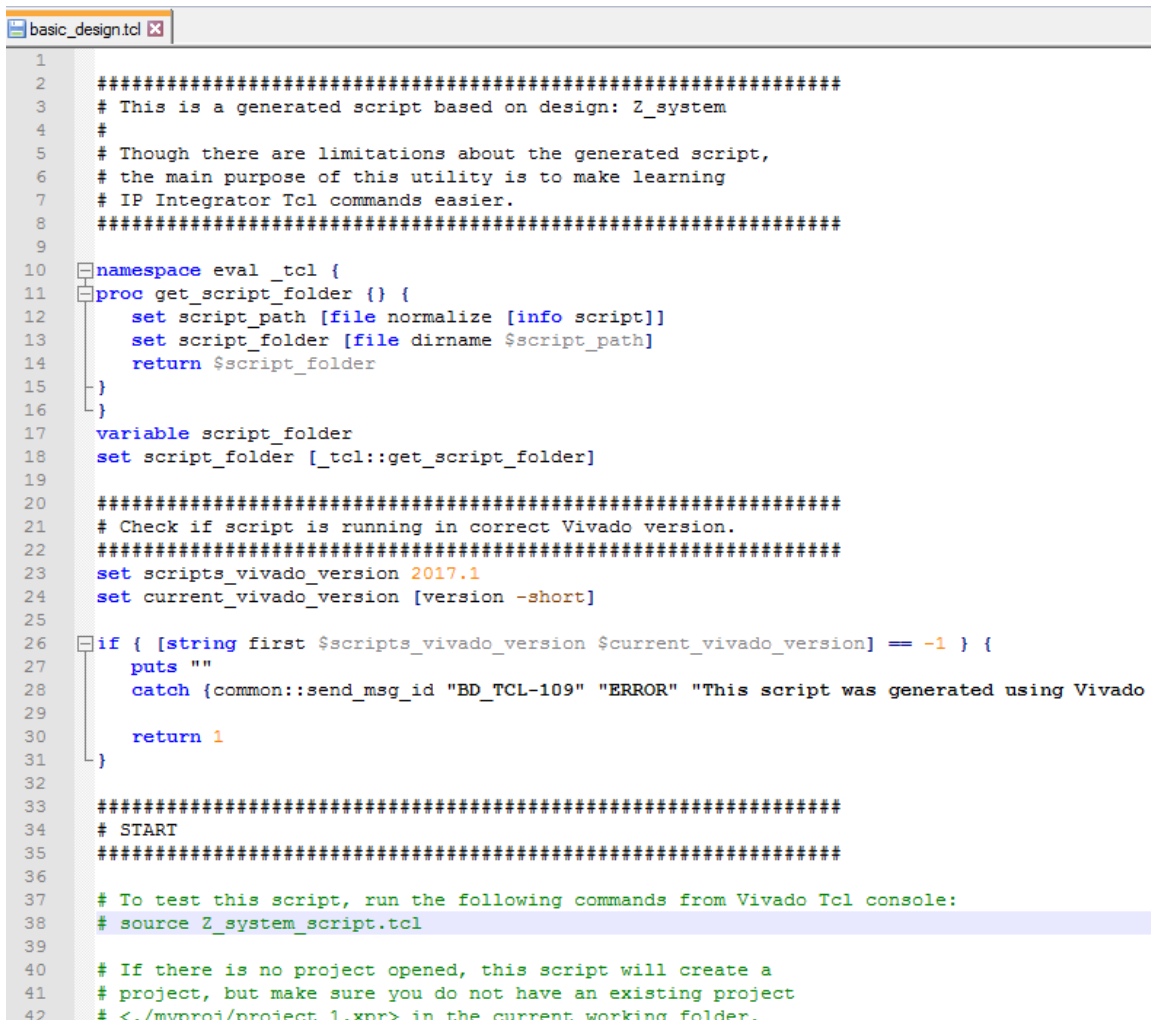
11. Run the following TCL command: **write_bd_tcl basic_design.tcl**



```
write_bd_tcl basic_design.tcl
INFO: [BD 5-148] Tcl file written out <c:/Speedway/ZynqHW/2017_1/ZynqDesign/basic_design.tcl>.
```

Figure 13 - Write out block design to TCL file

12. Using Window Explorer, **open this TCL file** with a text editor and browse through the script to see what was created.



```
1 #####
2 # This is a generated script based on design: Z_system
3 #
4 # Though there are limitations about the generated script,
5 # the main purpose of this utility is to make learning
6 # IP Integrator Tcl commands easier.
7 #####
8
9
10 namespace eval _tcl {
11   proc get_script_folder {} {
12     set script_path [file normalize [info script]]
13     set script_folder [file dirname $script_path]
14     return $script_folder
15   }
16 }
17
18 variable script_folder
19 set script_folder [_tcl::get_script_folder]
20
21 #####
22 # Check if script is running in correct Vivado version.
23 #####
24 set scripts_vivado_version 2017.1
25 set current_vivado_version [version -short]
26
27 if { [string first $scripts_vivado_version $current_vivado_version] == -1 } {
28   puts ""
29   catch {common::send_msg_id "BD_TCL-109" "ERROR" "This script was generated using Vivado .
30   return 1
31 }
32
33 #####
34 # START
35 #####
36
37 # To test this script, run the following commands from Vivado Tcl console:
38 # source Z_system_script.tcl
39
40 # If there is no project opened, this script will create a
41 # project, but make sure you do not have an existing project
42 # <./myproj/project 1.xpr> in the current working folder.
```

Figure 14 - basic_design.tcl output

13. This encapsulates our block design, but not our Project settings. Another TCL script, write_project_tcl, does that. Run help on write_project_tcl: **help write_project_tcl**.

```

Tcl Console x Messages Log Reports Design Runs
[Icons: Search, Zoom In, Zoom Out, Run, Copy, Paste, Delete]

help write_project_tcl
write_project_tcl

Description:
(User-written application)
Export Tcl script for re-creating the current project

Syntax:
write_project_tcl [-paths_relative_to <arg>] [-target_proj_dir <arg>] [-force]
                  [-all_properties] [-no_copy_sources] [-absolute_path]
                  [-dump_project_info] [-quiet] [-verbose] <file>

Returns:
true (0) if success, false (1) otherwise

Usage:
  Name                Description
  -----
  [-paths_relative_to] Override the reference directory variable for source
                        file relative paths
                        Default: Script output directory path
  [-target_proj_dir]   Directory where the project needs to be restored
                        Default: Current project directory path
  [-force]             Overwrite existing tcl script file
  [-all_properties]    Write all properties (default & non-default) for the
                        project object(s)
  [-no_copy_sources]   Do not import sources even if they were local in the
                        original project
                        Default: 1
  [-absolute_path]     Make all file paths absolute wrt the original project
                        directory
  [-dump_project_info] Write object values
  [-quiet]             Ignore command errors
  [-verbose]           Suspend message limits during command execution
  <file>               Name of the tcl script file to generate

Categories:
xilinx_tclstore, projutils, user-written
Description:

Create a Tcl script to re-create the current project. The generated script
will contain the Tcl commands for creating the project, setting the project
<
Type a Tcl command here

```

Figure 15 - TCL Console: help write_project_tcl

14. Run the following command to export the project settings:

**write_project_tcl -paths_relative_to
C:/Speedway/ZynqHW/2017_1/ZynqDesign project_setup.tcl**

15. Open the **project_setup.tcl** file with a text editor and browse through the script to see what was created.
16. Read the warning about local sources. Look for the list of required sources.
17. **Close Vivado**, you do not need to save the design.
18. Keep the TCL files open in the text editor.

Questions:

Browse the `basic_design.tcl` file and answer the following questions:

- What is the project name?

ZynqDesign

- How many interface ports do you see? What are they?

FIXED_IO, DDR

- What is the MIO Bank1 Voltage set to?

CONFIG.PCW_MIO_10_IOTYPE {LVC MOS 3.3V} \

Browse the `project_setup.tcl` file and answer the following questions:

- In what directory will the project be created?

"C:/Speedway/ZynqHW/2017_1/ZynqDesign"

- What part is selected?

set_property -name "target_part" -value "xc7z007sclg225-1" -objects \$obj

- Does the TCL file load the block design?

```
set files [list \
[file normalize "${origin_dir}/ZynqDesign.srcs/sources_1/bd/Z_system/hdl/Z_system_wrapper.vhd" ]\
]
```

- Does the TCL file reload the synthesis and implementation settings?

yes, current_run -synthesis [get_runs synth_1]

- In the `project_setup.tcl` script, what files does the script need to recreate the project?

```
create_fileset -srcset sources_1 set files [list \
[file normalize "${origin_dir}/ZynqDesign.srcs/sources_1/bd/Z_system/hdl/Z_system_wrapper.vhd" ]\
]
```

Experiment 3: TCL Test

This experiment will test the TCL commands we've just explored. We'll put these commands together to create a script that recreates our project.

Experiment 3 General Instruction:

Modify the generated TCL scripts to recreate our project

Experiment 3 Step-by-Step Instructions:

1. Open Vivado.
2. In Vivado, Type **pwd** into the TCL Console. What is the current directory?
3. Change the directory to **C:/Speedway/ZynqHW/2017_1/**

a. **cd C:/Speedway/ZynqHW/2017_1/**

As we learned in the last experiment, the `create_project.tcl` file creates the project in the current directory. So we don't overwrite our current directory, create a new directory. We've used **cd** and **pwd** to change and print working directories, thus you'd expect a **mkdir** command, but it does not exist. Instead it is a function of the **file** command.

4. Type **help file** and enter it into the Vivado TCL Console.

Scroll down and you'll see help on "**file mkdir**". This is the command we need to create a new directory.

5. Create a new directory running the following command: **file mkdir Temp**
6. Type "**cd ./**" and the TCL window will list all available directories:

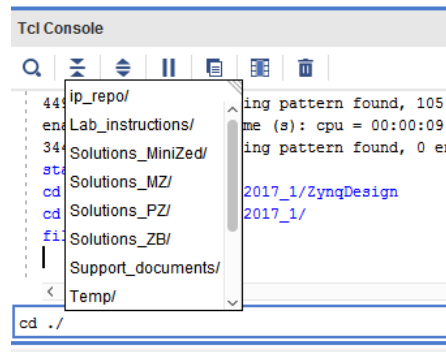


Figure 16 - Current Directory Structure

7. Click on the Temp/ directory from that menu list or enter the change directory command, **cd Temp**.

Now it's time to run the TCL script that will create our project.

8. Source the *project_setup.tcl*, which exists in the directory just above the Temp directory we just created.

source ../ZynqDesign/project_setup.tcl

The Vivado Project Manager will open:

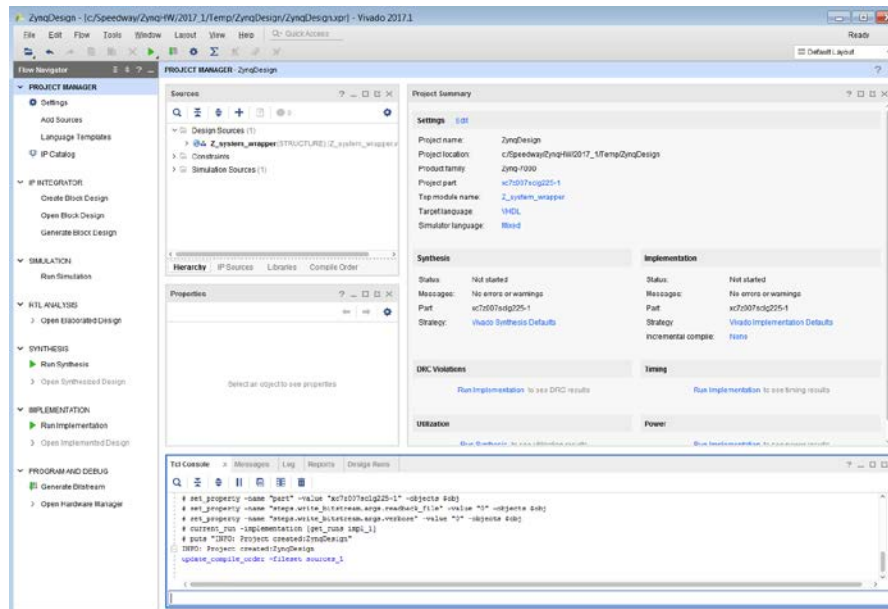


Figure 17 - Launching Vivado Project from TCL

Because we opened this on the same PC as the project was created on it was able to reference the Block Design from the original project directory as done in these TCL commands from *project_setup.tcl*:

```
58 # Set 'sources_1' fileset object
59 set obj [get_filesets sources_1]
60 # Import local files from the original project
61 set files [list \
62 "[file normalize "$origin_dir/ZynqDesign.srscs/sources_1/bd/Z_system/Z_system.bd"]" \
63 "[file normalize "$origin_dir/ZynqDesign.srscs/sources_1/bd/Z_system/hdl/Z_system_wrapper.v"]" \
64 ]
65 set imported_files [import_files -fileset sources_1 $files]
```

Figure 18 - project_setup.tcl Imports Block Design from Original Project

But if this project were opened on another PC, those files would not be available. Thus we created a separate TCL script to create the block design, *basic_design.tcl*.

9. To show this, open the block design

open_bd_design

./ZynqDesign/ZynqDesign.srscs/sources_1/bd/Z_system/Z_system.bd

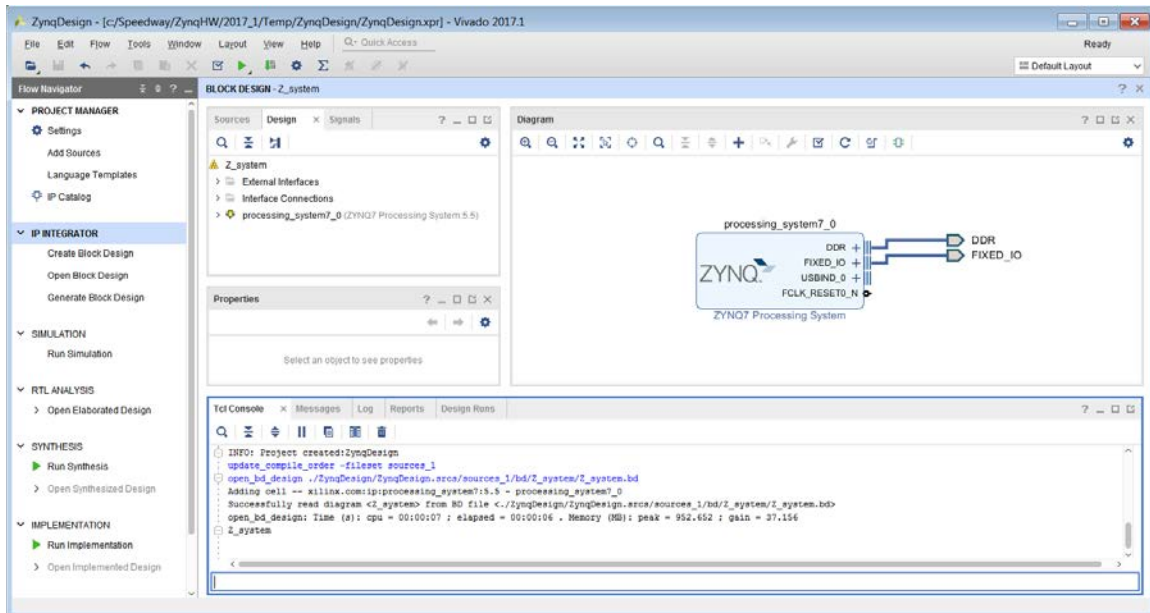


Figure 19 - Open Block Design

10. **Close** the block design by running:

```
close_bd_design [get_bd_designs Z_system]
```

11. If asked, click **Yes** to save it.

12. Delete the block design by right-clicking on it in the **Sources** view and clicking on **Remove File from Project...** Then check the **Also delete the project local file/directory from check box** and click **OK**.

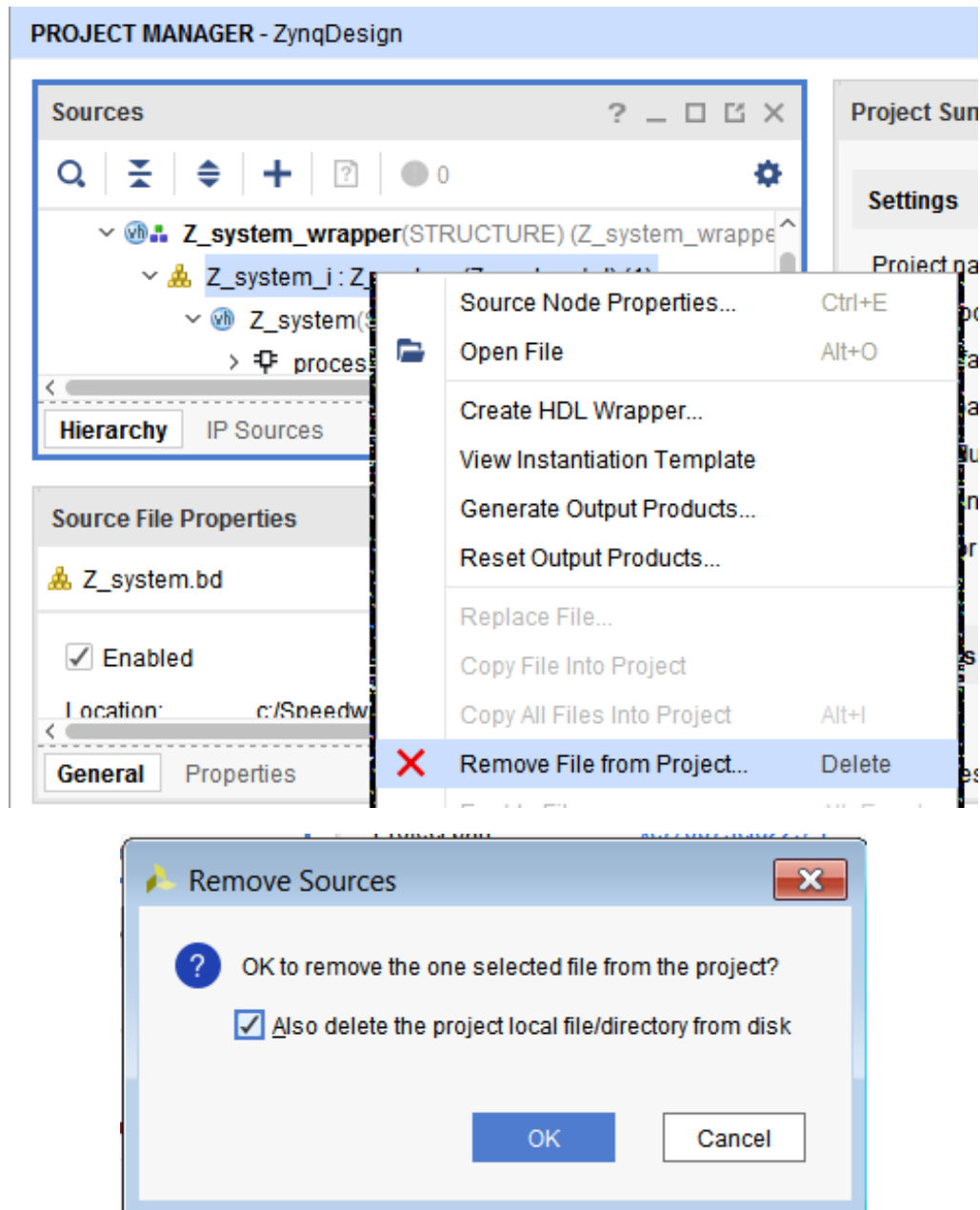


Figure 20 - Delete Block Design

Or, run the TCL commands:

```
file delete -force \
C:/Speedway/ZynqHW/2017_1/Temp/ZynqDesign/ZynqDesign.srcs/sources_1/bd/Z
_system
```

```
remove_files \
C:/Speedway/ZynqHW/2017_1/Temp/ZynqDesign/ZynqDesign.srcs/sources_1/bd/Z
_system/Z_system.bd
```


Now that the block design has been deleted and removed, it would be similar to opening our project on a different computer.

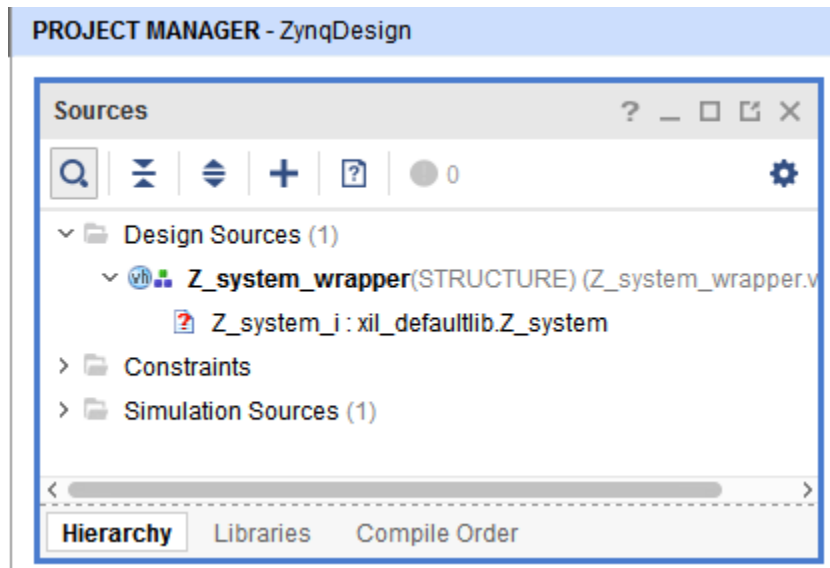


Figure 21 - Block Design Missing

13. Recreate the block design running the TCL script, **basic_design.tcl**.

source ../ZynqDesign/basic_design.tcl

The block design not only is recreated but it also opens in the GUI. Open the Zynq IP block to verify it's the same. (Note: Click Regenerate Layout to see this view)

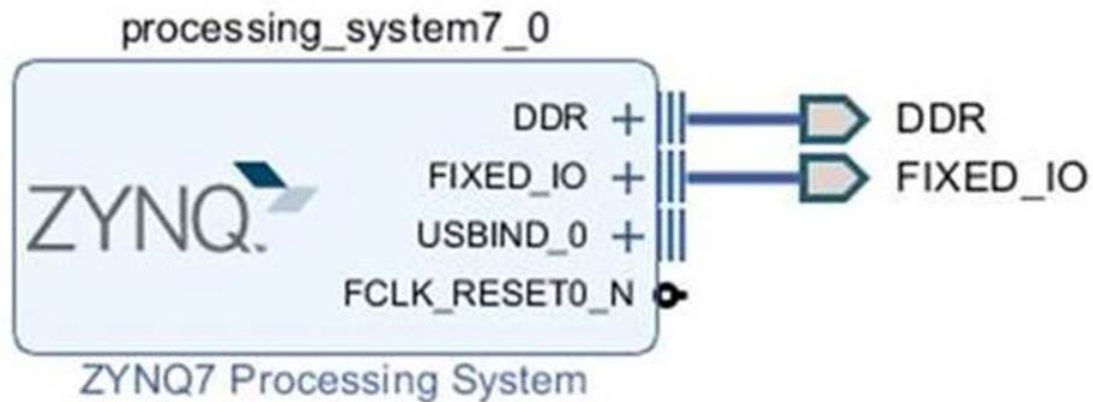


Figure 22 - Restored Block Design

14. Close the Vivado project using the TCL command **close_project**.
15. Close Vivado
16. The C:\Speedway\ZynqHW\2017_1\Temp directory can be deleted.

That concludes this lab. We now know a few basic TCL commands, how to enter them, and also how Vivado will create simple scripts that automatically restore our project!

Questions:

Browse the `basic_design.tcl` file and answer the following questions:

- Does the Zynq IP block look the same? Are all I/O peripherals mapped the same?

- Could both of the TCL files used to recreate this project be combined into one TCL file?

- What source files were needed to recreate this entire project? In other words, what do you need to archive for revision control at this point?

Exploring Further

If you have more time and would like to investigate more...

- Create a single TCL file that recreates the entire project

This concludes Lab 4.

Revision History

Date	Version	Revision
6 Nov 13	02	Initial Draft
30 Oct 14	03	Updated for Vivado 2014.3
5 Jan 15	04	Updated for Vivado 2014.4
04 Mar 15	05	Finalized for Vivado 2014.4
17 Mar 15	06	Minor edits for release
Oct 2015	07	Updated to Vivado 2015.2
July 2016	08	Updated to Vivado 2016.2
May 2017	09	Updated to Vivado 2017.1
June 2017	10	Updated to Vivado 2017.1 for MiniZed + Rebranding

Resources

www.minized.org

www.microzed.org

www.picozed.org

www.zedboard.org

www.xilinx.com/zynq

www.xilinx.com/sdk

www.xilinx.com/vivado

Answers

Experiment 2

- *What is the project name?*

CHANGE DESIGN NAME HERE

set design_name Z_system

- *How many interface ports do you see? What are they?*

2 – DDR & FIXED_IO

- *What is the MIO Bank1 Voltage set to?*

CONFIG.PCW_PRESET_BANK1_VOLTAGE {LVCMOS 3.3V}

- *In what directory will the project be created?*

This is a tough question. The first TCL command in this script sets the original project location, but that is only used to import source files. The second TCL command creates the project and does it in whatever directory the TCL script was sourced from. So before running this script, make sure you are in the directory you want the project created.

- *What part is selected?*

set_property "part" "xc7z007sclg225-1" \$obj

- *Does the TCL file load the block design?*

Yes it does:

Import local files from the original project

**set files [list **

"[file normalize "\$origin_dir/ZynqDesign.srcs/sources_1/bd/Z_system/Z_system.bd"]"

But it gets this source from our original project directory. So if run on another PC, the project will not get recreated correctly.

- *Does the TCL file reload the synthesis and implementation settings?*

Yes, see Create 'synth_1' run and Create 'impl_1' run

- *In the project_setup.tcl script, what files does the script need to recreate the project?*

ZynqDesign/ZynqDesign.srcs/sources_1/bd/Z_system/Z_system.bd

ZynqDesign/ZynqDesign.srcs/sources_1/bd/Z_system/hdl/Z_system_wrapper.v

Experiment 3

- *Does the Zynq IP block look the same? Are all I/O peripherals mapped the same?*

Yes, it is the same

- *Could both of the TCL files used to recreate this project be combined into one TCL file?*

Yes, they could simply be combined, but the `project_setup.tcl` would need to remove importing the block design source. Otherwise when the `basic_design.tcl` is run, it will already see a block design with the same name and quit running.

- *What source files were needed to recreate this entire project? In other words, what do you need to archive for revision control at this point?*

None, only the two TCL scripts are required. Per the above question, one script could be created that performs the entire project creation, block design creation, and Zynq IP Customization. That's quite powerful. Thus only the TCL file would need to be checked into a revision control system. The `system_wrapper` HDL file is not required as it can be recreated with a TCL command.