



## Project Delivery

**Authors:** \$metadata.from("Contributors")  
**Revision:** \$metadata.from("DocRevision")  
**Date:** 25.05.2016 11:23

# Table of Contents

---

Table of contents	3
Zip Project Delivery	4
Zip Name Description	4
Last supported Release	4
Directory structure	4
Windows Command Files	5
TE-TCL-Extentsions	7
Design Environment: Usage	10
Reference-Design: Getting Started	10
Basic Design Settings	11
Project Configuration	11
Initialise TE-scripts on Vivado/LabTools	12
Use predefined TE-Script functions	12
Hardware Design	13
Block Design Conventions	13
XDC Conventions	14
Backup Block Design as TCL-File	14
Software Design	14
HSI: Generate predefined software from libraries	14
SDK: Create user software project	14
Checklist / Troubleshoot	16
References	17

# Table of contents

---

- [Table of contents](#)
- [Zip Project Delivery](#)
  - [Zip Name Description](#)
  - [Last supported Release](#)
  - [Directory structure](#)
  - [Windows Command Files](#)
  - [TE-TCL-Extensions](#)
- [Design Environment: Usage](#)
  - [Reference-Design: Getting Started](#)
  - [Basic Design Settings](#)
    - [Project Configuration](#)
    - [Initialise TE-scripts on Vivado/LabTools](#)
    - [Use predefined TE-Script functions](#)
  - [Hardware Design](#)
    - [Block Design Conventions](#)
    - [XDC Conventions](#)
    - [Backup Block Design as TCL-File](#)
  - [Software Design](#)
    - [HSI: Generate predefined software from libraries](#)
    - [SDK: Create user software project](#)
- [Checklist / Troubleshoot](#)
- [References](#)

# Zip Project Delivery

## Zip Name Description

Description	PCB Name		Project Name+(opt. Variant)		supported VIVADO Version		Build Version and Date	
Example:	TE0726	-	test_board_noprebuilt	-	vivado_2015.4	-	build_26_20160415133543	.zip

## Last supported Release

Type or File	Version
Vivado Design Suite	2015.4
Trenz Project Scripts	2015.4.32
Trenz <board_series>_board_files.csv	1.2
Trenz apps_list.csv	1.7
Trenz zip_ignore_list.csv	1.0

## Directory structure

File or Directory	Type	Description
<design_name>	base directory	Base directory with predefined batch files (*.cmd) to generate or open VIVADO-Project
<design_name>/block_design/	source	Script to generate Block Design in Vivado (*_bd.tcl). (optional) Some board part designs used subfolder <board_file_shortname> with Board Part specific Block Design (*_bd.tcl).
<design_name>/board_files/	source	Local board part files repository and a list of available board part files (<board_series>_board_files.csv)
<design_name>/constraints/	source	Project constrains (*.xdc). Some board part designs used subfolder <board_file_shortname> with additional constrains (*.xdc)
<design_name>/doc/	source	Documentation
<design_name>/ip_lib/	source	Local Vivado IP repository
<design_name>/misc/	source	(Optional) Directory with additional sources
<design_name>/prebuilt/boot_images/	prebuilt	Directory with prebuilt boot images (*.bin) and configuration files (*.bif) for zynq and configured hardware files (*.bit and *.mcs) for micoblaze included in sub-folders: default or <board_file_shortname>/<app_name>

File or Directory	Type	Description
<design_name> /prebuilt /hardware/	prebuilt	Directory with prebuilt hardware sources (*.bit, *.hdf, *.mcs) and reports included in subfolders: default or <board_file_shortname>
<design_name> /prebuilt /software/	prebuilt	(Optional) Directory with prebuilt software sources (*.elf) included in subfolders: default or <board_file_shortname>/<app_name>
<design_name> /prebuilt/os/	prebuilt	(Optional) Directory with predefined OS images included in subfolders <os_name> /<board_file_shortname> or <os_name>/default
<design_name> /scripts/	source	TCL scripts to build a project
<design_name> /settings/	source	(Optional) Additional design settings: zip_ignore_list.csv, vivado project settings, SDSOC settings
<design_name> /software/	source	(Optional) Directory with additional software
<design_name> /os/	source	(Optional) Directory with additional os sources in in subfolders <os_name>
<design_name> /sw_lib/	source	(Optional) Directory with local SDK/HSI software IP repository and a list of available software (apps_list.csv)
<design_name> /v_log/	generated	(Temporary) Directory with vivado log files (used only when Vivado is started with predefined command files (*.cmd) from base folder otherwise this logs will be written into the vivado working directory)
<design_name> /vivado/	work, generated	(Temporary) Working directory where Vivado project is created. Vivado project file is <design_name>.xpr
<design_name> /vivado_lab/	work, generated	(Optional/Temporary) Working directory where Vivado LabTools is created. LabTools project file is <design_name>.lpr
<design_name> /workspace/hsi	work, generated	(Optional/Temporary) Directory where hsi project is created
<design_name> /workspace/sdk	work, generated	(Optional) Directory where sdk project is created
<design_name> /sdsoc	work, generated	(Optional) Directory where SDSOC project is created
<design_name> /backup/	generated	(Optional) Directory for project backups

## Windows Command Files

File Name	Description
<b>Design + Settings</b>	

File Name	Description
design_basic_settings.cmd	<p>Settings for the other *.cmd files. Following Settings are available:</p> <ul style="list-style-type: none"> <li>• General Settings:                             <ul style="list-style-type: none"> <li>• (optional) <b>DO_NOT_CLOSE_SHELL</b>: Shell do not closed after processing</li> <li>• (optional) <b>ZIP_PATH</b>: Set Path to installed Zip-Program. Currently 7-Zip are supported. IUsed for predefined TCL-function to Backup project.</li> <li>• (optional) <b>ENABLE_SDSOC</b>: Enable SDSOC Setting. Currently only for some reference project as beta version!</li> </ul> </li> <li>• Xilinx Setting:                             <ul style="list-style-type: none"> <li>• <b>XILDIR</b>: Set Xilinx installation path (Default: c:\Xilinx).</li> <li>• <b>VIVADO_VERSION</b>: Current Vivado/LabTool/SDK Version (Example:2015.4). Don't change Vivado Version.                                     <ul style="list-style-type: none"> <li>• Xilinx Software will be searched in:</li> <li>• VIVADO (optional for project creation and programming): %XILDIR%\Vivado%\VIVADO_VERSION%</li> <li>• SDK (optional for software projects and programming): %XILDIR%\SDK%\VIVADO_VERSION%</li> <li>• LabTools (optional for programming only): %XILDIR%\Vivado_Lab%\VIVADO_VERSION%</li> <li>• SDSOC (optional): %XILDIR%\SDSOC%\VIVADO_VERSION%</li> </ul> </li> </ul> </li> <li>• Board Setting:                             <ul style="list-style-type: none"> <li>• <b>PARTNUMBER</b>: Set Board part number of the project which should be created                                     <ul style="list-style-type: none"> <li>• Available Numbers: (you can use ID,PRODID,BOARDNAME or SHORTNAME from TExxx_board_file.csv list)</li> <li>• Used for project creation and programming</li> <li>• To create empty project without board part, used PARTNUMBER=-1 (use GUI to create your project. No block design tcl-file should be in /block_design)</li> <li>• Example TE0726 Module :</li> <li>• USE ID  USE PRODID  Use Boardname  Use Shortname PARTNUMBER=1 PARTNUMBER=te0726-01  PARTNUMBER=trenz.biz:te0726-01:part0:1.0  PARTNUMBER=TE0726-01</li> </ul> </li> </ul> </li> <li>• Programming Settings(program*file.cmd):                             <ul style="list-style-type: none"> <li>• <b>SWAPP</b>: Select Software App, which should be configured.                                     <ul style="list-style-type: none"> <li>• Use the folder name of the &lt;design_name&gt;/prebuilt/boot_image/&lt;partname&gt;/* subfolder. The *.bin,*.mcs or *.bit from this folder will be used.</li> <li>• If you will configure the raw *.bit or *.mcs *.bin from the &lt;design_name&gt;/prebuilt /hardware/&lt;partname&gt;/ folder, use @set SWAPP=NA or @set SWAPP="".</li> <li>• Example: SWAPP=hello_world used the file from prebuilt/boot_image/&lt;partname&gt;/hello_world SWAPP=NA used the file from &lt;design_name&gt;/prebuilt/boot_image/&lt;partname&gt;/</li> </ul> </li> <li>• <b>PROGRAM_ROOT_FOLDER_FILE</b>: If you want to program design file from the rootfolder &lt;design_name&gt;, set to 1                                     <ul style="list-style-type: none"> <li>• Attention: it should be only one *.bit, *.mcs or *.bin file in the root folder.</li> </ul> </li> </ul> </li> </ul>
design_clear_design_folders.cmd	(optional) Attention: Delete "<design_name>/v_log/", "<design_name>/vivado/", "<design_name>/vivado_lab/", "<design_name>/sdsoc/", and "<design_name>/workspace/" directory with related documents!
design_run_project_batchmode.cmd	(optional) Create Project with setting from "design_basic_settings.cmd" and source folders. Build all Vivado hardware and software files if the sources are available.  Delete "<design_name>/vivado/", and "<design_name>/workspace/hsi/" directory with related documents before Projekt will created.
<b>Hardware Design</b>	
vivado_create_project_guiemode.cmd	Create Project with setting from "design_basic_settings.cmd" and source folders. Vivado GUI will be opened during the process.  Delete "<design_name>/vivado/", and "<design_name>/workspace/" directory with related documents before Projekt will created.

File Name	Description
vivado_create_project_batchmode.cmd	(optional) Create Project with setting from "design_basic_settings.cmd" and source folders.  Delete "<design_name>/vivado/", and "<design_name>/workspace/" directory with related documents before Projekt will created.
vivado_open_existing_project_guiemode.cmd	Opens an existing Project "<design_name>/vivado/<design_name>.xpr" and restore Script-Variables.
<b>Software Design</b>	
sdk_create_prebuilt_project_guiemode.cmd	(optional) Create SDK project with hardware definition file from prebuild folder. It used the *.hdf from: <design_name>/prebuilt/hardware/<board_file_shortcode>/. Set <board_file_shortcode> and <app_name> in "design_basic_settings.cmd".
<b>Programming</b>	
program_flash_binfile.cmd	(optional) For Zynq Systems only. Programming Flash Memory via JTAG with specified Boot.bin. Used SDK Programmer (Same as SDK "Program Flash") or LabTools Programmer (Vivado or LabTools only), depends on installation settings. Default, it used the boot.bin from: <design_name>/prebuilt/boot_images/<board_file_shortcode>/<app_name>. Settings are done in "design_basic_settings.cmd".
program_flash_mcsfile.cmd	(optional) For Non-Zynq Systems only. Programming Flash Memory via JTAG with specified <design_name>.mcs. Used LabTools Programmer (Vivado or LabTools only), depends on installation settings. Default, it used the <design_name>.mcs from: <design_name>/prebuilt/hardware/<board_file_shortcode>. Settings are done in "design_basic_settings.cmd".
program_fpga_bitfile.cmd	(optional) Programming FPGA via JTAG with specified <design_name>.bit. Used LabTools Programmer (Vivado or LabTools only), depends on installation settings. Default, it used the <design_name>.bit from: <design_name>/prebuilt/hardware/<board_file_shortcode>. Settings are done in "design_basic_settings.cmd".
labtools_open_project_guiemode.cmd	(optional) Create or open an existing Vivado Lab Tools Project. (Additional TCL functions from Programming and Utilities Group are usable). Settings are done in "design_basic_settings.cmd".

## TE-TCL-Extentsions

Name	Options	Description (Default Configuration)
TE::help		Display currently available functions. Important: Use only displayed functions and no functions from sub-namespaces
<b>Hardware Design</b>		
TE::hw_blockdesign_export_tcl	[-no_mig_contents] [-no_validate] [-mod_tcl] [-svntxt <arg>] [-board_part_only] [-help]	Export Block Design to project folder <design_name>/block_design/. Old *bd.tcl will be overwritten!
TE::hw_build_design	[-export_prebuilt] [-export_prebuilt_only] [-help]	Run Synthese, Implement, and generate Bit-file, optional MCS-file and some report files
<b>Software Design</b>		

Name	Options	Description (Default Configuration)
TE::sw_run_hsi	[-run_only] [-prebuilt_hdf <arg>] [-no_hsi] [-no_bif] [-no_bin] [-no_bitmcs] [-clear] [-help]	Copies current Hardware files and reports from the vivado project to the prebuilt folder, if -prebuild_hdf <arg> isn't set. Copy the Hardware Definition file to the working directory: <design_name>/workspace/hsi Run HSI in <design_name>/workspace/hsi for all Programmes listed in <design_name>/sw_lib/apps_list.csv If HSI is finished, BIF-GEN and BIN-Gen are running for these Apps in the prebuilt folders <design_name>/prebuilt/... You can deactivate different steps with following args : <ul style="list-style-type: none"> <li>• -no_hsi : *.elf filesgeneration is disabled</li> <li>• -no_bif : *.bif files generation is disabled</li> <li>• -no_bin : *.bin files generation is disabled</li> <li>• -no_bitmcs: *.bit and *.mcs file (with software design) is disabled</li> </ul>
TE::sw_run_sdk	[-open_only] [-update_hdf_only] [-prebuilt_hdf <arg>] [-clear] [-help]	Copies current Hardware files and reports from the vivado project to the prebuilt folder, if -prebuild_hdf <arg> isn't set. Copy the Hardware Definition file to the working directory: <design_name>/workspace/sdk Start SDK GUI in this workspace
<b>Programming</b>		
TE::pr_init_hardware_manager	[-help]	Open Hardwaremanager, autoconnect target device and initialise flash memory with configuration from *_board_files.csv.
TE::pr_program_jtag_bitfile	[-used_board <arg>] [-swapp <arg>] [-available_apps] [-used_basefolder_bitfile] [-help]	Copies current Hardware files and reports from the vivado project to the prebuilt folder, if -used_board <arg> isn't set (Vivado only). Programming Bitfile from <design_name>/prebuilt/hardware/<board_file_shortname> to the fpga device. If "-used_basefolder_bitfile" is set, the Bitfile (*.bit) from the base folder (<design_name>) is used instead of the prebuilts. Attention: Take only one Bitfile in the basefolder!  (MicroBlaze only) If "-swapp" is set, the Bitfile with *.elf configuration is used from <design_name>/prebuilt/boot_images/<board_file_shortname>/<app_name>
TE::pr_program_flash_binfile	[-no_reboot] [-used_board <arg>] [-swapp <arg>] [-available_apps] [-force_hw_manager] [-used_basefolder_binfile] [-help]	Attention: For Zynq Systems only! Program the Bootbin from <design_name>/prebuilt/boot_images/<board_file_shortname>/<app_name> to the fpga device. Appname is selected with: -swapp <app_name> After programming device reboot from memory will be done. Default SDK Programmer is used, if not available LabTools Programmer is used. If "-used_basefolder_binfile" is set, the Binfile (*.bin) from the base folder (<design_name>) is used instead of the prebuilts. Attention: Take only one Binfile in the basefolder!



Name	Options	Description (Default Configuration)
TE:: pr_program_flash_mcsfile	[-no_reboot] [-used_board <arg>] [-swapp <arg>] [-available_apps] [-used_basefolder_mcsfile] [-help]	Copies current Hardware files and reports from the vivado project to the prebuilt folder, if -used_board <arg> isn't set (Vivado only). Initialise flash memory with configuration from *_board_files.csv Programming MCSfile from <design_name>/prebuilt/hardware /<board_file_shortname> to the Flash Device. After programming device reboot from memory will be done. If "-used_basefolder_binfile" is set, the MCSfile (*.mcs) from the base folder (<design_name>) is used instead of the prebuilts. Attention: Take only one MCSfile in the basefolder!  (MicroBlaze only) If "-swapp" is set, the MCSfile with *.elf configuration is used from <design_name>/prebuilt /boot_images/<board_file_shortname>/<app_name>
<b>Utilities</b>		
TE::util_zip_project	[-save_all] [-remove_prebuilt] [-manual_filename <arg>] [-help]	Make a Backup from your Project in <design_name>/backup/ Zip-Program Variable must be set in start_settings.cmd. Currently only 7-Zip is supported.
<b>Beta Test (Advanced usage only!)</b>		
TE::ADV:: beta_util_sdsoc_project	[-check_only] [-start_sdsoc_only] [-help]	Create SDSOC-Workspace. Currently only on some Reference-Designs available. Run [-check_only] option to check SDSOC ready state.
TE::ADV:: beta_hw_remove_board_part	[-permanent] [-help]	Reconfigure Vivado project as project without board part. Generate XDC-File from board part IO definitions and change ip board part properties. No all IPs are supported.
TE::ADV:: beta_hw_export_rtl_ip	\[-help\]	Save IPs used on rtl designs as *.xci in <design_name>hdl/xci. If sub folder <board_file_shortname> is defined this will be saved there.

# Design Environment: Usage

---

## Reference-Design: Getting Started

---

- Install **Xilinx Vivado Design Suite** or **Xilinx Vivado Webpack** (free license for some FPGA only: see <http://www.xilinx.com/products/design-tools/vivado/vivado-webkit.html>)  
(optional) Install **Xilinx Vivado LabTools** (Lab Edition)
- Configure the reference-design:
  1. Open “ **design\_basic\_settings.cmd** ” with a text-editor:
    - a. Set correct Xilinx Environment:  
**@set XILDIR=C:/Xilinx**  
**@set VIVADO\_VERSION=2015.4**  
Program settings will be search in :  
%XILDIR%/VIVADO/%VIVADO\_VERSION%/   
%XILDIR%/Vivado\_Lab/%VIVADO\_VERSION%/   
%XILDIR%/SDK/%VIVADO\_VERSION%/   
Example directory: c:/Xilinx/Vivado/2015.4/  
**Attention:** Scripts are supported only with predefined Vivado Version!
    - b. Set the correct module part-number:  
**@set PARTNUMBER=x**  
You found the available Module Numbers in [<design\\_name>/board\\_files/<board\\_series>\\_board\\_files.csv](#)
    - c. Set Application name (for programming with batch-files only):  
**@set SWAPP=NA**  
NA (No Software Project) used \*.bit or \*.mcs from [<design\\_name>/prebuilt/hardware/<board\\_file\\_shortcode>](#)  
<app\_name> (Software Project) used \*.bit or \*.mcs or \*.bin from [<design\\_name>/prebuilt/boot\\_images/<board\\_file\\_shortcode>/<app\\_name>](#)
  2. Run “ **design\_run\_project\_batchmode.cmd** ”
- (optional to Step 2) Create all prebuilt files in single steps:
  3. Run “ **vivado\_create\_project\_guiemode.cmd** ”:  
A Vivado Project will be create and open in ./vivado
  4. Type “ **TE::hw\_build\_design** ” on Vivado TCL-Console:  
Run Synthese, Implement and create Bitfile and optional MCSfile
  5. Type “ **TE::sw\_run\_hsi** ” on Vivado TCL-Console:  
Create all Software Applications from [<design\\_name> /sw\\_lib/apps\\_list.csv](#)
  6. (optional to Step 5) Type “ **TE::sw\_run\_sdk** ” on Vivado TCL-Console:  
Create a SDK Project in [<design\\_name>/workspace/sdk](#)  
Include Hardware-Definition-File, Bit-file and local Software-libraries from [<design\\_name>/sw\\_lib/sw\\_apps](#)

- Programming FPGA or Flash Memory with prebuilt Files:
  7. Connect your Hardware-Modul with PC via JTAG.  
With Batch-file:
    8. (optional) Zynq-Devices Flash Programming (\*.bin):  
Run “ **program\_flash\_binfile.cmd** ”
    9. (optional) FPGA-Device Flash Programming (\*.mcs):  
Run “ **program\_flash\_mcsfile.cmd** ”
    10. (optional) FPGA-Device Programming (\*.bit):  
Run “ **program\_fpga\_bitfile.cmd** ”
  - With Vivado/Labtools TCL-Console:
    11. Run “ **vivado\_open\_existing\_project\_guiemode.cmd** ” or “ **labtools\_open\_project\_guiemode.cmd** ” to open Vivado or LabTools
    12. (optional) Zynq-Devices Flash Programming (\*.bin):  
Type “ **TE::pr\_program\_flash\_binfile -swap <app\_name>** ” on Vivado TCL-Console  
Used \*.bin from <design\_name>/prebuilt/boot\_images/<board\_file\_shortcode>/<app\_name>
    13. (optional) FPGA-Device Flash Programming (\*.mcs):  
Type “ **TE:: pr\_program\_flash\_mcsfile -swap <app\_name>** ” on Vivado TCL-Console  
Used \*.mcs from <design\_name>/prebuilt/boot\_images/<board\_file\_shortcode>/<app\_name>
    14. (optional) FPGA-Device Programming (\*.bit):  
Type “ **TE:: pr\_program\_jtag\_bitfile -swap <app\_name>** ” on Vivado TCL-Console  
Used \*.bit from <design\_name>/prebuilt/boot\_images/<board\_file\_shortcode>/<app\_name>


## Basic Design Settings

---

### Project Configuration

1. Unzip project files
2. Rename basefolder (basefolder name is used as project name)
3. Edit **design\_basic\_settings.cmd**
  - a. Select the correct Xilinx Program path (See: **Windows Command Files design\_basic\_settings .cmd** )
  - b. Select the correct board part number for your PCB (See: **Windows Command Files design\_basic\_settings .cmd** )
  - c. Other settings are optional (See: **Windows Command Files design\_basic\_settings .cmd** )
4. Execute **vivado\_create\_project\_guiemode.cmd** or **vivado\_create\_project\_batchmode.cmd** to generate a vivado project with the predefined Block Design from the Block Design folder
5. Open Vivado with **vivado\_open\_existing\_project\_guiemode.cmd** (if you use **vivado\_create\_project\_guiemode .cmd** on step 4, you didn't need this)
6. Open the Block Design and create your own design inside this Block Design.
7. Backup your Block Design as tcl-script: Type “ **TE::hw\_blockdesign\_export\_tcl** ” on Vivado Tcl Console. The old one will be overwritten.
8. Build your Design...

## Initialise TE-scripts on Vivado/LabTools

- Variant 1 (recommended):
  - Start the project with the predefined command file ( **vivado\_open\_existing\_project\_guiemode.cmd** ) respectively LabTools with ( **labtools\_open\_project\_guiemode.cmd** )
- Variant 2:
  - Create your own Initialisation Button on the Vivado GUI:
    - Tools Customize Commands Customize Commands...
    - Push 
    - Type Name ex.: Init Scripts
    - Press Enter
    - Select Run command and insert :
      - for Vivado: `cd [get_property DIRECTORY [current_project]]; source -notrace "../scripts/reinitialise_all.tcl"`
      - for LabTool: `cd [pwd]; source -notrace "../scripts/reinitialise_all.tcl"`
    - Press Enter
    - A new Button is shown on the Vivado Gui: All Scripts are reinitialised, if you press this Button.
- Variant 3:
  - Reinitialise Script on Vivado TCL-Console:
    - Type: `source ../scripts/reinitialise_all.tcl`

## Use predefined TE-Script functions

- Variant 1 (recommended):
  - Typ function on Vivado TCL Console, ex.: `TE::help`
  - `TE::help`
    - Show all predefined TE-Script functions.
  - `TE:<functionname> -help`
    - Show short description of this function.
    - **Attention** : If `-help` argument is set, all other args will be ignored.

- Variant 2:
  - Create your own function Button on the Vivado GUI:
    - Tools Customize Commands Customize Commands...
    - Push +
    - Type Name ex.: Run SDK
    - Press Enter
    - Select Run command and insert function:
      - Variante 1 (no Vivado request window for args):
        - insert function and used args, ex.: TE::sw\_program\_zynq -swapp hello\_world
      - Variant 2 ( Vivado request window for args ):
        - insert function, ex.: TE::sw\_program\_zynq
        - Press Define Args...
        - For every arg:
          - Push +
          - Typ Name, Comment, Default Value and set optional
          - Press Enter
          - Example for args:
            - Push +
            - Index, Key Name, -swapp, ✓
            - Push +
            - Appname, Arg, hello\_world, ✓

## Hardware Design

---

### Block Design Conventions

- Only one Block-Design per project is supported
- Recommended BD-Names (currently importend for some TE-Scripts):

Name	Description
zsys	Identify project as Zynq Project with processor system (longer name with *zsys* are supported too)
zusys	Identify project as UltraScaleZynq Project with processor system (longer name with *zusys* are supported too)
msys	Identify project as Microblaze Project with processor system (longer name with *msys* are supported too)
fsys	Identify project as FPGA-fabric Project without processor system (longer name with *fsys* are supported too)

## XDC Conventions

- All \*.xdc from <design\_name>/constrains/ are load into the vivado project on project creation.  
**Attention** : If subfolder <design\_name>/constrains/ <board\_file\_shortcode> is defined, it will be used the subfolder constrains only for this module!
- Recommended XDC-Names (used for Vivado XDC-options):

Property	Name part	Description
Set Processing Order	*_e_*	set to early
	*_l_*	set to late
		set to normal
Set Used In	*_s_*	used in synthese only
	*_i_*	used in implement only
		used in both, synthese and implement

## Backup Block Design as TCL-File

- Backup your Block-Design with TCL-Command " **TE::hw\_blockdesign\_export\_tcl** " in <design\_name>/block\_design/  
 It will be saved as \*\_bd.tcl  
**Attention** : If subfolder <design\_name>/block\_design/<board\_file\_shortcode> is defined, it will be saved there!  
 Only one \*.tcl file should be in the backup folder respectively the subfolder <board\_file\_shortcode>

## Software Design

---

### HSI: Generate predefined software from libraries

- To generate predefined software from libraries, run " **TE::sw\_run\_hsi** " on Vivado TCL-Console
- All programs in in <design\_name>/sw\_lib/apps\_list.csv are generated automatically
- Supported are local application libraries from <design\_name>/sw\_lib/sw\_apps or the most Xilinx SDK Applications found in %XILDIR%/SDK/%VIVADO\_VERSION%/data/embeddedsw/lib/sw\_app

### SDK: Create user software project

- To start SDK project, run " **TE::sw\_run\_sdk** " on Vivado TCL-Console  
 Include Hardware-Definition-File, Bit-file and local Software-libraries from <design\_name>/sw\_lib/sw\_apps

- To use Hardware-Definition-File, Bit-file from prebuilt folder without building the vivado hardware project, run " **sdk\_create\_prebuilt\_project\_gui\_mode.cmd** " or type " **TE::sw\_run\_sdk -prebuilt\_hdf <board\_number>** " on Vivado-TCL-Console
- To open an existing SDK-project without update HDF-Data, type " **TE::sw\_run\_sdk -open\_only** " on Vivado-TCL-Console

## Checklist / Troubleshoot

---

1. Are you using exactly the same Vivado version? If not then the scripts will not work, no need to try.
2. Are you using Vivado in Windows PC? Vivado works in Linux also, but the scripts are tested on Windows only.
3. Is your PC OS installation English? Vivado may work on national versions also, but there have been known problems.
4. Are there space characters on the project path? Sometimes TCL-Scripts can't handle this correctly. Remove spaces from the project path.
5. Did you have the newest reference design build version? Maybe it's only a bug from an older version.
6. Check `<design_name>/v_log/vivado.log`? If no logfile exists, wrong Xilinx paths are set in `design_basic_settings.cmd`
7. If nothing helps, send a mail to Trenz support ([support\(at\)trenz-electronic.de](mailto:support(at)trenz-electronic.de)) with subject line "[TE-Reference Designs]", the complete zip-name from your reference design and the last log file (`<design_name>/v_log/vivado.log`)



## References

---

1. Vivado Design Suite User Guide - Getting Started (UG910)
2. Vivado Design Suite User Guide - Using the Vivado IDE (UG893)
3. Zynq-7000 All Programmable SoC Software Developers Guide (UG821)
4. SDSoC Environment User Guide - Getting Started (UG1028)
5. SDSoC Environment - User Guide (UG1027)
6. SDSoC Environment User Guide - Platforms and Libraries (UG1146)