Project Delivery - Xilinx devices

Table of contents

Quick Start

The most Trenz Electronic FPGA Reference Designs are TCL-script based project.

The "normal" Vivado project will be generated in the subfolder "/vivado/" after executing scripts (YouTube: TE0720 Project Creation).

There are several options to create the Vivado project from the project delivery. These options are described in Vivado Projects.

Since 2018.3 special "Module Selection Guide" is included into "_create_win_setup.cmd" and "_create_linux_setup.sh"

- Execute "_create_win_setup.cmd" or "_create_linux_setup.sh"
- Select "Module Selection Guide" (press "0" and Enter)
- Follow instructions

For manuell configuration or addition command files for execution will be generated with "_create_win_setup.cmd" on Windows OS and "_create_linux_setup.sh" on Linux OS. If you use our prepared batch files for project creation do the following steps:

1. open "design_basic_settings.cmd/.sh" with text editor and set correct vivado path and board part number (this will be also done automatically with the "Module Selection Guide"). **How select the correct board part number is described on** TE Board Part Files
2. run "vivado_create_project_guimode.cmd/.sh"

See Reference Design: Getting Started for more details.

If you need our Board Part files only, see Board Part Installation.

For Problems, please check Checklist / Troubleshoot at first.

Zip Project Delivery

Zip Name Description

Description	PCB Name		Project Name+(opt. Variant)		supported VIVADO Version		Build Version and Date	
Example:	te0715	-	-test_board(_noprebuilt)	-	vivado_2017.4	-	build_01_20180105195436	.zip

Last supported Release

Type or File	Version
Vivado Design Suite	2018.3
Trenz Project Scripts	2018.3.09
Trenz <board_series>_board_files.csv	1.4
Trenz apps_list.csv	2.2
Trenz zip_ignore_list.csv	1.0
Trenz mod_bd.csv (not included)	1.1

Currently limitations of functionality

- Important Note: QSPI Programming need special FSBL on 2017.4 or higher for all Zynq/ZynqMP. Special programming FSBL will be provided on all newer reference designs
- Linux OS only: HSI software generation failed.
  - Reason: start "gmake" failed, alias is not set on Ubuntu
  - Workaround: "sudo ln -s /usr/bin/make /usr/bin/gmake" to generate alias or use SDK GUI to generate applications and boot files.
- Linux OS only: Function, which used external programs.
  - Reason: Currently only set correctly for Win OS.
  - Workaround: Change TCL scripts program path manually.

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> /board_files /carrier_extension	source	(Optional) Additional TCL-Scripts to extend Board Part PS-Preset with carrier board specific settings.
<design_name> /console	source	folder with different console command files. Use _create_win_setup.cmd or _create_linux_setup.sh to generate files on top folder.
<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>/hdl/	source	HDL-File and XCI-Files. Advanced usage only!
<design_name> /firmware/	source	ELF-File Location for MicroBlaze Firmware. Additional sub folder is used for MicroBlaze identification.
<design_name>/ip_lib/	source	Local Vivado IP repository
<design_name>/misc/	source	(Optional) Directory with additional sources
<design_name> /prebuilt/	prebuilt	Contains a readme with location information of different assembly variants
<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>
<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 writen 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_PFM	work, generated	(Optional) Directory where SDSOC project is created
<design_name> /backup/	generated	(Optional) Directory for project backups

Command Files

Command files will be generated with "_create_win_setup.cmd" on Windows and "_create_linux_setup.sh" on Linux OS. Linux shell files are currently not available for this release.

Windows Command Files

File Name	Description
Design + Settings	
_create_win_setup.cmd	Use to create bash files. With 2018.3 and newer also "Module Selection Guide" is included
_use_virtual_drive.cmd	(Option) Create virtual drive for project execution. See Xilinx AR#52787

design_basic_settings.cmd	Settings for the other *.cmd files. Following Settings are avaliable:
	• General Settings:
	  • (optional) **DO_NOT_CLOSE_SHELL**: Shell do not closed after processing
	  • (optional) **ZIP_PATH**: Set Path to installed Zip-Program. Currently 7-Zip are supported. IUsed for predefined TCL-function to Backup project.
	  • (optional) **ENABLE_SDSOC**: Enable SDSOC Setting. Currently only for some reference project as beta version!
	• Xilinx Setting:
	  • **XILDIR**: Set Xilinx installation path (Default: c:\Xilinx).
	  • **VIVADO_VERSION**: Current Vivado/LabTool/SDK Version (Example:2017.4). Don't change Vivado Version.
	  • Xilinx Software will be searched in:
	  • VIVADO (optional for project creation and programming): %XILDIR%\Vivado\%VIVADO_VERSION%\ and for SDSoC on %XILDIR%\SDx\%VIVADO_VERSION%\Vivado\
	  • SDK (optional for software projects and programming): %XILDIR%\SDK\%VIVADO_VERSION%\
	  • LabTools (optional for programming only): %XILDIR%\Vivado_Lab\%VIVADO_VERSION%\
	  • SDSOC (optional): %XILDIR%\SDx\%VIVADO_VERSION%\
	• Board Setting:
	  • **PARTNUMBER**: Set Board part number of the project which should be created
	  • Available Numbers: (you can use ID,PRODID,BOARDNAME or SHORTNAME from TExxxx_board_file.csv list)
	  • Used for project creation and programming
	  • 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)
	  • Example TE0726 Module :
	  • USE ID |USE PRODID
	  PARTNUMBER=1 |PARTNUMBER=te0726-01
	• Programming Settings(program*file.cmd):
	  • **SWAPP**: Select Software App, which should be configured.
	  • Use the folder name of the <design_name>/prebuilt/boot_image/<partname>/* subfolder. The *bin,*.mcs or *.bit from this folder will be used.
	  • If you will configure the raw *.bit or *.mcs *.bin from the <design_name>/prebuilt/hardware/<partname>/ folder, use @set SWAPP=NA or @set SWAPP="".
	  • Example: SWAPP=hello_world used the file from prebuilt/boot_image/<partname>/hello_world
	  SWAPP=NA used the file from <design_name>/prebuilt/boot_image/<partname>/
	  • **PROGRAM_ROOT_FOLDER_FILE**: If you want to program design file from the rootfolder <design_name>, set to 1
	  • Attention: it should be only one *.bit, *.msc or *.bin file in the root folder.

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! Type "Y" into the command line input to start deleting files
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 Project will created.
Hardware Design	
vivado_create_project_guimode.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 Project will created. If old vivado project exists, type "y" into the command line input to start project creation again.
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 Project will created. If old vivado project exists, type "y" into the command line input to start project creation again.
vivado_open_existing_project_guimode.cmd	Opens an existing Project "<design_name>/vivado/<design_name>.xpr" and restore Script-Variables.
Software Design	

sdk_create_prebuilt_project_guimode.cmd	(optional) Create SDK project with hardware definition file from prebuild folder. It used the *.hdf from: <design_name>/prebuilt/hardware/<board_file_shortname>/. Set <board_file_shortname> and <app_name> in "design_basic_settings.cmd".
Programming	
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 installion settings. Default, it used the boot.bin from: <design_name>/prebuilt/boot_images/<board_file_shortname>/<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 installion settings. Default, it used the <design_name>.mcs from: <design_name>/prebuilt/hardware/<board_file_shortname>. 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 installion settings. Default, it used the <design_name>.bit from: <design_name>/prebuilt/hardware/<board_file_shortname>. Settings are done in "design_basic_settings.cmd".
labtools_open_project_guimode.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".

Linux Command Files

File Name	Status	Description
Design + Settings		
_create_linux_setup.sh	available	Use to create bash files. With 2018.3 and newer also "Module Selection Guide" is included

design_basic_settings.sh	available	Settings for the other *.cmd files. Following Settings are avaliable:
		• General Settings:
		  • (optional) **DO_NOT_CLOSE_SHELL**: Shell do not closed after processing
		  • (optional) **ZIP_PATH**: Set Path to installed Zip-Program. Currently 7-Zip are supported. IUsed for predefined TCL-function to Backup project.
		  • (optional) **ENABLE_SDSOC**: Enable SDSOC Setting. Currently only for some reference project as beta version!
		• Xilinx Setting:
		  • **XILDIR**: Set Xilinx installation path (Default: /opt/Xilinx/).
		  • **VIVADO_VERSION**: Current Vivado/LabTool/SDK Version (Example:2018.3). Don't change Vivado Version.
		  • Xilinx Software will be searched in:
		  • VIVADO (optional for project creation and programming): %XILDIR%/Vivado/%VIVADO_VERSION%/ and for SDSoC on %XILDIR%\SDx\%VIVADO_VERSION%\Vivado\
		  • SDK (optional for software projects and programming): %XILDIR%/SDK\%VIVADO_VERSION%/
		  • LabTools (optional for programming only): %XILDIR%/Vivado_Lab/%VIVADO_VERSION%/
		  • SDSOC (optional): %XILDIR%/SDx/%VIVADO_VERSION%/
		• Board Setting:
		  • **PARTNUMBER**: Set Board part number of the project which should be created
		  • Available Numbers: (you can use ID,PRODID,BOARDNAME or SHORTNAME from TExxxx_board_file.csv list)
		  • Used for project creation and programming
		  • 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)
		  • Example TE0726 Module :
		  • USE ID |USE PRODID
		  PARTNUMBER=1 |PARTNUMBER=te0726-01
		• Programming Settings(program*file.cmd):
		  • **SWAPP**: Select Software App, which should be configured.
		  • Use the folder name of the <design_name>/prebuilt/boot_image/<partname>/* subfolder. The *bin,*.mcs or *.bit from this folder will be used.
		  • If you will configure the raw *.bit or *.mcs *.bin from the <design_name>/prebuilt/hardware/<partname>/ folder, use @set SWAPP=NA or @set SWAPP="".
		  • Example: SWAPP=hello_world used the file from prebuilt/boot_image/<partname>/hello_world
		  SWAPP=NA used the file from <design_name>/prebuilt/boot_image/<partname>/
		  • **PROGRAM_ROOT_FOLDER_FILE**: If you want to program design file from the rootfolder <design_name>, set to 1
		  • Attention: it should be only one *.bit, *.msc or *.bin file in the root folder.

design_clear_design_folders.sh	not available	(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! Type "Y" into the command line input to start deleting files
design_run_project_bashmode.sh	not available	(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 Project will created.
Hardware Design		
vivado_create_project_guimode.sh	available	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 Project will created. If old vivado project exists, type "y" into the command line input to start project creation again.
vivado_create_project_bashmode.sh	not available	(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 Project will created. If old vivado project exists, type "y" into the command line input to start project creation again.

vivado_open_existing_project_guimode.sh	available	Opens an existing Project "<design_name>/vivado/<design_name>.xpr" and restore Script-Variables.
Software Design		
sdk_create_prebuilt_project_guimode.sh	not available	(optional) Create SDK project with hardware definition file from prebuild folder. It used the *.hdf from: <design_name>/prebuilt/hardware/<board_file_shortname>/. Set <board_file_shortname> and <app_name> in "design_basic_settings.cmd".
Programming		
program_flash_binfile.sh	not available	(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 installion settings. Default, it used the boot.bin from: <design_name>/prebuilt/boot_images/<board_file_shortname>/<app_name>. Settings are done in "design_basic_settings.cmd".
program_flash_mcsfile.sh	not available	(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 installion settings. Default, it used the <design_name>.mcs from: <design_name>/prebuilt/hardware/<board_file_shortname>. Settings are done in "design_basic_settings.cmd".
program_fpga_bitfile.sh	not available	(optional) Programming FPGA via JTAG with specified <design_name>.bit. Used LabTools Programmer (Vivado or LabTools only), depends on installion settings. Default, it used the <design_name>.bit from: <design_name>/prebuilt/hardware/<board_file_shortname>. Settings are done in "design_basic_settings.cmd".
labtools_open_project_guimode.sh	not available	(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
Hardware Design		
TE::hw_blockdesign_create_bd	[-bd_name] [-msys_local_mem] [-msys_ecc] [-msys_cache] [-msys_debug_module] [-msys_axi_periph] [-msys_axi_intc] [-msys_clk] [-help]	Create new Block-Design with initial Setting for PS, for predefined bd_names: fsysFabric Only, msysMicroblaze, zsys7Series Zynq, zusysUltraScale+ Zynq Typ TE::hw_blockdesign_create_bd -help for more information
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	\[-disable_synth\] \[-disable_bitgen\] \[-disable_hdf\] \[-disable_mcsgen\] \[-disable_reports\] \[-export_prebuilt\] \[-export_prebuilt_only\] \[-help\]	Run Synthese, Implement, and generate Bit-file, optional MCS-file and some report files
Software Design		

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 Defintition file to the working directory: <design_name>/workspace/hsi Run HSI in <design_name>/workspace/hsi for all Programes 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 prepuilt folders <design_name>/prebuilt/... You can deactivate different steps with following args : • -no_hsi : *.elf filesgeneration is disabled • -no_bif : *.bif files generation is disabled • -no_bin : *.bin files generation is disabled • -no_bitmcs: *.bit and *.mcs file (with software design) is disabled
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 Defintition file to the working directory: <design_name>/workspace/sdk Start SDK GUI in this workspace
Programming		
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!
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>
Utilities		

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.
TE:: util_package_length	[-help]	Export Package IO length information to *.csv on the doc folder
Beta Test (Advanced usage only!)		
TE::ADV:: beta_util_sdsoc_project	[-check_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.
TE::ADV:: beta_hw_create_board_part	\[-series <arg>\] \[-all\] \[-preset\] \[-existing_ps\] \[-help\]	create PS or preset.xml PS settings from external tcl scripts
TE::ADV:: beta_hw_export_binary	\[-mode <arg>\] \[-app <arg>\] \[-folder <arg>\] \[-all\] \[-help\]	export prebuilt files to an given folder (based from project folder). Special folder is used, if emtpy

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-webpack.html)
  (optional) Install **Xilinx Vivado LabTools** (Lab Edition)
- Automatically configuration of the reference-designs (only with 2018.3 scripts and newer):
  - Run "_create_win_setup.cmd" or "_create_linux_setup.sh"
    - select "module selection guide" and follow instructions.
      - "**design_basic_settings.cmd**" will be configured over this menu
- Manual Configure the reference-design (Note: batch/bash files works only in the basefolder of the project, use _create_*_setup.cmd/sh or copy manually):
  1. Open "**design_basic_settings.cmd**" with a text-editor:
     a. Set correct Xilinx Environment:
        @set XILDIR=C:/Xilinx
        @set VIVADO_VERSION=2018.3
        Program settings will be search in :
        %XILDIR%/VIVADO/%VIVADO_VERSION%/
        %XILDIR%/Vivado_Lab/%VIVADO_VERSION%/
        %XILDIR%/SDK/%VIVADO_VERSION%/
        Example directory: c:/Xilinx/Vivado/2018.3/
        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_shortname>
        <app_name> (Software Project) used *.bit or *.mcs or *.bin from <design_name>/prebuilt/boot_images/<board_file_shortname>/<app_name>

- Create all prebuilt files in one step:
  2. Run “**design_run_project_batchmode.cmd**”
- (optional to Step 2) Create all prebuilt files in single steps:
  3. Run “**vivado_create_project_guimode.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_guimode.cmd**” or “**labtools_open_project_guimode.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_shortname>/<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_shortname>/<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_shortname>/<app_name>

Basic Design Settings

Initialise TE-scripts on Vivado/LabTools

- Variant 1 (recommended):
  - Start the project with the predefined command file (**vivado_open_existing_project_guimode.cmd)** respectively LabTools with **(labtools_open_project_guimode.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,
        - Press Enter
        - A new Button is shown on the Vivado Gui.

Hardware Design

Board Part Files

More details see TE Board Part Files

Structure Board Parts

Board Parts are located on subfolder "board_files", with the name of the special board. Revisions are splitt in the subfolder of the board part <boardpart_name><version>

Every Version of a Board Parts consists of four files:

- board.xml
- part0_pins.xml
- preset.xml
- picture.jpg or picture.png

Board Part or Design Extension

Board Part Extensions are TCL-Scripts, which can be sourced in Vivado Block Design. Thy are usable with TE-Scripts only. It contains additional settings of PS-settings or special carrier-board design changes.

Use Reference Designs or Vivado TCL-Console(TE-Script extensions, see Initialise TE-scripts on Vivado/LabTools): TE::hw_blockdesign_create_bd -help to create PS with full settings. Or source the TCL file manually direct after "Run Block Automation"

Possible:

- Board Part PS settings are located on subfolder "board_files/preset_extension/" with file name *_preset.tcl.
- Design modiflications are located on subfolder "board_files/bd_mod/" with file name *_bd.tcl.

Board Part CSV Description

Board Part csv file is used for TE-Scripts only.

Name	Description	Value
ID	ID to identify the board variant of the module series, used in TE-Scripts	Number, should be unique in csv list
PRODID	Product ID	Product Name
PARTNAME	FPGA Part Name, used in Vivado and TE-Scripts	Part Name, which is available in Vivado, ex. xc7z045ffg900-2
BOARDNAME	Board Part Name, used in Vivado and TE-Scripts	set Board Part Name or "NA", which is available in Vivado, NA is not defined to run without board part and board part ex. trenz.biz:te0782-02-45:part0:1.0
SHORTNAME	Subdirectory name, used for multi board projects to get correct sources and save prebuilt data	name to save prebuilt files or search for sources
ZYNQFLASHTYP	Flash typ used for programming Zynq-Devices via SDK-Programming Tools (program_flash)	"qspi_single" or "NA", NA is not defined
FPGAFLASHTYP	Flash typ used for programming Devices via Vivado /LabTools	"<Flash Name from Vivado>|<SPI Interface>|<Flash Size in MB>" or "NA" , NA is not defined, ex. s25fl256s-3.3v-qspi-x4-single|SPIx4|32 Flash Name is used for programming, SPI Interface and Size in MB is used for *.mcs build. For Zynq and ZynqMO only Flash name is necessary
PCB_REV	Supported PCB Revision	"<supported PCB Revision>|<supported PCB Revision>", for ex. "REV02" or "REV03|REV02"
DDR_SIZE	Size of Module DDR	use GB or MB, for ex. "2GB" or "512MB" or "NA" if not available
FLASH_SIZE	Size of Module Flash	use MB, for ex. "64MB" or "NA" if not available
EMMC_SIZE	Size of Module EMMC	use GB or MB, for ex. "4GB" or "NA" if not available
OTHERS	Other module relevant changes to distinguish assembly variants	
NOTES	Additional Notes	

Block Design Conventions

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

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

- Create Basic Block Design with PS Board-Part Preset and Carrier-Board extended settings (only if subfolder carrier_extension with tcl files is available), use TE::hw_blockdesign_create_bd -help

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_shortname> 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_shortname> or <design_name>/block_design/PCB Revision> is defined, it will be saved there!
  Only one *.tcl file shoud be in the backup folder respectively the subfolder <board_file_shortname>

Microblaze Firmeware

- Microblaze Firmware (*.elf) can be add to the source folder <design_name>/firmware/<Microblaze IP Instance>.
- For MCS-Core use MCS IP Instance Name. This name must use *mcs* or *syscontrol* in the name.

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 automaticity
- Supported are local application libaries 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_guimode.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

Advanced Usage

Attention not all features of the TE-Scripts are supported in the advanced usage!

User defined board part csv file

To modifiy current board part csv list, make a copy of the original csv and rename with suffix "_mod.csv", ex.TE0782_board_files.csv as TE0782_board_files_mod.csv. Scripts used modified csv instead of the original file.

See Chapter Board Part Files for more information.

User defined Settings

Vivado settings:

Vivado Project settings (corresponding TCL-Commands) can be saved as a user defined file "<design_name>/settings/project_settings.tcl". This file will be loaded automatically on project creation.

Script settings:

Additional script settings (only some predefined variables) can be saved as a user defined file "<design_name>/settings/development_settings.tcl". This file will be loaded automatically on script initialisation.

Design settings:

Additional script settings (only some predefined variables) can be saved as a user defined file "<design_name>/settings/design_settings.tcl". This file will be loaded automatically on script initialisation.

ZIP ignore list:

Files which should not be added in the backup file can be can be defined in this file: "<design_name>/settings/zip_ignore_list.tcl". This file will be loaded automaticaly on script initialisation.

SDSOC settings:

SDSOC settings will are deposited on the following folder: "<design_name>/settings/sdsoc"

User defined TCL Script

TCL Files from "<design_name>/settings/usr" will be load automaticaly on script initialisation.

SDSOC-Template

SDSOC description and files to generate SDSoC project are deposited on the following folder: "<design_name>/settings/sdsoc"

HDL-Design

HDL files can be saved in the subfolder "<design_name>/hdl/" as single files or <design_name>/hdl/folder/ and all subfolders or "<design_name>/hdl/<shortname>" and all subfolders of "<design_name>/hdl/<shortname>". They will be loaded automatically on project creation. Available formats are *.vhd, *.v and *.sv. A own top-file must be specified with the name "<design_name>_top.v" or "<design_name>_top.vhd".

To set file attributes, the file name must include "_simonly_" for simulation only and "_synonly_" for synthese only.

RTL-IP-cores (*.xci). can be saved in the subfolder "<design_name>/hdl/xci" or "<design_name>/hdl/xci/<shortname>". They will be loaded automatically on project creation.

IP -TCL desciption (*_preset.tcl). can be saved in the subfolder "<design_name>/hdl/tcl" or "<design_name>/hdl/tcl/<shortname>". They will be loaded automatically on project creation.

*_preset.tcl must include

- TCL part for IP creation: create_ip -name ...
- TCL part for IP configuration: set_property -dict...
- TCL part for IP target generation: generate_target {instantiation_template} .....

Checklist / Troubleshoot

1. Are you using exactly the same Vivado version? If not then the scripts will not work, no need to try.
2. Ary you using Vivado in Windows PC? Vivado works in Linux also, but the scripts are tested on Windows only.
3. Is you PC OS Installation English? Vivado may work on national versions also, but there have been known problems.
4. Win OS only: Use short path name, OS allows only 256 characters in normal path.
5. Linux OS only: Use bash as shell and add access rights to bash files. Check with "ls ls /bin/sh". It should be desplay: /bin/sh -> bash. Access rights can be changed with "chmod"
6. Are space character on the project path? Somtimes TCL-Scripts can't handle this correctly. Remove spaces from project path.
7. Did you have the newest reference design build version? Maybe it's only a bug from a older version.
8. Check <design_name>/v_log/vivado.log? If no logfile exist, wrong xilinx paths are set in design_basic_settings.cmd
9. On project creation process old files will be deleted. Sometimes the access will be denied by os (synchronisiation problem) and the scripts canceled. Please try again.
10. If nothing helps, send a mail to Trenz Electronic Support (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. Vivado Design Suite User Guide - I/O and Clock Planning (UG899)
4. Vivado Design Suite User Guide - Programming and Debugging (UG908)
5. Zynq-7000 All Programmable SoC Software Developers Guide (UG821)
6. SDSoC Environment User Guide - Getting Started (UG1028)
7. SDSoC Environment - User Guide (UG1027)
8. SDSoC Environment User Guide - Platforms and Libraries (UG1146)

Document Change History

To get content of older revision got to "Change History" of this page and select older revision number.

Date	Revision	Vivado Version	Authors	Description
2019-10-29	v.147	2018.3	John Hartfiel	Work in progress
---	---	2018.2	John Hartfiel	Last Vivado 2018.2 supported project delivery version • no document update was done
10 Jul 2018	v.142	2017.4	John Hartfiel	Last Vivado 2017.4 supported project delivery version
2017-11-03	v.134	2017.2	John Hartfiel	Last Vivado 2017.2 supported project delivery version
2017-09-12	v.131	2017.1	John Hartfiel	Last Vivado 2017.1 supported project delivery version

2017-04-12	v.126	2016.4	John Hartfiel	Last Vivado 2016.4 supported project delivery version
2017-01-16	v.114	2016.2	John Hartfiel	Last Vivado 2016.2 supported project delivery version
2016-06-21	v.83	2015.4	John Hartfiel	Last Vivado 2015.4 supported project delivery version
2013-03-11	v.1	---	Antti Lukats	Initial release
	All		Antti Lukats , John Hartfiel , Susanne Kunath	