TE0723 Test Board

Table of contents

- 1 Table of contents
- 2 Overview
  - 2.1 Key Features
  - 2.2 Revision History
  - 2.3 Release Notes and Know Issues
  - 2.4 Requirements
    - 2.4.1 Software
    - 2.4.2 Hardware
  - 2.5 Content
    - 2.5.1 Design Sources
    - 2.5.2 Additional Sources
    - 2.5.3 Prebuilt
    - 2.5.4 Download
- 3 Design Flow
- 4 Launch
  - 4.1 Programming
    - 4.1.1 QSPI
    - 4.1.2 SD
    - 4.1.3 JTAG
  - 4.2 Usage
    - 4.2.1 Linux
    - 4.2.2 Vivado HW Manager
- 5 System Design - Vivado
  - 5.1 Block Design
    - 5.1.1 PS Interfaces
  - 5.2 Constrains
    - 5.2.1 Basic module constrains
    - 5.2.2 Design specific constrain
- 6 Software Design - SDK/HSI
  - 6.1 Application
    - 6.1.1 zynq_fsbl
    - 6.1.2 zynq_fsbl_flash
    - 6.1.3 hello_te0723
    - 6.1.4 u-boot
- 7 Software Design - PetaLinux
  - 7.1 Config
  - 7.2 U-Boot
  - 7.3 Device Tree
  - 7.4 Kernel
  - 7.5 Rootfs
  - 7.6 Applications
    - 7.6.1 startup
- 8 Additional Software
- 9 Appx. A: Change History and Legal Notices
  - 9.1 Document Change History
  - 9.2 Legal Notices
  - 9.3 Data Privacy
  - 9.4 Document Warranty
  - 9.5 Limitation of Liability
  - 9.6 Copyright Notice
  - 9.7 Technology Licenses
  - 9.8 Environmental Protection
  - 9.9 REACH, RoHS and WEEE

Overview

Zynq PS Design with Linux Example.
Refer to http://trenz.org/te0723-info for the current online version of this manual and other available documentation.

Key Features

- Vivado 2018.3
- PetaLinux
- SD
- USB
- I2C
- Special FSBL for QSPI programming

Revision History

Date	Vivado	Project Built	Authors	Description
2019-03-14	2018.3	TE0723-test_board_noprebuilt-vivado_2018.3-build_02_20190314070505.zip TE0723-test_board-vivado_2018.3-build_02_20190314070455.zip	John Hartfiel	• TE Script update • rework of the FSBLs
2018-09-11	2017.4	te0723-test_board-vivado_2017.4-build_11_20180911144828.zip te0723-test_board_noprebuilt-vivado_2017.4-build_11_20180911144844.zip	John Hartfiel	• correction netboot address for te0723-03(r) (REV02 /REV03)
2018-02-20	2017.4	te0723-test_board-vivado_2017.4-build_06_20180220121024.zip te0723-test_board_noprebuilt-vivado_2017.4-build_06_20180220121039.zip	John Hartfiel	• initial release 2017.4

Design Revision History

Release Notes and Know Issues

Issues	Description	Workaround	To be fixed version
No known issues	---	---	---

Known Issues

Requirements

Software

Software	Version	Note
Vivado	2018.3	needed
SDK	2018.3	needed
PetaLinux	2018.3	needed

Software

Hardware

Basic description of TE Board Part Files is available on TE Board Part Files.

Complete List is available on <design name>/board_files/*_board_files.csv

Design supports following modules:

Module Model	Board Part Short Name	PCB Revision Support	DDR	QSPI Flash	EMMC	Others	Notes
TE0723-01	01_128mb	REV01	128MB LPDDR2	16MB	NA	NA	LPDDR2
TE0723-03	r_128mb	REV03|REV02	128MB DDR3L	16MB	NA	NA	LPDDR3
TE0723-03M	m_512mb	REV03|REV02	512MB DDR3L	16MB	NA	NA	LPDDR3
TE0723-03-07S-1C	7s_512mb	REV03|REV02	512MB DDR3L	16MB	NA	NA	LPDDR3

Hardware Modules

Design supports following carriers:

Carrier Model	Notes

Hardware Carrier

Additional HW Requirements:

Additional Hardware	Notes
USB Cable	Connect to USB2 or better USB3 Hub for proper power supply over USB

Additional Hardware

Content

For general structure and of the reference design, see Project Delivery - AMD devices

Design Sources

Type	Location	Notes
Vivado	<design name>/block_design <design name>/constraints <design name>/ip_lib	Vivado Project will be generated by TE Scripts
SDK/HSI	<design name>/sw_lib	Additional Software Template for SDK/HSI and apps_list.csv with settings for HSI
PetaLinux	<design name>/os/petalinux	PetaLinux template with current configuration

Design sources

Additional Sources

Type	Location	Notes
init.sh	<design name>/sd/	Additional Initialization Script for Linux

Additional design sources

Prebuilt

File	File-Extension	Description
BIF-File	*.bif	File with description to generate Bin-File
BIN-File	*.bin	Flash Configuration File with Boot-Image (Zynq-FPGAs)
BIT-File	*.bit	FPGA (PL Part) Configuration File
DebugProbes-File	*.ltx	Definition File for Vivado/Vivado Labtools Debugging Interface
Diverse Reports	---	Report files in different formats
Hardware-Platform-Specification-Files	*.hdf	Exported Vivado Hardware Specification for SDK/HSI and PetaLinux
LabTools Project-File	*.lpr	Vivado Labtools Project File
OS-Image	*.ub	Image with Linux Kernel (On Petalinux optional with Devicetree and RAM-Disk)
Software-Application-File	*.elf	Software Application for Zynq or MicroBlaze Processor Systems

Prebuilt files (only on ZIP with prebult content)

Download

Reference Design is only usable with the specified Vivado/SDK/PetaLinux/SDx version. Do never use different Versions of Xilinx Software for the same Project.

Reference Design is available on:

- TE0723 "Test Board" Reference Design

Design Flow

⚠ Reference Design is available with and without prebuilt files. It's recommended to use TE prebuilt files for first lunch.

Trenz Electronic provides a tcl based built environment based on Xilinx Design Flow.

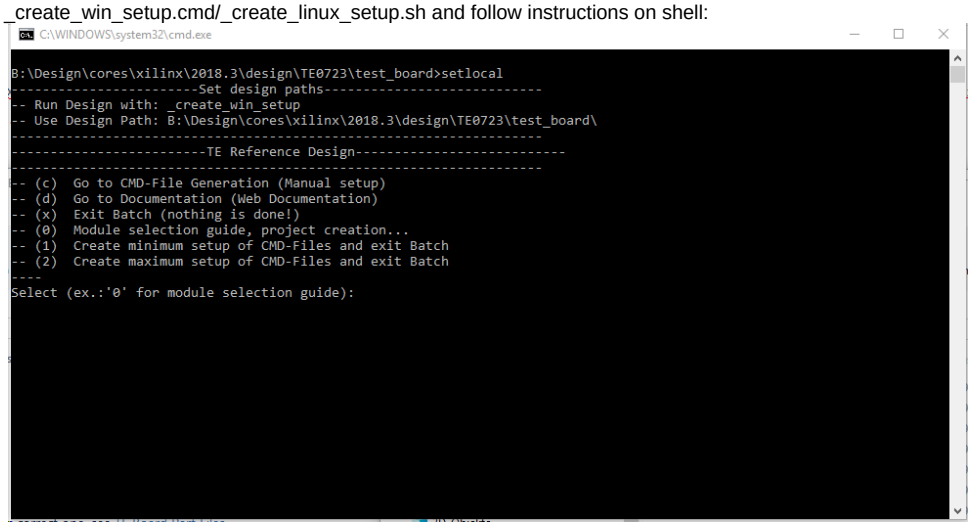
See also:

- AMD Development Tools#XilinxSoftware-BasicUserGuides
- Vivado Projects - TE Reference Design
- Project Delivery.

The Trenz Electronic FPGA Reference Designs are TCL-script based project. Command files for execution will be generated with "_create_win_setup.cmd" on Windows OS and "_create_linux_setup.sh" on Linux OS.

TE Scripts are only needed to generate the vivado project, all other additional steps are optional and can also executed by Xilinx Vivado/SDK GUI. For currently Scripts limitations on Win and Linux OS see: Project Delivery Currently limitations of functionality

1. _create_win_setup.cmd/_create_linux_setup.sh and follow instructions on shell:



2. Press 0 and enter to start "Module Selection Guide"
3. (optional Win OS) Generate Virtual Drive or use short directory for the reference design (for example x:\<design name>)
4. Create Project (follow instruction of the product selection guide), settings file will be configured automatically during this process
    a. (optional for manual changes) Select correct device and Xilinx install path on "design_basic_settings.cmd" and create Vivado project with "vivado_create_project_guimode.cmd"
    Note: Select correct one, see TE Board Part Files
5. Create HDF and export to prebuilt folder
    a. Run on Vivado TCL: TE::hw_build_design -export_prebuilt
    Note: Script generate design and export files into \prebuilt\hardware\<short dir>. Use GUI is the same, except file export to prebuilt folder
6. Create Linux (uboot.elf and image.ub) with exported HDF
    a. HDF is exported to "prebuilt\hardware\<short name>"
    Note: HW Export from Vivado GUI create another path as default workspace.
    Create Linux images on VM, see PetaLinux KICKstart
        i. Use TE Template from /os/petalinux
        Note: For 128MB only: Netboot Offset must be reduced manually, see Config
7. Add Linux files (uboot.elf and image.ub) to prebuilt folder
    a. "prebuilt\os\petalinux\<DDR size>" or "prebuilt\os\petalinux\<short name>"
8. Generate Programming Files with HSI/SDK
    a. Run on Vivado TCL: TE::sw_run_hsi
    Note: Scripts generate applications and bootable files, which are defined in "sw_lib\apps_list.csv"
    b. (alternative) Start SDK with Vivado GUI or start with TE Scripts on Vivado TCL: TE::sw_run_sdk
    Note: See SDK Projects

Launch

Programming

⚠ Check Module and Carrier TRMs for proper HW configuration before you try any design.

Xilinx documentation for programming and debugging: Vivado/SDK/SDSoC-Xilinx Software Programming and Debugging

QSPI

1. Connect JTAG and power on carrier with module
2. Open Vivado Project with "vivado_open_existing_project_guimode.cmd" or if not created, create with "vivado_create_project_guimode.cmd"
3. Type on Vivado TCL Console: TE::pr_program_flash_binfile -swapp u-boot
   Note: To program with SDK/Vivado GUI, use special FSBL (zynqmp_fsbl_flash) on setup
   optional "TE::pr_program_flash_binfile -swapp hello_tec0850" possible
4. Copy image.ub on SD-Card
   - For correct prebuilt file location, see <design_name>/prebuilt/readme_file_location.txt
5. Insert SD-Card

SD

Xilinx Zynq devices in CLG225 package do not support SD Card boot directly from ROM bootloader. Use QSPI for primary boot and SD for secondary boot (uboot).

Modification for FSBL on QSPI and bitstream and applicaton on SD, see Xilinx AR# 66846.

JTAG

Not used on this Example.u

Usage

1. Prepare HW like described on section Programming
2. Connect UART USB (most cases same as JTAG)
3. Insert SD Card with image.ub
4. Power On PCB
   Note: 1. Zynq Boot ROM loads FSBL from QSPI into OCM, 2. FSBL loads U-boot from QSPI into DDR, 3. U-boot load Linux from SD into DDR
   UART0 over PL is used on this reference design, access is available after PL design is loaded from FSBL.

Linux

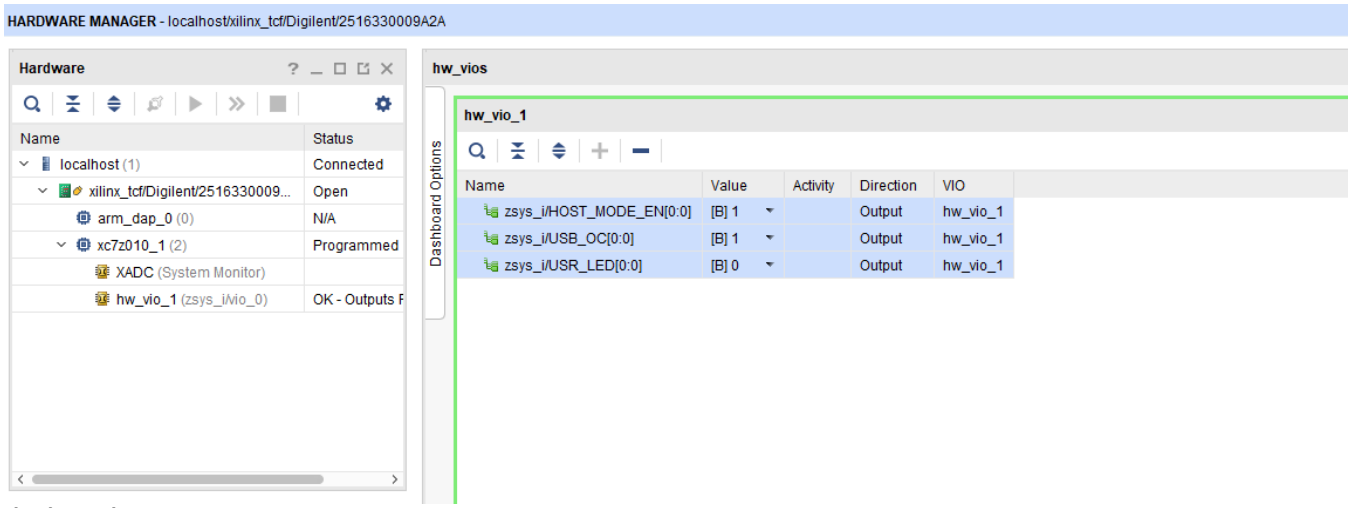
1. Open Serial Console (e.g. putty)
   a. Speed: 115200
   b. COM Port: Win OS, see device manager, Linux OS see dmesg |grep tty (UART is *USB1)
2. Linux Console:
   Note: Wait until Linux boot finished For Linux Login use:
   a. User Name: root
   b. Password: root
3. You can use Linux shell now.
   a. I2C 1 Bus type: i2cdetect -y -r 0
   b. USB: insert USB device
4. Option Features
   a. init.sh scriptsadd init.sh script on SD, content will be load automatically on startup (template included in ./misc/SD)

Vivado HW Manager

Open Vivado HW-Manager and add VIO signal to dashboard (*.ltx located on prebuilt folder).

Control:

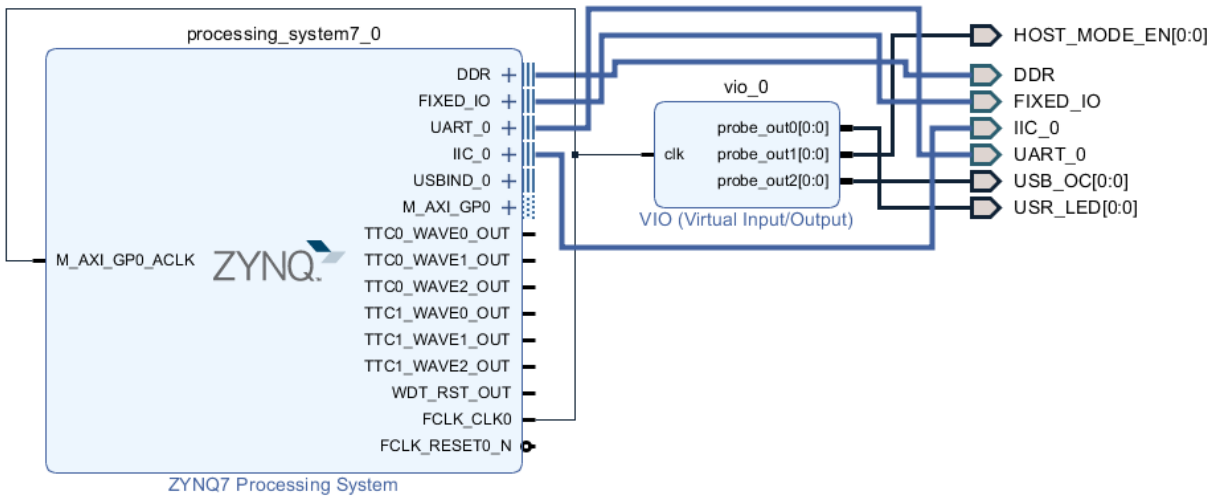
- LED: Green LED D6 on TE0723
- USB: Host Mode and OC (see schematics)



Vivado Hardware Manager

System Design - Vivado

Block Design



Block Design

PS Interfaces

Type	Note
DDR	---
QSPI	MIO
USB0	MIO
SD1	MIO
UART0	EMIO
UART1	MIO
I2C0	EMIO
GPIO	MIO
USB RST	MIO
TTC0..1	
WDT	

PS Interfaces

Constrains

Basic module constrains

_i_bitgen_common.xdc

```
#
# Common bitgen related settings
#

set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
#set_property BITSTREAM.CONFIG.CONFIGRATE 66 [current_design]
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

set_property BITSTREAM.CONFIG.USR_ACCESS TIMESTAMP [current_design]
```

_i_unused_io.xdc

```
#
# Set unused pin pullup: PULLNONE, PULLUP, PULLDOWN
#

set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design]

#set_property BITSTREAM.CONFIG.UNUSEDPIN PULLUP [current_design]
#set_property BITSTREAM.CONFIG.UNUSEDPIN PULLDONE [current_design]
```

Design specific constrain

_i_io.xdc

```
# ########################
# UART0 to FTDI
set_property PACKAGE_PIN H14 [get_ports UART_0_txd]
set_property PACKAGE_PIN H13 [get_ports UART_0_rxd]

set_property PACKAGE_PIN J15 [get_ports UART_0_ctsn]
set_property PACKAGE_PIN J14 [get_ports UART_0_rtsn]

set_property PACKAGE_PIN K15 [get_ports UART_0_dsrn]
set_property PACKAGE_PIN L15 [get_ports UART_0_dtrn]

#NC: UART_0_dcdn, UART_0_ri
set_property PACKAGE_PIN L14 [get_ports UART_0_dcdn]
set_property PACKAGE_PIN M15 [get_ports UART_0_ri]

set_property IOSTANDARD LVCMOS33 [get_ports UART_0_*]

# ########################
#I2C to J1 connector
set_property PACKAGE_PIN P13 [get_ports IIC_0_scl_io]
set_property PACKAGE_PIN R13 [get_ports IIC_0_sda_io]
set_property IOSTANDARD LVCMOS33 [get_ports IIC_0_scl_io]
set_property IOSTANDARD LVCMOS33 [get_ports IIC_0_sda_io]
# ########################
#LED to D6 green
set_property PACKAGE_PIN G14 [get_ports {USR_LED[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {USR_LED[0]}]
# ########################
#USB
set_property PACKAGE_PIN F15 [get_ports {USB_OC[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {USB_OC[0]}]

set_property PACKAGE_PIN L13 [get_ports {HOST_MODE_EN[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {HOST_MODE_EN[0]}]
```

Software Design - SDK/HSI

For SDK project creation, follow instructions from:

SDK Projects

Application

zynq_fsbl

TE modified 2018.3 FSBL

General:

- Modified Files:main.c, fsbl_hooks.h/.c (search for 'TE Mod' on source code)
- Add Files: te_fsbl_hooks.h/.c(for hooks and board)\n\
- General Changes:
    - Display FSBL Banner and Device ID

Module Specific:

- Add Files: all TE Files start with te_*
    - no design specific modification

zynq_fsbl_flash

TE modified 2018.3 FSBL

General:

- Modified Files: main.c
- General Changes:
    - Display FSBL Banner
    - Set FSBL Boot Mode to JTAG
    - Disable Memory initialisation

hello_te0723

Hello TE0723 is a Xilinx Hello World example as endless loop instead of one console output.

u-boot

U-Boot.elf is generated with PetaLinux. SDK/HSI is used to generate Boot.bin.

Software Design - PetaLinux

For PetaLinux installation and project creation, follow instructions from:

- PetaLinux KICKstart

Config

Start with **petalinux-config** or **petalinux-config --get-hw-description**

Changes:

- CONFIG_SUBSYSTEM_SERIAL_PS7_UART_0_SELECT=y
- CONFIG_SUBSYSTEM_SERIAL_IP_NAME="ps7_uart_0"
- CONFIG_SUBSYSTEM_NETBOOT_OFFSET = 0x4000000 !Must be done manually for 128MB DDR only not done on with HDF import from the template!

U-Boot

Start with **petalinux-config -c u-boot**

Changes:

- CONFIG_ENV_IS_NOWHERE=y
- # CONFIG_ENV_IS_IN_SPI_FLASH is not set

Change platform-top.h:

Device Tree

```
/include/ "system-conf.dtsi"
/ {
};

/* USB */
/{
    usb_phy0: usb_phy@0 {
        compatible = "ulpi-phy";
        //compatible = "usb-nop-xceiv";
        #phy-cells = <0>;
        reg = <0xe0002000 0x1000>;
        view-port = <0x0170>;
        drv-vbus;
    };
};

&usb0 {
    dr_mode = "host";
    //dr_mode = "peripheral";
    usb-phy = <&usb_phy0>;
};
```

Kernel

Start with **petalinux-config -c kernel**

Changes:

- No changes.

Rootfs

Start with **petalinux-config -c rootfs**

Changes:

- i2c-tools

Applications

startup

Script App to load init.sh from SD Card if available.

See: \os\petalinux\project-spec\meta-user\recipes-apps\startup\files

Additional Software

No additional software is needed.

Appx. A: Change History and Legal Notices

Document Change History

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

Date	Document Revision	Authors	Description

Error rendering macro 'page-info' Ambiguous method overloading for method jdk.proxy244.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes between: [interface com.atlassian.confluence.user.ConfluenceUser, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject] [interface com.atlassian.user.User, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject]	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy244.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes between: [interface com.atlassian.confluence.user.ConfluenceUser, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject] [interface com.atlassian.user.User, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject] Unknown macro: 'metadata'	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy244.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes between: [interface com.atlassian.confluence.user.ConfluenceUser, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject] [interface com.atlassian.user.User, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject]	• 2018.3 release • Design + Script Update
2018-09-11	v.6	John Hartfiel	• TE0723-03(r) design update
2018-02-20	v.4	John Hartfiel	• 2017.4 release

---	all	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy244.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes between: [interface com.atlassian.confluence.user.ConfluenceUser, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject] [interface com.atlassian.user.User, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject]	--

Legal Notices

Data Privacy

Please also note our data protection declaration at https://www.trenz-electronic.de/en/Data-protection-Privacy

Document Warranty

The material contained in this document is provided “as is” and is subject to being changed at any time without notice. Trenz Electronic does not warrant the accuracy and completeness of the materials in this document. Further, to the maximum extent permitted by applicable law, Trenz Electronic disclaims all warranties, either express or implied, with regard to this document and any information contained herein, including but not limited to the implied warranties of merchantability, fitness for a particular purpose or non infringement of intellectual property. Trenz Electronic shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein.

Limitation of Liability

In no event will Trenz Electronic, its suppliers, or other third parties mentioned in this document be liable for any damages whatsoever (including, without limitation, those resulting from lost profits, lost data or business interruption) arising out of the use, inability to use, or the results of use of this document, any documents linked to this document, or the materials or information contained at any or all such documents. If your use of the materials or information from this document results in the need for servicing, repair or correction of equipment or data, you assume all costs thereof.

Copyright Notice

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Trenz Electronic.

Technology Licenses

The hardware / firmware / software described in this document are furnished under a license and may be used /modified / copied only in accordance with the terms of such license.

Environmental Protection

To confront directly with the responsibility toward the environment, the global community and eventually also oneself. Such a resolution should be integral part not only of everybody's life. Also enterprises shall be conscious of their social responsibility and contribute to the preservation of our common living space. That is why Trenz Electronic invests in the protection of our Environment.

REACH, RoHS and WEEE

REACH

Trenz Electronic is a manufacturer and a distributor of electronic products. It is therefore a so called downstream user in the sense of REACH. The products we supply to you are solely non-chemical products (goods). Moreover and under normal and reasonably foreseeable circumstances of application, the goods supplied to you shall not release any substance. For that, Trenz Electronic is obliged to neither register nor to provide safety data sheet. According to present knowledge and to best of our knowledge, no SVHC (Substances of Very High Concern) on the Candidate List are contained in our products. Furthermore, we will immediately and unsolicited inform our customers in compliance with REACH - Article 33 if any substance present in our goods (above a concentration of 0,1 % weight by weight) will be classified as SVHC by the European Chemicals Agency (ECHA).

RoHS

Trenz Electronic GmbH herewith declares that all its products are developed, manufactured and distributed RoHS compliant.

WEEE

Information for users within the European Union in accordance with Directive 2002/96/EC of the European Parliament and of the Council of 27 January 2003 on waste electrical and electronic equipment (WEEE).

Users of electrical and electronic equipment in private households are required not to dispose of waste electrical and electronic equipment as unsorted municipal waste and to collect such waste electrical and electronic equipment separately. By the 13 August 2005, Member States shall have ensured that systems are set up allowing final holders and distributors to return waste electrical and electronic equipment at least free of charge. Member States shall ensure the availability and accessibility of the necessary collection facilities. Separate collection is the precondition to ensure specific treatment and recycling of waste electrical and electronic equipment and is necessary to achieve the chosen level of protection of human health and the environment in the European Union. Consumers have to actively contribute to the success of such collection and the return of waste electrical and electronic equipment. Presence of hazardous substances in electrical and electronic equipment results in potential effects on the environment and human health. The symbol consisting of the crossed-out wheeled bin indicates separate collection for waste electrical and electronic equipment.

Trenz Electronic is registered under WEEE-Reg.-Nr. DE97922676.

Error rendering macro 'page-info'

Ambiguous method overloading for method jdk.proxy244.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes between: [interface com.atlassian.confluence.user.ConfluenceUser, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject] [interface com.atlassian.user.User, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject]