TEC0330 Test Board

Table of contents

- 1 Overview
  - 1.1 Key Features
  - 1.2 Revision History
  - 1.3 Release Notes and Know Issues
  - 1.4 Requirements
    - 1.4.1 Software
    - 1.4.2 Hardware
  - 1.5 Content
    - 1.5.1 Design Sources
    - 1.5.2 Additional Sources
    - 1.5.3 Prebuilt
    - 1.5.4 Download
- 2 Design Flow
- 3 Vitis/Vivado 2021.2
  - MicroBlaze
  - SPI ELF Bootloader
  - I2C
  - Flash
  - MIG
  - FMeter
  - SI5338 initialisation with I2C
  - PCIe
  - DDR3 ECC SODIMM (currently ECC disabled)
  - 3.1 Programming
    - 3.1.1 Get prebuilt boot binaries
    - 3.1.2 QSPI-Boot mode
    - 3.1.3 SD-Boot mode
    - 3.1.4 JTAG
  - 3.2 Usage
    - 3.2.1 JTAG UART Console:
    - 3.2.2 Vivado HW Manager:
    - 3.2.3 I2C
- 4 System Design - Vivado
  - 4.1 Block Design
  - 4.2 Constraints
    - 4.2.1 Basic module constrains
    - 4.2.2 Design specific constrain
- 5 Software Design - Vitis
  - 5.1 Application
    - 5.1.1 hello_tec0330
    - 5.1.2 scu
    - 5.1.3 spi_bootloader
- 6 Additional Software
  - 6.1 SI5338
- 7 Appx. A: Change History and Legal Notices
  - 7.1 Document Change History
  - 7.2 Legal Notices
  - 7.3 Data Privacy
  - 7.4 Document Warranty
  - 7.5 Limitation of Liability
  - 7.6 Copyright Notice
  - 7.7 Technology Licenses
  - 7.8 Environmental Protection
  - 7.9 REACH, RoHS and WEEE
- 8 Table of contents

Overview

This TEC0330 reference design implements the SI5338 Configuration, DDR Configuration and PCIe Core Example Design.

Refer to http://trenz.org/tec0330-info for the current online version of this manual and other available documentation.

Key Features

Revision History

Date	Vivado	Project Built	Authors	Description
2022-09-22	2021.2	TEC0330-test_board_noprebuilt-vivado_2021.2-build_17_20220922155804.zip TEC0330-test_board-vivado_2021.2-build_17_20220922155804.zip	Waldemar Hanemann	• version 2021.2 update
2018-10-30	2018.2	TEC0330-test_board_noprebuilt-vivado_2018.2-build_03_20181030122205.zip TEC0330-test_board-vivado_2018.2-build_03_20181030122147.zip	John Hartfiel	• initial release

Design Revision History

Release Notes and Know Issues

Issues	Description	Workaround	To be fixed version

DDR3 ECC SODIMM	DDR3 does not work with ECC enabled	Disable ECC: • for Block Design MIG with AXI Interface, create 64Bit MIG • for RTL MIG with Native Interface, disable ECC on MIG configuration and use 72Bit for Data	---

Known Issues

Requirements

Software

Software	Version	Note
Vitis	2021.2	needed, Vivado is included into Vitis installation
Clockbuilder Pro	4.5(used in this design)	optional

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	Others	Notes
TEC0330-04-(330-2C)	330_2	REV04	DDR3 ECC SODIMM	32MB		• DDR configured for AW24P7228BLK0M (8GB)
TEC0330-05	330_2	REV05	DDR3 ECC SODIMM	32MB		• DDR configured for AW24P7228BLK0M (8GB)

TEC0330-05-S	330_2	REV05	DDR3 ECC SODIMM	32MB		• DDR configured for AW24P7228BLK0M (8GB)

Hardware Modules

Design supports following carriers:

Carrier Model	Notes
PC with PCIe Card slot	Card need 3.3V from PCIe and 12V from ATX connector

Hardware Carrier

Additional HW Requirements:

Additional Hardware	Notes
JTAG Programmer	• TE0790 with TE0791 for CPLD or FPGA • Xilinx compatible JTAG programmer for FPGA
DDR3 (204 Pin with ECC)	• in this design used: ○ AW24P7228BLK0M (max. 8GB)

Additional Hardware

Content

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

Design Sources

Type	Location	Notes
Vivado	<project folder>\block_design <project folder>\constraints <project folder>\ip_lib <project folder>\board_files	Vivado Project will be generated by TE Scripts
Vitis	<project folder>\sw_lib	Additional Software Template for Vitis and apps_list.csv with settings automatically for Vitis app generation

Design sources

Additional Sources

Type	Location	Notes

SI5338	<project folder>/misc/Si5338	SI5338 Project with current PLL Configuration

Additional design sources

Prebuilt

File	File-Extension	Description
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	*.xsa	Exported Vivado hardware description file for Vitis and PetaLinux
LabTools Project-File	*.lpr	Vivado Labtools Project File
MCS-File	*.mcs	Flash Configuration File with Boot-Image (MicroBlaze or FPGA part only)
MMI-File	*.mmi	File with BRAM-Location to generate MCS or BIT-File with *.elf content (MicroBlaze only)
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/Vitis/PetaLinux version. Do never use different Versions of Xilinx Software for the same Project.

Reference Design is available on:

- TEC0330 "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

- 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/Vitis GUI. For currently Scripts limitations on Win and Linux OS see: Project Delivery Currently limitations of functionality

Caution! Win OS has a 260 character limit for path lengths which can affect the Vivado tools. To avoid this issue, use Virtual Drive or the shortest possible names and directory locations for the reference design (for example "x:\<project folder>")

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

_create_win_setup.cmd/_create_linux_setup.sh

```
------------------------Set design paths---------------------------
-- Run Design with: _create_win_setup
-- Use Design Path: <absolute project path>
-------------------------------------------------------------------
------------------------TE Reference
Design--------------------------
-------------------------------------------------------------------
-- (0)  Module selection guide, project creation...prebuilt export...
-- (1)  Create minimum setup of CMD-Files and exit Batch
-- (2)  Create maximum setup of CMD-Files and exit Batch
-- (3)  (internal only) Dev
-- (4)  (internal only) Prod
-- (c)  Go to CMD-File Generation (Manual setup)
-- (d)  Go to Documentation (Web Documentation)
-- (g)  Install Board Files from Xilinx Board Store (beta)
-- (a)  Start design with unsupported Vivado Version (beta)
-- (x)  Exit Batch (nothing is done!)
----
Select (ex.:'0' for module selection guide):
```

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 and follow instructions of the product selection guide, settings file will be configured automatically during this process.
    - 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 also Vivado Board Part Flow

5. Create hardware description file (.xsa file) for PetaLinux project and export to prebuilt folder

run on Vivado TCL (Script generates design and export files into "<project folder>\prebuilt\hardware\<short name>")

```
TE::hw_build_design -export_prebuilt
```

Using Vivado GUI is the same, except file export to prebuilt folder.

6. Generate Programming Files with Vitis

run on Vivado TCL (Script generates applications and bootable files, which are defined in "test_board\sw_lib\apps_list.csv")

```
TE::sw_run_vitis -all
TE::sw_run_vitis (optional; Start Vitis from Vivado GUI or start with TE Scripts on Vivado TCL)
```

TCL scripts generate also platform project, this must be done manually in case GUI is used. See Vitis

7. (Optional) BlockRam Firmware Update
    a. Copy "<project folder>\prebuilt\software\<short name>\spi_bootloader.elf" into "<project folder>\firmware\microblaze_0\"
    b. Copy "<project folder>\workspace\sdk\scu\Release\scu.elf" into "\firmware\microblaze_mcs_0\"
    c. Regenerate Vivado Project or Update Bitfile only with "spi_bootloader.elf" and "scu_te0712.elf"

```
TE::hw_build_design -export_prebuilt
TE::sw_run_vitis -all
```

Launch

Programming

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

Reference Design is also available with prebuilt files. It's recommended to use TE prebuilt files for first launch.

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

Get prebuilt boot binaries

1. Run _create_win_setup.cmd/_create_linux_setup.sh and follow instructions on shell
2. Press 0 and enter to start "Module Selection Guide"
    a. Select assembly version
    b. Validate selection
    c. Select create and open delivery binary folder

Note: Folder "<project folder>_binaries_<Article Name>" with subfolder "boot_<app name>" for different applications will be generate

QSPI-Boot mode

1. Connect JTAG and Power ON PC
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:

run on Vivado TCL (Script programs u-boot.mcs onto QSPI flash)
`TE::pr_program_flash -swapp hello_tec0330`

4. Reboot PC

SD-Boot mode

Not used on this Example.

JTAG

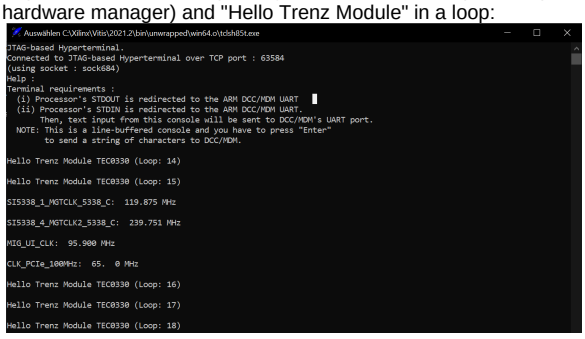
- Connect Vivado HW Manager and program FPGA

Usage

1. Prepare HW like described on section Programming
2. Power On PCB
   Note: 1. FPGA Load Bitfile into FPGA,MCS configure SI5338 and starts microblaze design, modified SPI Bootloader to load hello_tec0330 application from QSPI into DDR (Depends on linker script)

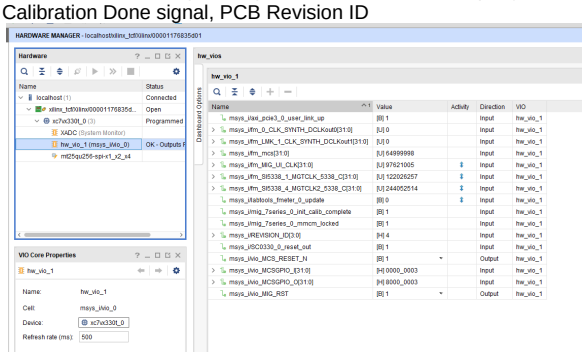
JTAG/UART Console:

- Launch XSCT or the XSDB console on Vitis:
  - type: connect
  - type: targets -set -filter {name =~ "MicroBlaze Debug*"} -index 0
  - type: jtagterminal -start
  - Separate console starts printing out four internal frequencies(can also be seen in the hardware manager) and "Hello Trenz Module" in a loop:



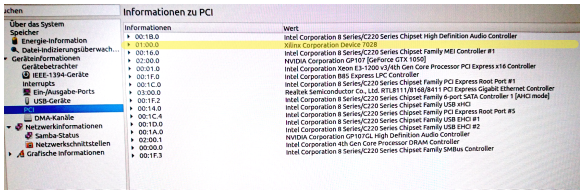
Vivado HW Manager:

1. Open Vivado HW Manager
2. Add VIO to Dashboard
3. Set Radix to unsigned integer for FMeterCLKs (fm_*). Note measurement is not accurate
4. Control:
    a. MCS Reset
    b. MIG Reset
5. Read: SI5338 CLKs (Unit Hz), PCIe Core User Link Up signal, MIG MMCM Lock signal, MIG Init Calibration Done signal, PCB Revision ID



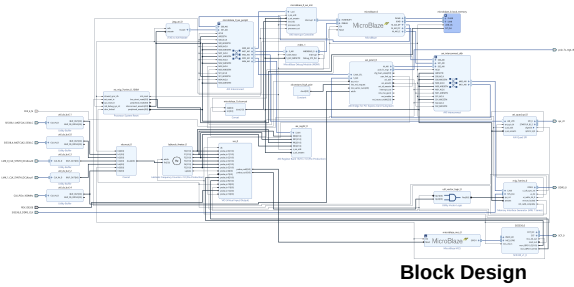
PC:

- Use for example PCI-Z (Win) or KInfoCenter (Linux) to detect PCIe Card



System Design - Vivado

Block Design



Block Design

Constraints

Basic module constrains

_i_bitgen_common.xdc

```
#
# Default common settings that do not depend assembly variant
#
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 66 [current_design]

set_property BITSTREAM.CONFIG.SPI_32BIT_ADDR YES [current_design]
set_property BITSTREAM.CONFIG.SPI_BUSWIDTH 4 [current_design]
set_property BITSTREAM.CONFIG.M1PIN PULLNONE [current_design]
set_property BITSTREAM.CONFIG.M2PIN PULLNONE [current_design]
set_property BITSTREAM.CONFIG.M0PIN PULLNONE [current_design]

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

_i_common.xdc

```
#
#
#
set_property BITSTREAM.CONFIG.UNUSEDPIN PULLDOWN [current_design]
```

Design specific constrain

_i_io.xdc

```
#----------
#IIC to CPLD
set_property PACKAGE_PIN W29 [get_ports SCF_0_cpld_25_scl]
set_property PACKAGE_PIN W26 [get_ports SCF_0_cpld_19_oe]
set_property PACKAGE_PIN V29 [get_ports SCF_0_cpld_24_sda]
set_property IOSTANDARD LVCMOS18 [get_ports SCF_0_cpld_25_scl]
set_property IOSTANDARD LVCMOS18 [get_ports SCF_0_cpld_19_oe]
set_property IOSTANDARD LVCMOS18 [get_ports SCF_0_cpld_24_sda]
#----------
#PCIe
set_property PACKAGE_PIN E33 [get_ports FEX_4_N]
set_property IOSTANDARD LVCMOS18 [get_ports FEX_4_N]
set_property PACKAGE_PIN AD6 [get_ports {CLK_PCIe_100MHz_clk_p[0]}]
#todo check auto placement:
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets msys_i/axi_pcie3_0/inst
/pcie3_ip_i/inst/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i
/CLK_TXOUTCLK]
#----------
#Revision ID
set_property PACKAGE_PIN AP27 [get_ports {REV_ID[0]}]
set_property PACKAGE_PIN AN27 [get_ports {REV_ID[1]}]
set_property PACKAGE_PIN AP26 [get_ports {REV_ID[2]}]
set_property PACKAGE_PIN AP25 [get_ports {REV_ID[3]}]
set_property IOSTANDARD LVCMOS18 [get_ports {REV_ID[*]}]
#----------
```

```
#QSPI
set_property PACKAGE_PIN AL33 [get_ports {spi_rtl_ss_io[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {spi_rtl_ss_io[0]}]
set_property PACKAGE_PIN AN33 [get_ports spi_rtl_io0_io]
set_property PACKAGE_PIN AN34 [get_ports spi_rtl_io1_io]
set_property PACKAGE_PIN AK34 [get_ports spi_rtl_io2_io]
set_property PACKAGE_PIN AL34 [get_ports spi_rtl_io3_io]
set_property IOSTANDARD LVCMOS18 [get_ports spi_rtl_io0_io]
set_property IOSTANDARD LVCMOS18 [get_ports spi_rtl_io1_io]
set_property IOSTANDARD LVCMOS18 [get_ports spi_rtl_io2_io]
set_property IOSTANDARD LVCMOS18 [get_ports spi_rtl_io3_io]
#----------
#CLKS
##SI5338_0_DDR3_CLK #diff 1.5V AG17/AH17
set_property PACKAGE_PIN AG17 [get_ports {SI5338_0_DDR3_CLK_clk_p}]
set_property IOSTANDARD DIFF_SSTL15 [get_ports {SI5338_0_DDR3_CLK_clk_p}]
##SI5338_1_MGTCLK_5338_C #diff MGT 1.8V AB6/AB5
set_property PACKAGE_PIN AB6 [get_ports {SI5338_1_MGTCLK_5338_C_clk_p[0]}]
###SI5338_3_LMK_CLK #diff MGT 1.8V to LMK CLKin1
##SI5338_4_MGTCLK2_5338_C #diff MGT 1.8V H6/H5
set_property PACKAGE_PIN H6 [get_ports {SI5338_4_MGTCLK2_5338_C_clk_p[0]}]
##LMK_0_CLK_SYNTH_DCLKout0 #diff  1.8V AD29/AE29
set_property PACKAGE_PIN AD29 [get_ports {LMK_0_CLK_SYNTH_DCLKout0_clk_p
[0]}]
set_property IOSTANDARD LVDS [get_ports {LMK_0_CLK_SYNTH_DCLKout0_clk_p
[0]}]
set_property DIFF_TERM TRUE [get_ports {LMK_0_CLK_SYNTH_DCLKout0_clk_p[0]}]
##LMK_1_CLK_SYNTH_DCLKout1 #diff  1.8V AE31/AF31
set_property PACKAGE_PIN AE31 [get_ports {LMK_1_CLK_SYNTH_DCLKout1_clk_p
[0]}]
set_property IOSTANDARD LVDS [get_ports {LMK_1_CLK_SYNTH_DCLKout1_clk_p
[0]}]
set_property DIFF_TERM TRUE [get_ports {LMK_1_CLK_SYNTH_DCLKout1_clk_p[0]}]
###LMK_2_CLKIN_5338_P #diff  1.8Vto Si5338 IN1/IN2
###LMK_3_CLK_SYNTH_SDCLKout3 #diff  1.8Vto N.C.
###LMK_4_CLK_SYNTH_SDCLKout4 #diff MGT 1.8V T6/T5
###LMK_5_CLK_SYNTH_SDCLKout5 #diff  1.8Vto N.C.
###LMK_6_CLK_SYNTH_SDCLKout6 #diff  1.8Vto N.C.
###LMK_7_CLK_SYNTH_SDCLKout7 #diff MGT 1.8V F6/F5
###LMK_8_CLK_SYNTH_SDCLKout8 #diff  1.8Vto N.C.
###LMK_9_CLK_SYNTH_SDCLKout9 #diff  1.8Vto N.C.
###LMK_10_CLK_SYNTH_SDCLKout10 #diff  1.8Vto N.C.
###LMK_11_CLK_SYNTH_SDCLKout11 #diff  1.8Vto N.C.
###LMK_12_CLK_SYNTH_SDCLKout12 #diff  1.8Vto N.C.
###LMK_13_CLK_SYNTH_SDCLKout13 #diff  1.8Vto N.C.


#----------
```

Software Design - Vitis

For Vitis project creation, follow instructions from:

Vitis

Application

Template location: ./sw_lib/sw_apps/

hello_tec0330

- Hello World as endless loop with output of four frequencies.

scu

- Si5338 I2C Configuration via MicroBlaze MCS.

spi_bootloader

TE modified SPI Bootloader from Henrik Brix Andersen.

Bootloader to load app or second bootloader from flash into DDR.

Here it loads the u-boot.elf from QSPI-Flash to RAM. Hence u-boot.srec becomes redundant.

Descriptions:

- Modified Files: bootloader.c
- Changes:
    - Change the SPI defines in the header
    - Add some reiteration in the frist spi read call

Additional Software

SI5338

File location "<project folder>\misc\Si5338\Si5338-*.slabtimeproj"

General documentation how you work with this project will be available on Si5338

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
			• 2021.2 release

Error rendering macro 'page-info' Ambiguous method overloading for method jdk.proxy24.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page]	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy24.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page]	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy24.$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.

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.

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]	atlassian . confluence.core. ContentEntityObject]	atlassian . confluence.core. ContentEntityObject]	
2018-10-32	v.4	John Hartfiel	• 2018.2 release
--	all	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy24.$Proxy3589#hasContentLevelPermission. Cannot resolve which method to invoke	--

for [null, class java. lang. String, class com. atlassian . confluen ce. pages. Page] due to overlapp ing prototyp es between : [interfac e com. atlassian . confluen ce.user. Conflue nceUser , class java. lang. String, class com. atlassian . confluen ce.core. Content

		EntityObject] [interface com.atlassian.user.User, class java.lang.String, class com.atlassian.confluence.core.ContentEntityObject]	

Document change history.

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]