TE0729 Test Board

Error rendering macro 'toc'

null

mple via VIO Interface.
current online version of this manual and other available documentation.

Key Features

- Vitis/Vivado 2023.2
- PetaLinux
- SD
- ETH (1 x 1 GBit, 2 x 100 MBit)
- USB
- I2C
- RTC
- Watchdog Test example over VIO
- Modified FSBL to optionally select eMMC instead of SD
- Special FSBL for QSPI programming

Revision History

Date	Vivado	Project Built	Authors	Description
2024-06-27	2023.2	TE0729-test_board_noprebuilt-vivado_2023.2-build_4_20240627160410.zip TE0729-test_board-vivado_2023.2-build_4_20240627160410.zip	Waldemar Hanemann	• Vitis update 2023.2
2020-10-01	2019.2	TE0729-test_board_noprebuilt-vivado_2019.2-build_15_20201001141241.zip TE0729-test_board-vivado_2019.2-build_15_20201001141230.zip	John Hartfiel	• Vitis update • FSBL template update • Petalinux webfwu, interface features • uboot doen't check qspi for enviroment • new Script Features
2018-07-16	2018.2	TE0729-test_board-vivado_2018.2-build_02_20180716161110.zip TE0729-test_board_noprebuilt-vivado_2018.2-build_02_20180716161138.zip	John Hartfiel	• initial release

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
Vitis	2023.2	needed, Vivado is included into Vitis installation
PetaLinux	2023.2	needed
SI ClockBuilder Pro	---	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	EMMC	Others	Notes
TE0729-02-2IF	2if_512mb	REV02	512MB	32MB	4GB	2IF-K is the same with head sink	
TE0729-02-62I63MA	2if_512mb	REV02	512MB	32MB	8GB		
TE0729-02-62I63MAK	2if_512mb	REV02	512MB	32MB	8GB	Premounted Heatspreader	
TE0729-03-62I63MAK	2if_512mb	REV03	512MB	32MB	8GB	Premounted Heatspreader	
TE0729-03-62I63MAS	2if_512mb	REV03	512MB	32MB	8GB		
TE0729-03-62I63MA*	2if_512mb	REV03	512MB	32MB	8GB		

*used as reference

Hardware Modules

Design supports following carriers:

Carrier Model	Notes
TEB0729	Used as reference carrier.

Hardware Carrier

Additional HW Requirements:

Additional Hardware	Notes
USB Cable for JTAG/UART	Check Carrier Board and Programmer for correct typ
XMOD Programmer	Carrier Board dependent, only if carrier has no own FTDI

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
PetaLinux	<project folder>\os\petalinux	PetaLinux template with current configuration

Design sources

Additional Sources

Type	Location	Notes
init.sh	<design name>/misc/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	*.xsa	Exported Vivado Hardware Specification for Vitis 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/Vitis/PetaLinux version. Do never use different Versions of AMD Software for the same Project.

Reference Design is available on:

- TE0729 "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 AMD 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 AMD 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. 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 Vitis 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

4. 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.

5. Create and configure your PetaLinux project with exported .xsa-file, see PetaLinux KICKstart
    - use TE Template from "<project folder>\os\petalinux"
    - use exported .xsa file from "<project folder>\prebuilt\hardware\<short name>" . **Note:** HW Export from Vivado GUI creates another path as default workspace.
    - The build images are located in the "<plnx-proj-root>/images/linux" directory
6. (Optional) Configure the **boot.scr** file as needed, see Distro Boot with Boot.scr
7. Generate Programming Files with Vitis (recommended)
    a. Copy PetaLinux build image files to prebuilt folder
        - copy **u-boot.elf**, **system.dtb**, **image.ub** and **boot.scr** from "<plnx-proj-root>/images/linux" to prebuilt folder

"<project folder>\prebuilt\os\petalinux\<ddr size>" or "<project folder>\prebuilt\os\petalinux\<short name>"

b. 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

8. Generate Programming Files with Petalinux (alternative), see PetaLinux KICKstart

Launch

Programming

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

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

Get prebuilt boot binaries

1. _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 foler>/_binaries_<Artikel Name>) with subfolder (boot_<app name>) for different applications will be generated

QSPI-Boot mode

Option for **Boot.bin** on QSPI Flash and **image.ub** and **boot.scr** on **SD**.

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"

run on Vivado TCL (Script programs BOOT.bin on QSPI flash)

```
TE::pr_program_flash -swapp u-boot
TE::pr_program_flash -swapp hello_te0820 (optional)
```

3. Copy **image.ub** and **boot.scr** on **SD**
   - use files from "<project folder>_binaries_<Article Name>\boot_linux" from generated binary folder, see: Get prebuilt boot binaries
   - or use prebuilt file location, see "<project folder>\prebuilt\file_location.txt"

4. Set Boot Mode to **QSPI-Boot** and insert **SD**
    - Depends on Carrier, see carrier TRM.
5. insert SD-Card.

SD-Boot mode

1. Copy **image.ub, boot.scr** and **Boot.bin** on **SD**
    - use files from "<project folder>_binaries_<Article Name>\boot_linux" from generated binary folder, see: Get prebuilt boot binaries
    - or use prebuilt file location, see "<project folder>\prebuilt\file_location.txt"
2. Set Boot Mode to SD-Boot.
    - Depends on Carrier, see carrier TRM.
3. Insert SD-Card in SD-Slot.

JTAG

Not used on this Example.

Usage

1. Prepare HW like described on section Programming
2. Connect UART USB (most cases same as JTAG)
3. Select SD Card as Boot Mode (or QSPI - depending on step 1)

> Note: See TRM of the Carrier, which is used.

> Starting with Petalinux version 2020.1, the industry standard "Distro-Boot" boot flow for U-Boot was introduced, which significantly expands the possibilities of the boot process and has the primary goal of making booting much more standardised and predictable.
> The boot options described above describe the common boot processes for this hardware; other boot options are possible.
> For more information see Distro Boot with Boot.scr

4. Power On PCB

    1. Zynq Boot ROM loads FSBL from SD/QSPI into OCM,

    2. FSBL init PS, programs PL using the bitstream and loads U-boot from SD into DDR,

    3. U-boot loads Linux (**image.ub**) from SD/QSPI/... into DDR

Linux

1. Open Serial Console (e.g. putty)
    - Speed: 115200
    - select COM Port

    > Win OS, see device manager, Linux OS see dmesg |grep tty (UART is *USB1)

2. Linux Console(not needed if autologin activated):

```
# password disabled
petalinux login: root
Password: root
```

3. You can use Linux shell now.

```
i2cdetect -y -r 0        (check I2C 0 Bus)
dmesg | grep rtc        (RTC check)
udhcpc                                (ETH0 check)
lsusb                                (USB check)
```

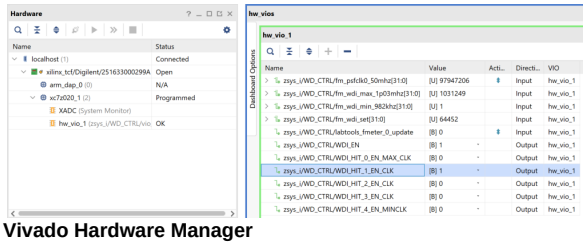
4. Optional Features

    - Webserver to get access to Zynq
        - insert IP on web browser to start web interface
    - init.sh scripts
        - add init.sh script on SD, content will be load automatically on startup (template included in "<project folder>\misc\SD")

Vivado HW Manager

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

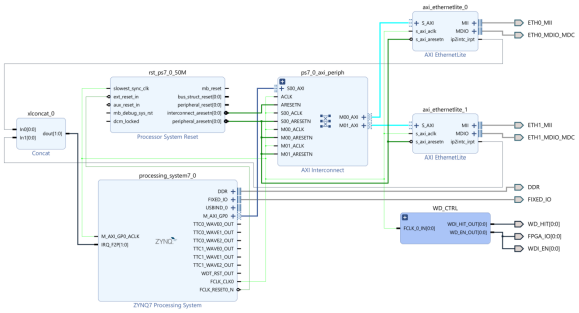
- Control:
    - "WDI_EN" and "WDI_HIT_*_EN_CLK" enables FPGA watchdog control.
    - Force WD to system reboot:
        1. Check on Hardware window VIO status is ok. (right click on vio symbol and click "commit output values to VIO core" for update).
        2. Enable one of the "WDI_HIT_*_EN_CLK" signals
        3. Enable "WDI_EN"
        4. To force system to reboot, disable WDI_HIT clocks.
- Monitoring:
    - Set radix for "fm_*" signals to unsigned integer to see frequ in Hz. PSFCLK0 50MHz was increased to 100MHz with version update 2023.2
    - "fm_*" shows some clk frequencies (unit Hz). Note: inaccurate Reference CLK is used for frequency measurement.



Vivado Hardware Manager

System Design - Vivado

Block Design



Block Design

PS Interfaces

Type	Note
Typ	Note
DDR	
QSPI	MIO
SD0	MIO
I2C0	MIO
I2C1	MIO
UART0	MIO
GPIO0	MIO
SWDT0	
TTC0..1	
ETH00	MIO
USB0	MIO
PL-PS IRQ	

PS Interfaces

Constraints

Basic module constraints

_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 constraints

_i_io.xdc

```
set_property PACKAGE_PIN F16 [get_ports {FPGA_IO[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {FPGA_IO[0]}]
set_property PACKAGE_PIN H15 [get_ports {WDI_EN[0]}]
set_property IOSTANDARD LVCMOS25 [get_ports {WDI_EN[0]}]
set_property PACKAGE_PIN R15 [get_ports {WD_HIT[0]}]
set_property IOSTANDARD LVCMOS25 [get_ports {WD_HIT[0]}]
```

Software Design - Vitis

For Vitis project creation, follow instructions from:

Vitis

Application

Source location: \sw_lib\sw_apps

zynq_fsbl

TE modified 2023.2 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_*
    - Optional define for eMMC selection with FSBL (default SD selected)
        - uncomment #define USE_EMMC on te_fsbl_hooks_te0729.h to select eMMC instead of SD

zynq_fsbl_flash

TE modified 2019.2 FSBL

General:

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

hello_te0729

Hello TE0729 is a Vitis Hello World example as endless loop instead of one console output.

u-boot

U-Boot.elf is generated with PetaLinux. VitisI is used to generate Boot.bin.

Software Design - PetaLinux

For PetaLinux installation and project creation, follow instructions from:

- PetaLinux KICKstart

Config

The project folder contains a folder named "os". This is a template for the TE0729 petalinux project with all necessary changes applied. You can just add the XSA file and build the project to get working binaries.

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

Changes:

- add new flash partition for bootscr and sizing
    - CONFIG_SUBSYSTEM_FLASH_PS7_QSPI_0_BANKLESS_PART0_SIZE=0x0100000
    - CONFIG_SUBSYSTEM_FLASH_PS7_QSPI_0_BANKLESS_PART1_SIZE=0x1400000
    - CONFIG_SUBSYSTEM_FLASH_PS7_QSPI_0_BANKLESS_PART2_SIZE=0x0020000
    - CONFIG_SUBSYSTEM_FLASH_PS7_QSPI_0_BANKLESS_PART3_NAME="bootscr"
    - CONFIG_SUBSYSTEM_FLASH_PS7_QSPI_0_BANKLESS_PART3_SIZE=0x40000

U-Boot

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

Changes:

- CONFIG_ENV_OVERWRITE=y

Boot Modes:

- CONFIG_QSPI_BOOT=y
- CONFIG_SD_BOOT=y
- CONFIG_BOOT_SCRIPT_OFFSET=0x1520000
  (Calculate the start address of partition 3 "bootscr" in the QSPI flash. To do this, add the sizes of partitions 0, 1 and 2 together)

Device Tree

```
/include/ "system-conf.dtsi"
/ {
  chosen {
    xlnx,eeprom = &eeprom;
  };
};

/* QSPI PHY */
&qspi {
    #address-cells = <1>;
    #size-cells = <0>;
    status = "okay";
    flash0: flash@0 {
        compatible = "jedec,spi-nor";
        reg = <0x0>;
        #address-cells = <1>;
        #size-cells = <1>;
    };
};


/* ETH PHY */
&gem0 {
    phy-handle = <&phy0>;
    mdio {
        #address-cells = <1>;
        #size-cells = <0>;
        phy0: phy@0 {
            compatible = "marvell,88e1510";
            device_type = "ethernet-phy";
            reg = <0>;
        };
    };
};

/* AXI ETH PHY0 */

&axi_ethernetlite_0 {
    local-mac-address = [00 0a 35 00 22 02];
    phy-handle = <&phy1>;
```

```
    xlnx,has-mdio = <0x1>;
    mdio {
        #address-cells = <1>;
        #size-cells = <0>;
        phy1: phy@1 {
            device_type = "ethernet-phy";
            reg = <1>;
        };
    };
};

/* AXI ETH PHY1 */
&axi_ethernetlite_1 {
    local-mac-address = [00 0a 35 00 22 03];
    phy-handle = <&phy2>;
    xlnx,has-mdio = <0x1>;
    mdio {
        #address-cells = <1>;
        #size-cells = <0>;
        phy2: phy@1 {
            device_type = "ethernet-phy";
            reg = <1>;
        };
    };
};


/* RTC */
&i2c0 {
    rtc@6F {        // Real Time Clock
       compatible = "isl12022";
       reg = <0x6F>;
   };
  //MAC EEPROM
  eeprom: eeprom@50 {
    compatible = "atmel,24c08";
    reg = <0x54>;
  };
};

/* USB PHY */

/{
    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:

- CONFIG_RTC_DRV_ISL12022=y

Rootfs

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

Changes:

- For web server app:
  - CONFIG_busybox-httpd=y
- For additional test tools only:
  - CONFIG_i2c-tools=y
  - CONFIG_packagegroup-petalinux-utils=y (util-linux,cpufrequtils,bridge-utils,mtd-utils,usbutils, pciutils,canutils,i2c-tools,smartmontools,e2fsprogs)
  - CONFIG_util-linux-umount=y
  - CONFIG_util-linux-mount=y
- For usage of phytool:
  - CONFIG_ethtool=y
- For auto login:
  - CONFIG_auto-login=y

Applications

See "<project folder>\os\petalinux\project-spec\meta-user\recipes-apps\"

startup

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

webfwu

Webserver application accemble for Zynq access. Need busybox-httpd

Core

To enable dhcp for all ethernet interfaces create path like: see "<project folder>\os\petalinux\project-spec\meta-user\recipes-core\init-ifupdown"

in petalinux build directory. Documented here Petalinux Yocto tips under " Method 2 : use your own interfaces file".

Additional Software

No additional software is needed.

App. 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.proxy241.$Proxy3496#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes between: [interface com.atlassian.	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy241.$Proxy3496#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.	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy241.$Proxy3496#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.	• 2023.2 Release

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.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.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]	
2020-10-01	v.6	John Hartfiel	• 2019.2 Release
2018-07-06	v.5	John Hartfiel	• 2018.2 Release
	All	**Error rendering macro 'page-info'**	

Ambiguous method overloading for method jdk.proxy241.$Proxy3496#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

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.proxy241.$Proxy3496#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]