TE0802 Test Board

Table of contents

Overview

Refer to http://trenz.org/te0802-info for the current online version of this manual and other available documentation.

Key Features

- Vitis/Vivado 2022.2
- PetaLinux
- SD
- ETH
- USB
- I2C
- Display Port (DP)
- VGA
- DIPS, LEDS, Buttons
- Audio
- MAC from EEPROM
- Modified FSBL for Resets

Revision History

Date	Vivado	Project Built	Authors	Description
2023-06-28	2022.2	TE0802-test_board-vivado_2022.2-build_2_20230628100458.zip TE0802-test_board_noprebuilt-vivado_2022.2-build_2_20230628100458.zip	Manuela Strücker	• 2022.2 update • new assembly variant
2023-02-10	2021.2.1	TE0802-test_board-vivado_2021.2-build_20_20230210132253.zip TE0802-test_board_noprebuilt-vivado_2021.2-build_20_20230210132253.zip	Manuela Strücker	• bugfix display port hot plug detection
2022-12-08	2021.2.1	TE0802-test_board-vivado_2021.2-build_20_20221208094356.zip TE0802-test_board_noprebuilt-vivado_2021.2-build_20_20221208094356.zip	Manuela Strücker	• script update

2022-08-24	2021.2.1	TE0802-test_board-vivado_2021.2-build_15_20220824130139.zip TE0802-test_board_noprebuilt-vivado_2021.2-build_15_20220824130139.zip	Manuela Strücker	• 2021.2.1 update • new assembly variants
2020-06-02	2019.2	TE0802-test_board-vivado_2019.2-build_12_20200602111955.zip TE0802-test_board_noprebuilt-vivado_2019.2-build_12_20200602112010.zip	John Hartfiel	• add NVME drivers
2019-08-30	2018.3	TE0802-test_board-vivado_2018.3-build_07_20190830103019.zip TE0802-test_board_noprebuilt-vivado_2018.3-build_07_20190830103313.zip	Oleksandr Kiyenko, 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	2022.2	needed, Vivado is included into Vitis installation
PetaLinux	2022.2	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 "<project folder>\board_files*_board_files.csv"

Design supports following modules:

Module Model	Board Part Short Name	PCB Revision Support	DDR	QSPI Flash	EMMC	Others	Notes
TE0802-02-2AEV2-A	2cg_s1gb	REV02	1GB	32MB	NA	NA	Samsung DDR4L
TE0802-02-2AEU2-A	2cg_i1gb	REV02	1GB	32MB	NA	NA	ISSI DDR4L
TE0802-02-1AEV2-A*	1cg_s1gb	REV02	1GB	32MB	NA	NA	Samsung DDR4L
TE0802-02-1BEV2-A*	1eg_s1gb	REV02	1GB	32MB	NA	NA	Samsung DDR4L

*used as reference

Hardware Modules

Design supports following carriers:

Carrier Model	Notes
---	

*used as reference

Hardware Carrier

Additional HW Requirements:

Additional Hardware	Notes
M2 SSD	tested with Samsung 050 Pro 256GB
headphones	
Monitor with DP support	Note: not all monitors will be supported by Xilinx. Adapter to other connector standard is not supported

*used as reference

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	<project folder>\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
Boot Script-File	*.scr	Distro Boot Script file
DebugProbes-File	*.ltx	Definition File for Vivado/Vivado Labtools Debugging Interface
Diverse Reports	---	Report files in different formats
Device Tree	*.dts	Device tree (2 possible, one for u-boot and one for linux)
Hardware-Platform-Description-File	*.xsa	Exported Vivado hardware description file 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 Xilinx Software for the same Project.

Reference Design is available on:

- TE0802 "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 launch.

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/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 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

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. Configure the **boot.scr** file as needed, see Distro Boot with Boot.scr
7. Generate Programming Files with Vitis
    a. Copy PetaLinux build image files to prebuilt folder
        - copy **u-boot.elf, system.dtb, bl31.elf**, **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

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.

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 generated

QSPI-Boot mode

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

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_te0802 (optional)
```


To program with Vitis/Vivado GUI, use special FSBL (fsbl_flash) on setup

3. Copy **image.ub** and **boot.scr** on **SD** or **USB**
   - 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** or **USB**.
   - Depends on Carrier, see carrier TRM.

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. Connect Monitors, ETH, M2...
4. 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

5. 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:

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

   > Note: Wait until Linux boot finished

3. You can use Linux shell now.

```
I2C
        i2cdetect -l        (Shows a list of the available I2C
buses)
        i2cdetect -y -r 0        (check I2C 0 Bus)
RTC
        dmesg | grep rtc        (RTC check)
ETH0
        udhcpc                                  (ETH0 check)
USB
        lsusb                                  (USB check)
PCIe (M2 SSD)
        lspci                                  (PCIe check)
Audio
        aplay /<link to mounted sd card>/<filename>.wav  (e.g. aplay
/run/mount/sd/<filename>.wav)
        Note: Display Port must be connected to activate audio
drivers. Use .wav or other aplay supported formate
VGA
        connect VGA to monitor and adjust source (it shows test
pattern)
Display port
        second console will be shown on the monitor, when boot
process is finished.
        Note: connect keyboard to TE0802 USB, to interact with the
second console
                petalinux login: root
                Password: root
```

4. Option Features

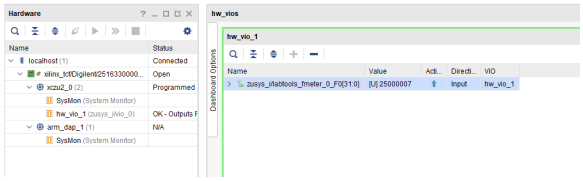
   - Webserver to get access to Zynq
     - insert IP on web browser to start web interface
   - init.sh scripts

    - add init.sh script on SD, content will be load automatically on startup (template included in "<project folder>\misc\SD")
5. All button cross will be reset LEDs with values from DIP
6. LCD is connected to counter

Vivado HW Manager

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

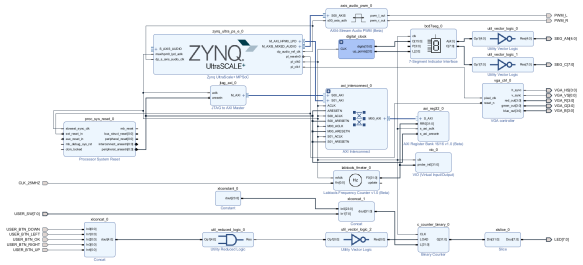
- Monitoring:
    - 25MHz CLK Set radix from VIO signals to unsigned integer. Note: Frequency Counter is inaccurate and displayed unit is Hz



Vivado Hardware Manager

System Design - Vivado

Block Design



Block Design

PS Interfaces

Activated interfaces:

Type	Note
DDR	
QSPI	MIO
SD0	MIO
I2C0	MIO
I2C1	MIO
UART0	MIO
GPIO0	MIO
GPIO1	MIO

GPIO2	MIO
SWDT0..1	
TTC0..3	
GEM3	MIO
USB0	MIO + GT Lane 1
PCIe	MIO + GT Lane 0 (as rootcomplex)
DP	MIO + GT Lane 2

PS Interfaces

Constrains

Basic module constrains

_i_bitgen_common.xdc

```
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design]
```

Design specific constrain

_i_io.xdc

```
set_property PACKAGE_PIN E3 [get_ports PWM_L]
set_property PACKAGE_PIN F4 [get_ports PWM_R]
set_property IOSTANDARD LVCMOS18 [get_ports PWM_*]

set_property PACKAGE_PIN P3 [get_ports {USER_SW[0]}]
set_property PACKAGE_PIN P2 [get_ports {USER_SW[1]}]
set_property PACKAGE_PIN M1 [get_ports {USER_SW[2]}]
set_property PACKAGE_PIN L1 [get_ports {USER_SW[3]}]
set_property PACKAGE_PIN K1 [get_ports {USER_SW[4]}]
set_property PACKAGE_PIN J2 [get_ports {USER_SW[5]}]
set_property PACKAGE_PIN M4 [get_ports {USER_SW[6]}]
set_property PACKAGE_PIN M5 [get_ports {USER_SW[7]}]
set_property IOSTANDARD LVCMOS18 [get_ports USER_SW*]

set_property PACKAGE_PIN U2 [get_ports {USER_BTN_UP}]
set_property PACKAGE_PIN U1 [get_ports {USER_BTN_RIGHT}]
set_property PACKAGE_PIN T2 [get_ports {USER_BTN_DOWN}]
set_property PACKAGE_PIN R1 [get_ports {USER_BTN_LEFT}]
set_property PACKAGE_PIN T1 [get_ports {USER_BTN_OK}]
set_property IOSTANDARD LVCMOS18 [get_ports USER_BTN*]

set_property PACKAGE_PIN P1 [get_ports {LED[0]}]
set_property PACKAGE_PIN N2 [get_ports {LED[1]}]
set_property PACKAGE_PIN M2 [get_ports {LED[2]}]
set_property PACKAGE_PIN L2 [get_ports {LED[3]}]
set_property PACKAGE_PIN J1 [get_ports {LED[4]}]
set_property PACKAGE_PIN H2 [get_ports {LED[5]}]
set_property PACKAGE_PIN L4 [get_ports {LED[6]}]
```

```
set_property PACKAGE_PIN L3 [get_ports {LED[7]}]
set_property IOSTANDARD LVCMOS18 [get_ports LED*]

set_property PACKAGE_PIN F2 [get_ports {VGA_R[0]}]
set_property PACKAGE_PIN F1 [get_ports {VGA_R[1]}]
set_property PACKAGE_PIN G2 [get_ports {VGA_R[2]}]
set_property PACKAGE_PIN G1 [get_ports {VGA_R[3]}]
set_property PACKAGE_PIN C2 [get_ports {VGA_G[0]}]
set_property PACKAGE_PIN D2 [get_ports {VGA_G[1]}]
set_property PACKAGE_PIN D1 [get_ports {VGA_G[2]}]
set_property PACKAGE_PIN E1 [get_ports {VGA_G[3]}]
set_property PACKAGE_PIN A3 [get_ports {VGA_B[0]}]
set_property PACKAGE_PIN A2 [get_ports {VGA_B[1]}]
set_property PACKAGE_PIN B2 [get_ports {VGA_B[2]}]
set_property PACKAGE_PIN B1 [get_ports {VGA_B[3]}]
set_property PACKAGE_PIN B7 [get_ports {VGA_VS[0]}]
set_property PACKAGE_PIN A6 [get_ports {VGA_HS[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_B[3]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_B[2]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_B[1]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_B[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_G[3]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_G[2]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_G[1]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_G[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {VGA_HS[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_R[3]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_R[2]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_R[1]}]
set_property IOSTANDARD LVCMOS18 [get_ports {VGA_R[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {VGA_VS[0]}]

set_property PACKAGE_PIN J3 [get_ports CLK_25MHZ]
set_property IOSTANDARD LVCMOS18 [get_ports CLK_25MHZ]
# SEG_C[0] = SEG_CA
set_property PACKAGE_PIN E4 [get_ports {SEG_C[0]}]
set_property PACKAGE_PIN D3 [get_ports {SEG_C[1]}]
set_property PACKAGE_PIN N5 [get_ports {SEG_C[2]}]
set_property PACKAGE_PIN P5 [get_ports {SEG_C[3]}]
set_property PACKAGE_PIN N4 [get_ports {SEG_C[4]}]
set_property PACKAGE_PIN C3 [get_ports {SEG_C[5]}]
set_property PACKAGE_PIN N3 [get_ports {SEG_C[7]}]
set_property PACKAGE_PIN R5 [get_ports {SEG_C[6]}]
set_property IOSTANDARD LVCMOS18 [get_ports SEG_C*]

set_property PACKAGE_PIN A8 [get_ports {SEG_AN[0]}]
set_property PACKAGE_PIN A9 [get_ports {SEG_AN[1]}]
set_property PACKAGE_PIN B9 [get_ports {SEG_AN[2]}]
set_property PACKAGE_PIN A7 [get_ports {SEG_AN[3]}]
set_property PACKAGE_PIN B6 [get_ports {SEG_AN[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports SEG_AN*]
```

Software Design - Vitis

For Vitis project creation, follow instructions from:

Vitis

Application

Template location: "<project folder>\sw_lib\sw_apps\"

zynqmp_fsbl

TE modified 2022.2 FSBL

General:

- Modified Files: xfsbl_main.c, xfsbl_hooks.h/.c, xfsbl_board.h/.c (search for 'TE Mod' on source code)
- Add Files: te_xfsbl_hooks.h/.c (for hooks and board)
- General Changes:
    - Display FSBL Banner and Device Name

Module Specific:

- Add Files: all TE Files start with te_*
    - Si5338 Configuration
    - ETH+OTG Reset over MIO

zynqmp_pmufw

Xilinx default PMU firmware.

hello_te0802

Hello TE0802 is a Xilinx Hello World example as endless loop instead of one console output.

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:

- add new flash partition for bootscr and sizing
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART0_SIZE=0xA00000
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART1_SIZE=0x1500000
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART2_SIZE=0x40000
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART3_NAME="bootscr"
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART3_SIZE=0x40000
- Identification
    - CONFIG_SUBSYSTEM_HOSTNAME="Trenz"
    - CONFIG_SUBSYSTEM_PRODUCT="TE0802"

U-Boot

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

Changes:

- MAC from eeprom together with uboot and device tree settings:
  - CONFIG_ENV_OVERWRITE=y
  - CONFIG_ZYNQ_MAC_IN_EEPROM is not set
  - CONFIG_NET_RANDOM_ETHADDR is not set
- Boot Modes:
  - CONFIG_QSPI_BOOT=y
  - CONFIG_SD_BOOT=y
  - CONFIG_ENV_IS_IN_FAT is not set
  - CONFIG_ENV_IS_IN_NAND is not set
  - CONFIG_ENV_IS_IN_SPI_FLASH is not set
  - CONFIG_SYS_REDUNDAND_ENVIRONMENT is not set
  - CONFIG_BOOT_SCRIPT_OFFSET=0x1F40000
- Identification
  - CONFIG_IDENT_STRING=" TE0802"

Change platform-top.h:

```
# no changes
```

Device Tree

project-spec\meta-user\recipes-bsp\device-tree\files\system-user.dtsi

```
/include/ "system-conf.dtsi"


/*------------------ gtr --------------------*/

//https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841716
/Zynq+Ultrascale+MPSOC+Linux+SIOU+driver

/ {
  refclk2:psgtr_dp_clock {
          compatible = "fixed-clock";
          #clock-cells = <0x00>;
          clock-frequency = <27000000>;
  };

   refclk1:psgtr_usb_clock {
           compatible = "fixed-clock";
           #clock-cells = <0x00>;
           clock-frequency = <26000000>;
   };

   refclk0:psgtr_pcie_clock {
           compatible = "fixed-clock";
           #clock-cells = <0x00>;
           clock-frequency = <100000000>;
   };

  //refclk1:psgtr_sata_clock {
  //        compatible = "fixed-clock";
  //        #clock-cells = <0x00>;
```

```
  //        clock-frequency = <150000000>;
  //};

  //refclk0:psgtr_unused_clock {
  //        compatible = "fixed-clock";
  //        #clock-cells = <0x00>;
  //        clock-frequency = <100000000>;
  //};
};

&psgtr {
  clocks = <&refclk0 &refclk1 &refclk2>;
  /* ref clk instances used per lane */
  clock-names = "ref0\0ref1\0ref2";
};


/*------------------ SD --------------------*/
&sdhci0 {
    disable-wp;
    no-1-8-v;
};


/*------------------ USB --------------------*/
&dwc3_0 {
    status = "okay";
    dr_mode = "host";
    snps,usb3_lpm_capable;
    snps,dis_u3_susphy_quirk;
    snps,dis_u2_susphy_quirk;
    phy-names = "usb2-phy","usb3-phy";
    maximum-speed = "super-speed";
};


/*------------------ LEDs --------------------*/
#include <dt-bindings/gpio/gpio.h>
#include <dt-bindings/leds/common.h>

/ {
    leds {
        compatible = "gpio-leds";
        ndp_en {
            label = "ndp_en";
            gpios = <&gpio 26 GPIO_ACTIVE_HIGH>;
            default-state = "on";
        };
        ssd_sleep {
            label = "ssd_sleep";
            gpios = <&gpio 32 GPIO_ACTIVE_HIGH>;
            default-state = "on";
        };
        usb_reset {
            label = "usb_reset";
            gpios = <&gpio 38 GPIO_ACTIVE_HIGH>;
            default-state = "on";
        };
    };
};
```

```
/*------------------ ETH PHY --------------------*/
&gem3 {
    phy-handle = <&phy0>;

    nvmem-cells = <&eth0_addr>;
    nvmem-cell-names = "mac-address";

    phy0: phy0@1 {
        device_type = "ethernet-phy";
        reg = <1>;
    };
};


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


/*------------------ I2C --------------------*/
&i2c1 {
    eeprom: eeprom@50 {
        compatible = "microchip,24aa025", "atmel,24c02";
        reg = <0x50>;

        #address-cells = <1>;
        #size-cells = <1>;
        eth0_addr: eth-mac-addr@FA {
          reg = <0xFA 0x06>;
        };

    };
};
```

Kernel

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

Changes:

- Only needed to fix JTAG Debug issue:
  - # CONFIG_CPU_FREQ is not set
- Support PCIe memory card
  - CONFIG_NVME_CORE=y
  - CONFIG_BLK_DEV_NVME=y
  - # CONFIG_NVME_MULTIPATH is not set
  - # CONFIG_NVME_HWMON is not set
  - # CONFIG_NVME_TCP is not set
  - CONFIG_NVME_TARGET=y
  - # CONFIG_NVME_TARGET_PASSTHRU is not set

    - # CONFIG_NVME_TARGET_LOOP is not set
    - # CONFIG_NVME_TARGET_FC is not set
    - # CONFIG_NVME_TARGET_TCP is not set
    - CONFIG_SATA_AHCI=y
    - CONFIG_SATA_MOBILE_LPM_POLICY=0

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)
- For audio application
    - CONFIG_alsa-utils=y
    - CONFIG_alsa-utils-aplay=y
- For auto login:
    - CONFIG_auto-login=y
    - CONFIG_ADD_EXTRA_USERS="root:root;petalinux:;"

FSBL patch (alternative for vitis fsbl trenz patch)

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

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 suitable for Zynq access. Need busybox-httpd

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.proxy279.$Proxy4022#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluen	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy279.$Proxy4022#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluen	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy279.$Proxy4022#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluen	• 2022.2 • new assembly variant

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
EntityOb
ject]
[interfac
e com.
atlassian
.user.
User,
class
java.
lang.

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
EntityOb
ject]
[interfac
e com.
atlassian
.user.
User,
class
java.
lang.

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
EntityOb
ject]
[interfac
e com.
atlassian
.user.
User,
class
java.
lang.

String, class com. atlassian . confluen ce.core. Content EntityOb ject]	String, class com. atlassian . confluen ce.core. Content EntityOb ject]	String, class com. atlassian . confluen ce.core. Content EntityOb ject]	
2023-02-10	v.9	Manuela Strücker	• bugfix display port hot plug detection
2022-12-08	v.8	Manuela Strücker	• script update
2022-09-09	v.7	Manuela Strücker	• 2021.2.1 • new assembly variant
2020-06-03	v.2	John Hartfiel	• 2019.2
2019-08-30	v.1	John Hartfiel	• 2018.3
--	all	**Error renderi ng macro 'page-info'** Ambiguo us method overload ing for method jdk.	--

proxy279.$Proxy4022#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.

		Conflue nceUser , class java. lang. String, class com. atlassian . confluen ce.core. Content EntityOb ject] [interfac e com. atlassian .user. User, class java. lang. String, class com. atlassian . confluen ce.core. Content EntityOb ject]	

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

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.proxy279.$Proxy4022#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]