TE0950 Test Board

Table of contents

Overview

Versal PS Design with Linux Example. HW-Manager.

Wiki Resources page: https://wiki.trenz-electronic.com/wiki/TE0950-info

Key Features

- Versal Prime VM1102 Fixed Exam
- Vitis/Vivado 2023.2.1
- PetaLinux
- SD
- eMMC
- ETH
- USB
- I2C
- MIPI-CSI2
- MAC from EEPROM
- User LEDs
- with Artix Reference Design Counterpart test_board_artix
  - to Artix: Chip2Chip connection
  - PWM Fan control via AXI Timer IP Core
  - to Artix: 3-wire I2C Multiplexer

Revision History

Date	Vivado	Project Built	Authors	Description

2024-06-04	2023.2.1	TE0950-test_board-vivado_2023.2-build_4_20240531092954.zip
		
		TE0950-test_board_artix_noprebuilt-vivado_2023.2-build_4_20240531084104.zip
		
		TE0950-test_board_artix-vivado_2023.2-build_4_20240531084104.zip
		
		TE0950-test_board_noprebuilt-vivado_2023.2-build_4_20240531092954.zip
			Markus Kirberg	• 2023.2 release
				  ◦ fixes for
				  ▪ Versal: unconstrained MIPI I2C Interface
				  ▪ Versal: QSPI access via Linux
				  ▪ Artix: Reset for I2C-MUX fixes unreliable start up behaviour
				  ◦ Artix: add fan control via PWM with AXI Timer IP Core

2024-02-01	2023.2.1	TE0950-test_board-vivado_2023.2-build_4_20240116133227.zip TE0950-test_board_noprebuilt-vivado_2023.2-build_4_20240116133227.zip TE0950-test_board_artix-vivado_2023.2-build_4_20240118214742.zip TE0950-test_board_artix_noprebuilt-vivado_2023.2-build_4_20240118214742.zip	Markus Kirberg	• 2023.2 release

Design Revision History

Release Notes and Know Issues

Issues	Description	Workaround	To be fixed version
Known Issues			

Requirements

Software

Software	Version	Note
Vivado	2023.2.1	needed (Note: only **2023.2.1** contains production level support for xcve2302 and is **required**, otherwise additional licensing issues will appear) **(using -es1 Parts need):** • **Installation of the ES Parts** • License for ES Part-Devices ([part]-**es1 and** [part]-**es1_bitgen**) • activation of Beta Devices in Vivado install folder Vivado\2023.2\scripts\Vivado_init.tcl via `enable_beta_device xcve*`
Vitis	2023.2	needed, Vitis is included in Vivado installation
PetaLinux	2023.2	needed

Vitis HLS	2023.2	needed (used for MIPI-Camera Pipeline) Vitis HLS is included optionally in Vivado installation **(using -es1 Parts need):** • activation of Beta Devices in Vitis_HLS install folder Vitis_HLS\2023.2\scripts\HLS_init.tcl via `enable_beta_device xcve*`

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
TE0950-02-EGBE21A	23_1lse_8gb_es1	REV02	8GB	128MB	32GB	NA	NA
TE0950-03-EGBE21A*	23_1lse_8gb	REV03	8GB	128MB	32GB	NA	NA

*used as reference

Hardware Modules

Additional HW Requirements:

Additional Hardware	Notes
USB Cable for JTAG/UART	Check Carrier Board and Programmer for correct type

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 (Versal-FPGAs)
BIT-File	*.pdi	FPGA 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
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 prebuilt 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:

- TE0950 "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. _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.
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
        i. 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.

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

Note: Depending on Boot Mode settings, QSPI boot with Linux image on SD or complete SD Boot is possible.

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 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, dtbos (folder)** 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`

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

3. Copy **image.ub, dtbos (folder)** 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**.

SD-Boot mode

1. Copy **image.ub, boot.src, dtbos (folder)** 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.
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. Versal Boot ROM loads PLM from SD/QSPI into OCM,

    2. PLM init the PS, programs the PL using the bitstream and loads PMU, ATF and U-boot from SD/QSPI 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:

> Note: Wait until Linux boot finished

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. Option Features

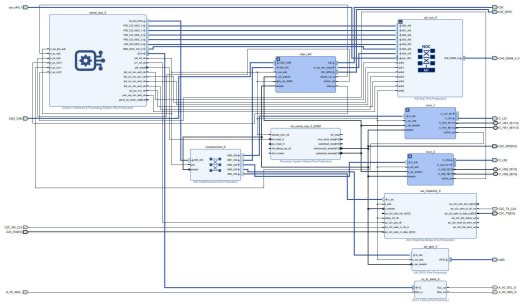
    - init.sh scripts
        - add init.sh script on SD, content will be load automatically on startup (template included in "<project folder>\misc\SD")

Vivado HW Manager

System Design - Vivado

Block Design

PCB REV03



Block Design PCB REV03

PS Interfaces

Activated interfaces:

Type	Note
DDR	
QSPI	MIO
SD0/eMMC	MIO
SD1/SD2.0	MIO
PMC_I2C	MIO
UART1	MIO
LPD_IC20	EMIO
LPD_IC21	MIO
TTC0..3	
GEM0	MIO
USB0	MIO, USB2.0

Constrains

Design specific constraints

_i_io.xdc

```
# CRUVI LOW SPEED 1
set_property PACKAGE_PIN C12 [get_ports {C_LS1_tri_io[7]}]; #C_LS1_SDA
set_property PACKAGE_PIN A11 [get_ports {C_LS1_tri_io[6]}]; #C_LS1_SCL
```

```
set_property PACKAGE_PIN B11 [get_ports {C_LS1_tri_io[5]}]; #C_LS1_D3
set_property PACKAGE_PIN B10 [get_ports {C_LS1_tri_io[4]}]; #C_LS1_D2
set_property PACKAGE_PIN C10 [get_ports {C_LS1_tri_io[3]}]; #C_LS1_D1
set_property PACKAGE_PIN D10 [get_ports {C_LS1_tri_io[2]}]; #C_LS1_D0

set_property PACKAGE_PIN D11 [get_ports {C_LS1_tri_io[1]}]; #C_LS1_SCK
set_property PACKAGE_PIN A10 [get_ports {C_LS1_tri_io[0]}]; #C_LS1_SEL


set_property IOSTANDARD LVCMOS33 [get_ports {C_LS1_tri_io*}]

# CRUVI LOW SPEED 2
set_property PACKAGE_PIN E12 [get_ports {C_LS2_tri_io[7]}]; #C_LS2_SDA
set_property PACKAGE_PIN F14 [get_ports {C_LS2_tri_io[6]}]; #C_LS2_SCL

set_property PACKAGE_PIN E13 [get_ports {C_LS2_tri_io[5]}]; #C_LS2_D3
set_property PACKAGE_PIN D14 [get_ports {C_LS2_tri_io[4]}]; #C_LS2_D2
set_property PACKAGE_PIN C14 [get_ports {C_LS2_tri_io[3]}]; #C_LS2_D1
set_property PACKAGE_PIN D12 [get_ports {C_LS2_tri_io[2]}]; #C_LS2_D0

set_property PACKAGE_PIN C13 [get_ports {C_LS2_tri_io[1]}]; #C_LS2_SCK
set_property PACKAGE_PIN E14 [get_ports {C_LS2_tri_io[0]}]; #C_LS2_SEL

set_property IOSTANDARD LVCMOS33 [get_ports {C_LS2_tri_io*}]

#B302 HD
set_property PACKAGE_PIN F11 [get_ports {CSI_GPIO_tri_io[0]}]; #CSI_GPIO0
set_property PACKAGE_PIN E11 [get_ports {CSI_GPIO_tri_io[1]}]; #CSI_GPIO1

set_property IOSTANDARD LVCMOS33 [get_ports {CSI_GPIO_tri_io*}]


set_property PACKAGE_PIN B12 [get_ports {USR_tri_io[1]}]; #V_USR_LED1
set_property PACKAGE_PIN A14 [get_ports {USR_tri_io[0]}]; #V_PL_USR_SW
set_property IOSTANDARD LVCMOS33 [get_ports {USR_tri_io*}]


### CRUVI HS1 ######
set_property IOSTANDARD DIFF_HSTL_I_12 [get_ports {C_HS1_P[*]}]
set_property PACKAGE_PIN D27 [get_ports {C_HS1_P[11]}]; #HS1_B5
set_property PACKAGE_PIN G27 [get_ports {C_HS1_P[10]}]; #HS1_B4
set_property PACKAGE_PIN H27 [get_ports {C_HS1_P[9]}];  #HS1_B3
set_property PACKAGE_PIN J27 [get_ports {C_HS1_P[8]}];  #HS1_B2
set_property PACKAGE_PIN C25 [get_ports {C_HS1_P[7]}];  #HS1_B1
set_property PACKAGE_PIN F23 [get_ports {C_HS1_P[6]}];  #HS1_B0
set_property PACKAGE_PIN A20 [get_ports {C_HS1_P[5]}];  #HS1_A5
set_property PACKAGE_PIN E27 [get_ports {C_HS1_P[4]}];  #HS1_A4
set_property PACKAGE_PIN C22 [get_ports {C_HS1_P[3]}];  #HS1_A3
set_property PACKAGE_PIN A23 [get_ports {C_HS1_P[2]}];  #HS1_A2
set_property PACKAGE_PIN A25 [get_ports {C_HS1_P[1]}];  #HS1_A1
set_property PACKAGE_PIN B26 [get_ports {C_HS1_P[0]}];  #HS1_A0
#C27 HS1_HSO
#B28 HS1_HSI
#D24 HS1_HSRST
#D26 HS1_HSMIO

### CRUVI HS2 ######
set_property IOSTANDARD DIFF_HSTL_I_12 [get_ports {C_HS2_P[*]}]
set_property PACKAGE_PIN C23 [get_ports {C_HS2_P[7]}]; #HS2_B5
set_property PACKAGE_PIN E22 [get_ports {C_HS2_P[6]}]; #HS2_B4
set_property PACKAGE_PIN F22 [get_ports {C_HS2_P[5]}]; #HS2_B3
```

```
# set_property PACKAGE_PIN H23 [get_ports {C_HS2_P[8]}]; #HS2_B2 not used
for loopback test
set_property PACKAGE_PIN B20 [get_ports {C_HS2_P[4]}]; #HS2_B1
set_property PACKAGE_PIN D20 [get_ports {C_HS2_P[3]}]; #HS2_A5
set_property PACKAGE_PIN D24 [get_ports {C_HS2_P[2]}]; #HS2_A4
set_property PACKAGE_PIN G21 [get_ports {C_HS2_P[1]}]; #HS2_A3
set_property PACKAGE_PIN E20 [get_ports {C_HS2_P[0]}]; #HS2_A1
#E24 HS2_HSMIO
#F25 HS2_HSO


set_property IOSTANDARD DIFF_HSTL_I_12 [get_ports {C_HS2_P[*]}]

#### ARTIX ################
set_property PACKAGE_PIN U23 [get_ports {C2C_RX_CLK}]; #U23 V_L12_P
#T24 V_L12_N
set_property PACKAGE_PIN T23 [get_ports {A_IIC_SCL_O}]; # T23 V_L13_P
set_property PACKAGE_PIN R24 [get_ports {A_IIC_SDA_I}]; # R24 V_L13_N
set_property PACKAGE_PIN R23 [get_ports {A_IIC_SDA_O}]; # R23 V_L14_P
set_property PACKAGE_PIN P24 [get_ports {C2C_TX[0]}]; #P24 V_L14_N
set_property PACKAGE_PIN M22 [get_ports {C2C_TX[1]}]; #M22 V_L15_P
set_property PACKAGE_PIN M23 [get_ports {C2C_TX[2]}]; #M23 V_L15_N
set_property PACKAGE_PIN L23 [get_ports {C2C_TX[3]}]; #L23 V_L16_P
set_property PACKAGE_PIN K24 [get_ports {C2C_TX[4]}]; #K24 V_L16_N
set_property PACKAGE_PIN K23 [get_ports {C2C_TX[5]}]; #K23 V_L17_P
set_property PACKAGE_PIN J24 [get_ports {C2C_TX[6]}]; #J24 V_L17_N
set_property PACKAGE_PIN V21 [get_ports {C2C_TX[7]}]; #V21 V_L18_P
set_property PACKAGE_PIN U22 [get_ports {C2C_TX[8]}]; #U22 V_L18_N
set_property PACKAGE_PIN T21 [get_ports {C2C_RX[0]}]; #T21 V_L19_P
set_property PACKAGE_PIN R22 [get_ports {C2C_RX[1]}]; #R22 V_L19_N
set_property PACKAGE_PIN R21 [get_ports {C2C_RX[2]}]; #R21 V_L20_P
set_property PACKAGE_PIN P22 [get_ports {C2C_RX[3]}]; #P22 V_L20_N
set_property PACKAGE_PIN N21 [get_ports {C2C_RX[4]}]; #N21 V_L21_P
set_property PACKAGE_PIN M21 [get_ports {C2C_RX[5]}]; #M21 V_L21_N
set_property PACKAGE_PIN K21 [get_ports {C2C_TX_CLK}];#K21 V_L22_P
#L22 V_L22_N
set_property PACKAGE_PIN J21 [get_ports {C2C_RX[8]}]; #J21 V_L23_P
set_property PACKAGE_PIN J22 [get_ports {C2C_RST}];   #J22 V_L23_N
set_property PACKAGE_PIN L24 [get_ports {C2C_RX[6]}]; #L24 V_L25_P
set_property PACKAGE_PIN L25 [get_ports {C2C_RX[7]}]; #L25 V_L25_N

set_property IOSTANDARD LVCMOS12 [get_ports {C2C_*}]

#N23 CLK_B702_P
#N24 CLK_B702_N

set_property IOSTANDARD LVCMOS12 [get_ports {A_IIC_*}]
```

Software Design - Vitis

For Vitis project creation, follow instructions from:

Vitis

Application

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

versal_plm

Xilinx default PLM firmware.

versal_psm

Xilinx default PSM firmware.

hello_te0950

Hello TE0950 is a Xilinx Hello World example as endless loop instead of one console output.

u-boot

U-Boot.elf is generated with PetaLinux. Vitis 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:

- Identification
  - CONFIG_SUBSYSTEM_HOSTNAME="Trenz"
  - CONFIG_SUBSYSTEM_PRODUCT="TE0950"
- Devicetree Overlays for Cameras and Artix Chip2Chip bridge (GPIO Controller)
  - CONFIG_SUBSYSTEM_EXTRA_DT_FILES="imx219-overlay.dtsi imx290-overlay.dtsi artix-overlay.dtsi ov5647-overlay.dtsi"

U-Boot

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

- read MAC from eeprom:
  - CONFIG_DM_RTC=y
    CONFIG_NVMEM=y

Fixes for BL31 (Petalinux 2023.2 Bug)

create **arm-trusted-firmware_%.bbappend** in meta-user/recipes-bsp/arm-trusted-firmware with content

meta-user/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_%.bbappend
`ATF_CONSOLE = "pl011_1"`

Device Tree

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

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

#include <dt-bindings/gpio/gpio.h>

/*------------------ eMMC --------------------*/
&sdhci0 {
        bus-width = <8>;
};

/*------------------ SD --------------------*/
&sdhci1 {
        no-1-8-v;
};

/*------------------ QSPI --------------------*/
&qspi {
        num-cs = <2>;

        flash@0 {
                compatible = "jedec,spi-nor";
                reg = <0>, <1>;
                parallel-memories = /bits/ 64 <0x8000000 0x8000000>; /*
128MB */
                spi-rx-bus-width = <4>;
                spi-tx-bus-width = <4>;
                spi-max-frequency = <40000000>; //40MHz no feedback pin

                #address-cells = <1>;
                #size-cells = <1>;
        };
};

/*------------------ ETH PHY --------------------*/
&gem0 {
        phy-handle = <&phy0>;

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

        //required otherwise petalinux gives a static address here
        /delete-property/ local-mac-address;

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

                        //only needed because of reset-gpios present
                        compatible = "ethernet-phy-id0141.0DD1"; //uboot:
[mii read 1 2].[mii read 1 3]

                        reset-names = "ETH_RESET";
                        reset-gpios = <&gpio0 23 GPIO_ACTIVE_LOW>;
                        reset-assert-us = <10000>; //minimum duration
```

```
according to datasheet 10ms
                        reset-deassert-us = <2000>;
                };
        };
};


/*------------------ GPIO MISC --------------------*/
&gpio0 {
        gpio-line-names =
                "", "", "", "", "", "", "", "", "", "",
                "", "", "", "", "", "", "", "", "", "",
                "", "", "LPD_MIO22", "";
};
&gpio1 {
        gpio-line-names =
                "", "", "", "", "", "", "", "", "", "",
                "", "", "", "", "", "", "", "", "", "",
                "", "", "", "", "", "", "", "PMC_MIO27", "", "",
                "", "", "", "", "", "", "", "USB_OC", "", "",
                "", "", "", "", "", "", "", "", "", "",
                "", "LED0", "", "", "", "", "", "", "", "";
};

/*------------------ MIPI CSI2 --------------------*/
&mipi_csi2_axi_gpio_2 {
        gpio-line-names = "CSI_GPIO0", "CSI_GPIO1";
};

&axi_gpio_2 {
        gpio-line-names = "V_PL_USR_SW", "V_USR_LED1";
};



&mipi_csi2_mipi_csi2_rx_subsystem_0 {
        status = "disabled";
        compatible = "xlnx,mipi-csi2-rx-subsystem-5.0";
};

&mipi_csi2_v_frmbuf_wr_0 {
        status = "disabled";
};

&mipi_csi2_v_proc_ss_csc {
        status = "disabled";
        compatible = "xlnx,v-vpss-csc";
};

&mipi_csi2_v_proc_ss_scaler {
        status = "disabled";
        compatible = "xlnx,v-vpss-scaler-2.2";
};

&mipi_csi_inmipi_csi2_mipi_csi2_rx_subsystem_0 {
        clock-lanes = <0>;
        data-lanes = <1 2>;
};

&mipi_csi2_v_demosaic_0 {
        status = "disabled";
```

```
        reset-gpios = <&mipi_csi2_axi_gpio_3 3 GPIO_ACTIVE_LOW>;
};


/*------------------ USB --------------------*/
&dwc3_0 {
        dr_mode = "host";
};

/*------------------ I2C --------------------*/
&i2c0 {
        i2cswitch@70 { // Artix I2C MUX Emulations
                compatible = "nxp,pca9548";
                #address-cells = <1>;
                #size-cells = <0>;
                reg = <0x70>;
                i2c-mux-idle-disconnect;

                i2c_cruvi_hs1: i2c@0 { // CRUVI HS1 IIC
                        reg = <0>;
                };
                i2c_cruvi_hs2: i2c@1 { // CRUVI HS2IIC
                        reg = <1>;
                };
                i2c_qsfp: i2c@2 { // QSFP IIC
                        reg = <2>;
                };
                i2c_fmc: i2c@3 { // FMC IIC
                        reg = <3>;
                };
        };
};

&i2c2 {
        status = "okay";

        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:

- Support for Video devices (the specific models are examplary devices that were tested)

        - CONFIG_VIDEO_DEV=y
          CONFIG_VIDEO_OV5647=y
          CONFIG_VIDEO_IMX290=y
          CONFIG_VIDEO_IMX219=y
          CONFIG_VIDEO_XILINX_TPG=y
    - Support for PWM via AXI Timer IP Core
        - CONFIG_PWM=y
          CONFIG_PWM_SYSFS=y
          CONFIG_PWM_XILINX=y

Rootfs

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

- For MIPI Camera/Video tools
    - CONFIG_yavta=y
    - CONFIG_packagegroup-petalinux-gstreamer=y
    - CONFIG_packagegroup-petalinux-v4lutils=y
- Misc Apps:
    - CONFIG_libgpiod-tools=y
    - CONFIG_mipi-example=y
    - CONFIG_startup=y
- For additional test tools:
    - CONFIG_packagegroup-petalinux-utils=y
    - CONFIG_packagegroup-petalinux-benchmarks=y
- Dropbear instead of OpenSSH
    - CONFIG_packagegroup-core-ssh-dropbear=y
- For auto login:
    - CONFIG_imagefeature-serial-autologin-root=y
    - CONFIG_imagefeature-debug-tweaks=y
    - CONFIG_imagefeature-empty-root-password=y
    - CONFIG_ADD_EXTRA_USERS="root:root;petalinux:petalinux;"

Applications

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

startup

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

cam-setup

The Versal design contains a Video Processing Pipeline for Cameras connected via the MIPI CSI-2 Interface.

cam-setup.sh is a demo application to configure the Video Pipeline it is installed into the Path, and can be called from anywhere.

The Reference Design was tested and includes drivers and devicetree overlays for the following Camera Models:

- Raspberry Pi 2.1 Camera (IMX219 Sensor)
- Raspberry Pi 1.3 Camera (OV5647 Sensor)
- Vision Components VK000435 Camera (IMX290 Sensor)

The Script can currently be used to either take a screenshot or start a MJPEG-encoded video stream via Ethernet. For all parameters call **cam-setup.sh -h**

The script cam-setup.sh can be modified to adjust resolution or other parameters.

Example

DTBO_PATH=***[path to dtbo folder, normally /run/media/[naming]-mmcblk1p1]*** cam-setup.sh -m rpi21 -o video

This stream can then be viewed e.g. by opening VLC on the network stream:

tcp://[board_ip]:5001

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.proxy241.$Proxy3496#hasContentLevelPermission. Cannot	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy241.$Proxy3496#hasContentLevelPermission. Cannot	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy241.$Proxy3496#hasContentLevelPermission. Cannot	• update design to latest

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.

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.

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]	atlassian . confluence.core. ContentEntityObject] [interface com. atlassian .user. User, class java. lang. String, class com. atlassian . confluence.core. ContentEntityObject]	atlassian . confluence.core. ContentEntityObject] [interface com. atlassian .user. User, class java. lang. String, class com. atlassian . confluence.core. ContentEntityObject]	
2024-04-10	v.53	Markus Kirberg	• fixed dual parallel QSPI access from Linux/U-boot • added small notes for cam-setup
2024-03-27	v.47	Markus Kirberg	• 2023.2 update • new assembly variants
2023-08-01	v.1	Markus Kirberg	Initial 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]	

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.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]