TE0808 StarterKit

Revision: v.27
Date: 02.04.2019 16:39

Table of Contents

Online version of this manual and other related documents can be found at https://wiki.trenz-electronic.de/display/PD/Trenz+Electronic+Documentation

Overview

Linux with basic periphery of TE0808 Starterkit (TEBF0808 Carrier).

Key Features

- TEBF0808
- Linux
- Si5345
- USB
- ETH
- PCIe
- SATA
- SD
- I2C
- RGPIO
- LED

Revision History

Date	Vivado	Project Built	Authors	Description
2017-12-18	2017.2	TE0808-StarterKit_noprebuilt-vivado_2017.2-build_07_20171219151749.zip TE0808-StarterKit-vivado_2017.2-build_07_20171219151728.zip	John Hartfiel	• initial release

Release Notes and Know Issues

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

Requirements

Software

Software	Version	Note
Vivado	2017.2	needed
SDK	2017.2	needed
PetaLinux	2017.2	needed

Hardware

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

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

Design supports following modules:

Module Model	Board Part Short Name	PCB Revision Support	DDR	QSPI Flash	Others	Notes
TE0808-ES1	es1	REV02, REV03	2GB	64MB		
TE0808-ES2	es2	REV03, REV04	2GB	64MB		
TE0808-2ES2	2es2	REV03, REV04	2GB	64MB		
TE0808-04-09EG-1EA	9eg_1ea	REV04	2GB	64MB		
TE0808-04-09EG-1EB	9eg_1eb	REV04	4GB	64MB		
TE0808-04-09EG-1ED	9eg_1eb	REV04	4GB	64MB	2,5 mm connector	
TE0808-04-09EG-1EE	9eg_1eb	REV04	4GB	128MB		
TE0808-04-09EG-1EL	9eg_1eb	REV04	4GB	128MB	2,5 mm connector	
TE0808-04-09EG-2IB	9eg_2ib	REV04	4GB	64MB		
TE0808-04-09EG-2IE	9eg_2ib	REV04	4GB	128MB		
TE0808-04-15EG-1EB	15eg_1eb	REV04	4GB	64MB		
TE0808-04-15EG-1EE	15eg_1eb	REV04	4GB	128MB		

Note: Design contains also Board Part Files for TE0808 only configuration, this boart part files are not used for this reference design.

Design supports following carriers:

Carrier Model	Notes
TEBF0808	Used as reference carrier.

Additional HW Requirements:

Additional Hardware	Notes

Content

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

Design Sources

Type	Location	Notes

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

Additional Sources

Type	Location	Notes
SI5345	<design name>/misc/Si5345	SI5345 Project with current PLL Configuration

Prebuilt

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

Download

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

Reference Design is available on:

- TE0808 StartKit

Design Flow

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

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

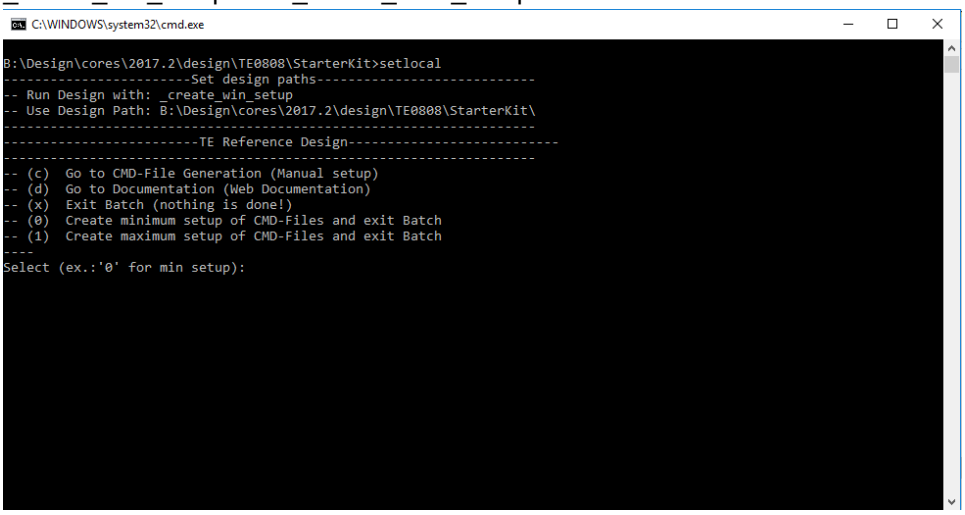
See also:

- Vivado/SDK/SDSoC#XilinxSoftware-BasicUserGuides
- Vivado Projects
- Project Delivery.

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

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

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



2. Press 0 and enter for minimum setup
3. (optional Win OS) Generate Virtual Drive or use short directory for the reference design (for example x:\<design name>)
4. Create Project
    a. Select correct device and Xilinx install path on "design_basic_settings.cmd" and create Vivado project with "vivado_create_project_guimode.cmd"
    Note: Select correct one, see TE Board Part Files
    Use Board Part Files, which ends with *_tebf0808

5. Create HDF and export to prebuilt folder
    a. Run on Vivado TCL: TE::hw_build_design -export_prebuilt
    Note: Script generate design and export files into \prebuilt\hardware\<short dir>. Use GUI is the same, except file export to prebuilt folder
6. Create Linux (bl31.elf, uboot.elf and image.ub) with exported HDF
    a. HDF is exported to "prebuilt\hardware\<short name>"
    Note: HW Export from Vivado GUI create another path as default workspace.
    b. Create Linux images on VM, see PetaLinux KICKstart
        i. Use TE Template from /os/petalinux
        Note: run init_config.sh before you start petalinux config. This will set correct temporary path variable.
7. Add Linux files (bl31.elf, uboot.elf and image.ub) to prebuilt folder
    a. "prebuilt\os\petalinux\default" or "prebuilt\os\petalinux\<short name>"
    Notes: Scripts select "prebuilt\os\petalinux\<short name>", if exist, otherwise "prebuilt\os\petalinux\default"
8. Generate Programming Files with HSI/SDK
    a. Run on Vivado TCL: TE::sw_run_hsi
    Note: Scripts generate applications and bootable files, which are defined in "sw_lib\apps_list.csv"
    b. (alternative) Start SDK with Vivado GUI or start with TE Scripts on Vivado TCL: TE::sw_run_sdk
    Note: See SDK Projects

Launch

Programming

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

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

QSPI

Not used on this Example.

SD

1. Copy image.ub and Boot.bin on SD-Card.
   - For correct prebuilt file location, see <design_name>/prebuilt/readme_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 (JTAG XMOD)
3. Select SD Card as Boot Mode
   Note: See TRM of the Carrier, which is used.
4. (Optional) Insert PCIe Card (detection depends on Linux driver. Only some basic drivers are installed)
5. (Optional) Connect Sata Disc
6. (Optional) Connect DisplayPort Monitor (List of usable Monitors: https://www.xilinx.com/support/answers/68671.html)
7. (Optional) Connect Network Cable
8. Power On PCB
   Note: 1. ZynqMP Boot ROM loads PMU Firmware and FSBL from SD into OCM, 2. FSBL loads ATF (bl31.elf) and U-boot from SD into DDR, 3. U-boot load Linux from SD into DDR.

Linux

1. Open Serial Console (e.g. putty)
    a. Speed: 115200
    b. COM Port: Win OS, see device manager, Linux OS see dmesg |grep tty (UART is *USB1)
2. Linux Console:
   Note: Wait until Linux boot finished For Linux Login use:
    a. User Name: root
    b. Password: root
3. You can use Linux shell now.
    a. I2C 0 Bus type: i2cdetect -y -r 0
    b. ETH0 works with udhcpc
    c. USB type "lsusb" or connect USB device
    d. PCIe type "lspci"

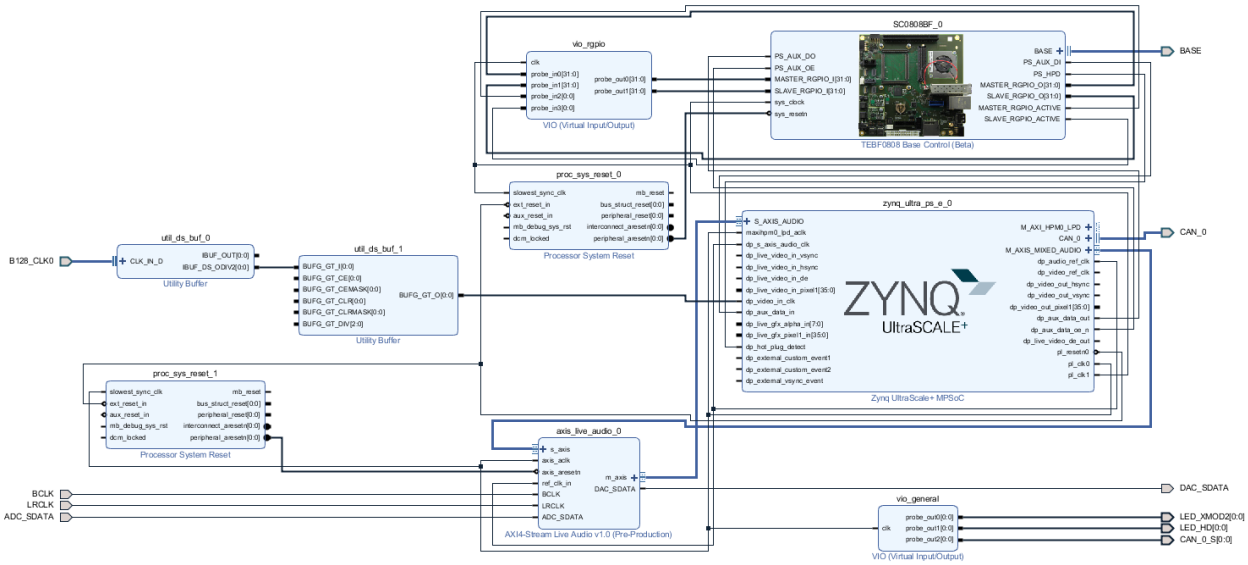
Vivado Hardware Manager

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

- RGPIO Interface:
    - Set Bit 31-28 to "1010" to activat RGPIO Interface of Master or Slave CPLD.
        - Description: TEBF0808 Master CPLD#RGPIO, TEBF0808 Slave CPLD#RGPIO
- LED Control:
    - XMOD 2(without green dot) and HD LED are accessible.

System Design - Vivado

Block Design



PS Interfaces

Activated interfaces:

Type	Note
DDR	
QSPI	MIO
SD0	MIO
SD1	MIO
CAN0	EMIO
I2C0	MIO
PJTAG0	MIO
UART0	MIO
GPIO0	MIO
TTC0	
GEM3	MIO
USB0	MIO/GTP
PCIe	MIO/GTP
SATA	GTP

Constrains

Basic module constrains

_i_bitgen.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 AH6 [get_ports {si570_clk_p[0]}]
#set_property IOSTANDARD LVDS [get_ports {si570_clk_p[0]}]
#set_property IOSTANDARD LVDS [get_ports {si570_clk_n[0]}]
#
#set_property PACKAGE_PIN G8 [get_ports {B230_CLK0_clk_p[0]}]
#set_property PACKAGE_PIN J8 [get_ports {B229_CLK1_clk_p[0]}]
#
#set_property PACKAGE_PIN F25 [get_ports {B128_CLK0_clk_p[0]}]


#LED_HD SC0 J3:31
set_property PACKAGE_PIN J14 [get_ports {LED_HD[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {LED_HD[0]}]
#LED_XMOD SC17 J3:48
set_property PACKAGE_PIN B13 [get_ports {LED_XMOD2[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {LED_XMOD2[0]}]

#System Controller IP
set_property PACKAGE_PIN A15 [get_ports base_sc10_io]
set_property PACKAGE_PIN B15 [get_ports BASE_sc11]
set_property PACKAGE_PIN C13 [get_ports BASE_sc12]
set_property PACKAGE_PIN C14 [get_ports BASE_sc13]
set_property PACKAGE_PIN E13 [get_ports BASE_sc14]
set_property PACKAGE_PIN E14 [get_ports BASE_sc15]
set_property PACKAGE_PIN G13 [get_ports BASE_sc5]
set_property PACKAGE_PIN J15 [get_ports BASE_sc6]
set_property PACKAGE_PIN K15 [get_ports BASE_sc7]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc5]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc6]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc7]
set_property IOSTANDARD LVCMOS18 [get_ports base_sc10_io]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc11]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc12]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc13]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc14]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc15]

# PLL
#set_property PACKAGE_PIN AH6 [get_ports {si570_clk_p[0]}]
#set_property IOSTANDARD LVDS [get_ports {si570_clk_p[0]}]
#set_property IOSTANDARD LVDS [get_ports {si570_clk_n[0]}]
```

```
# Clocks
#set_property PACKAGE_PIN J8 [get_ports {B229_CLK1_clk_p[0]}]
set_property PACKAGE_PIN F25 [get_ports {B128_CLK0_clk_p[0]}]
# SFP
#set_property PACKAGE_PIN G8 [get_ports {B230_CLK0_clk_p}]
# B230_RX3_P
#set_property PACKAGE_PIN A4 [get_ports {SFP1_rxp}]
# B230_TX3_P
#set_property PACKAGE_PIN A8 [get_ports {SFP1_txp}]
# B230_RX2_P
#set_property PACKAGE_PIN B2 [get_ports {SFP2_rxp}]
# B230_TX2_P
#set_property PACKAGE_PIN B6 [get_ports {SFP2_txp}]

# Audio Codec
#LRCLK          J3:49 B47_L9_N
#BCLK            J3:51 B47_L9_P
#DAC_SDATA    J3:53 B47_L7_N
#ADC_SDATA    J3:55 B47_L7_P
set_property PACKAGE_PIN G14 [get_ports LRCLK ]
set_property PACKAGE_PIN G15 [get_ports BCLK ]
set_property PACKAGE_PIN E15 [get_ports DAC_SDATA ]
set_property PACKAGE_PIN F15 [get_ports ADC_SDATA ]
set_property IOSTANDARD LVCMOS18 [get_ports LRCLK ]
set_property IOSTANDARD LVCMOS18 [get_ports BCLK ]
set_property IOSTANDARD LVCMOS18 [get_ports DAC_SDATA ]
set_property IOSTANDARD LVCMOS18 [get_ports ADC_SDATA ]

# CAN
#CAN RX SC19 J3:52 B47_L2_P
#CAN TX SC18 J3:50 B47_L2_N
#CAN S  SC16 J3:46 B47_L3_N

set_property PACKAGE_PIN A13 [get_ports CAN_0_S ]
set_property IOSTANDARD LVCMOS18 [get_ports CAN_0_S ]
set_property PACKAGE_PIN B14 [get_ports CAN_0_rx ]
set_property IOSTANDARD LVCMOS18 [get_ports CAN_0_rx ]
set_property PACKAGE_PIN A14 [get_ports CAN_0_tx ]
set_property IOSTANDARD LVCMOS18 [get_ports CAN_0_tx ]
```

Software Design - SDK/HSI

For SDK project creation, follow instructions from:

SDK Projects

Application

FSBL

TE modified 2017.2 FSBL

Changes:

- Si5345Configuration, PCIe Reset over GPIO see xfsbl_board.c and xfsbl_board.h
- Add Si5345-Registers.h, si5345.c, si5345.h

PMU

Xilinx default PMU firmware.

Hello TE0808

Hello TE0808 is a Xilinx Hello World example as endless loop instead of one console output.

U-Boot

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

Software Design - PetaLinux

For PetaLinux installation and project creation, follow instructions from:

- PetaLinux KICKstart

Config

No changes.

U-Boot

- Change platform-top.h

```
#include <configs/platform-auto.h>

/* Extra U-Boot Env settings */
#define CONFIG_EXTRA_ENV_SETTINGS \
    SERIAL_MULTI \
    CONSOLE_ARG \
    PSSERIAL0 \
    "nc=setenv stdout nc;setenv stdin nc;\0" \
    "ethaddr=00:0a:35:00:22:01\0" \
    "importbootenv=echo \"Importing environment from SD ...\"; " \
        "env import -t ${loadbootenv_addr} $filesize\0" \
    "loadbootenv=load mmc $sdbootdev:$partid ${loadbootenv_addr} ${bootenv}\0" \
    "sd_uEnvtxt_existence_test=test -e mmc $sdbootdev:$partid /uEnv.txt\0" \
    "uenvboot=" \
    "if run sd_uEnvtxt_existence_test; then" \
        "run loadbootenv" \
        "echo Loaded environment from ${bootenv};" \
        "run importbootenv; \0" \
    "sdboot=echo boot Petalinux; run uenvboot ; mmcinfo && fatload mmc 1 ${netstart}
${kernel_img} && bootm \0" \
    "autoload=no\0" \
    "clobstart=0x10000000\0" \
    "netstart=0x10000000\0" \
    "dtbnetstart=0x11800000\0" \
    "loadaddr=0x10000000\0" \
    "boot_img=BOOT.BIN\0" \
    "load_boot=tftpboot ${clobstart} ${boot_img}\0" \
    "update_boot=setenv img boot; setenv psize ${bootsize}; setenv installcmd \"install_boot\";
run load_boot ${installcmd}; setenv img; setenv psize; setenv installcmd\0" \
    "install_boot=mmcinfo && fatwrite mmc 1 ${clobstart} ${boot_img} ${filesize}\0" \
    "bootenvsize=0x40000\0" \
    "bootenvstart=0x100000\0" \
    "eraseenv=sf probe 0 && sf erase ${bootenvstart} ${bootenvsize}\0" \
    "jffs2_img=rootfs.jffs2\0" \
    "load_jffs2=tftpboot ${clobstart} ${jffs2_img}\0" \
    "update_jffs2=setenv img jffs2; setenv psize ${jffs2size}; setenv installcmd \"
install_jffs2\"; run load_jffs2 test_img; setenv img; setenv psize; setenv installcmd\0" \
```

```
    "sd_update_jffs2=echo Updating jffs2 from SD; mmcinfo && fatload mmc 1:1 ${clobstart}
${jffs2_img} && run install_jffs2\0" \
    "install_jffs2=sf probe 0 && sf erase ${jffs2start} ${jffs2size} && " \
        "sf write ${clobstart} ${jffs2start} ${filesize}\0" \
    "kernel_img=image.ub\0" \
    "load_kernel=tftpboot ${clobstart} ${kernel_img}\0" \
    "update_kernel=setenv img kernel; setenv psize ${kernelsize}; setenv installcmd \"
install_kernel\"; run load_kernel ${installcmd}; setenv img; setenv psize; setenv installcmd\0"
\
    "install_kernel=mmcinfo && fatwrite mmc 1 ${clobstart} ${kernel_img} ${filesize}\0" \
    "cp_kernel2ram=mmcinfo && fatload mmc 1 ${netstart} ${kernel_img}\0" \
    "dtb_img=system.dtb\0" \
    "load_dtb=tftpboot ${clobstart} ${dtb_img}\0" \
    "update_dtb=setenv img dtb; setenv psize ${dtbsize}; setenv installcmd \"install_dtb\"; run
load_dtb test_img; setenv img; setenv psize; setenv installcmd\0" \
    "sd_update_dtb=echo Updating dtb from SD; mmcinfo && fatload mmc 1:1 ${clobstart}
${dtb_img} && run install_dtb\0" \
    "fault=echo ${img} image size is greater than allocated place - partition ${img} is NOT
UPDATED\0" \
    "test_crc=if imi ${clobstart}; then run test_img; else echo ${img} Bad CRC - ${img} is NOT
UPDATED; fi\0" \
    "test_img=setenv var \"if test ${filesize} -gt ${psize}\\; then run fault\\; else run
${installcmd}\\; fi\"; run var; setenv var\0" \
    "netboot=tftpboot ${netstart} ${kernel_img} && bootm\0" \
    "default_bootcmd=run cp_kernel2ram && bootm ${netstart}\0" \
""
```

Device Tree

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

/* default */

/* SD */

&sdhci1 {
    // disable-wp;
    no-1-8-v;

};
/* ETH PHY */

&gem3 {
    phy-handle = <&phy0>;
    phy0: phy0@1 {
        device_type = "ethernet-phy";
        reg = <1>;
    };
};

/* QSPI */
```

```
&qspi {
    #address-cells = <1>;
    #size-cells = <0>;
    status = "okay";
    flash0: flash@0 {
        // compatible = "n25q256a";
        reg = <0x0>;
        #address-cells = <1>;
        #size-cells = <1>;
    };
};

/* I2C */

&i2c0 {
    i2cswitch@73 { // u
        compatible = "nxp,pca9548";
        #address-cells = <1>;
        #size-cells = <0>;
        reg = <0x73>;
        i2c-mux-idle-disconnect;

        i2c@2 { // PCIe
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <2>;
        };
        i2c@3 { // i2c SFP
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <3>;
        };
        i2c@4 { // i2c SFP
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <4>;
        };
        i2c@5 { // i2c EEPROM
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <5>;
        };
        i2c@6 { // i2c FMC
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <6>;

            si570_2: clock-generator3@5d {
                #clock-cells = <0>;
                compatible = "silabs,si570";
                reg = <0x5d>;
                temperature-stability = <50>;
                factory-fout = <156250000>;
                clock-frequency = <78800000>;
            };
        };
        i2c@7 { // i2c USB HUB
            #address-cells = <1>;
```

```
            #size-cells = <0>;
            reg = <7>;
        };
    };
    i2cswitch@77 { // u
        compatible = "nxp,pca9548";
        #address-cells = <1>;
        #size-cells = <0>;
        reg = <0x77>;
        i2c-mux-idle-disconnect;
        i2c@0 { // i2c PMOD
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <0>;
        };
        i2c@1 { // i2c Audio Codec
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <1>;
            /*
            adau1761: adau1761@38 {
                compatible = "adi,adau1761";
                reg = <0x38>;
            };
            */
        };
        i2c@2 { // i2c FireFly A
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <2>;
        };
        i2c@3 { // i2c FireFly B
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <3>;
        };
        i2c@4 { // i2c PLL
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <4>;
        };
        i2c@5 { // i2c SC
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <5>;
        };
        i2c@6 { // i2c
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <6>;
        };
        i2c@7 { // i2c
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <7>;
        };
    };
};
```

```
/* UNUSED DMA disable */

&lpd_dma_chan1 {
    status = "disabled";
};
&lpd_dma_chan2 {
    status = "disabled";
};
&lpd_dma_chan3 {
    status = "disabled";
};
&lpd_dma_chan4 {
    status = "disabled";
};
&lpd_dma_chan5 {
    status = "disabled";
};
&lpd_dma_chan6 {
    status = "disabled";
};
&lpd_dma_chan7 {
    status = "disabled";
};
&lpd_dma_chan8 {
    status = "disabled";
};
```

Kernel

No changes.

Rootfs

Activate:

- i2c-tools

Applications

startup

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

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

adau1761init

Audio initialisation.

Additional Software

SI5345

Download ClockBuilder Pro for SI5345

1. Install and start ClockBuilder
2. Open "/misc/SI5345/Si5345-RevB-0808-02A-Project.slabtimeproj"
3. Modify settings
4. Export Register File select C code header save to file
5. Replace Header files from FSBL template with generated file

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
2018-08-09	v.27	John Hartfiel	• Release 2017.2
	All	John Hartfiel	

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.

2018-09-18