TE0821 CR00202 Demo

Table of contents

Overview

Demonstration design for the CRUVI module board CR00202 in combination with the carrier board TEB0707-02 and the module board TE0821. The design implements a Linux example with web server application suitable for ZynqMP access via the Ethernet interface of the CRUVI module. The signals implemented in the VIO can be displayed and controlled by the Vivado HW Manager.
Wiki Resources page: https://wiki.trenz.com/te0821-info

Key Features

- Vitis/Vivado 2021.2.1
- PetaLinux
- SD
- ETH on CR00202 (J11 on TEB0707-02)
- FMeter
- Modified FSBL for SI5338 programming

Revision History

Date	Vivado	Project Built	Authors	Description
2022-12-06	2021.2.1	TE0821-CR00202_demo-vivado_2021.2-build_20_20221206113642.zip TE0821-CR00202_demo_nopre built-vivado_2021.2-build_20_20221206113642.zip	Manuela Strücker	• initial release

Design Revision History

Release Notes and Know Issues

Issues	Description	Workaround	To be fixed version
Random MAC address	MAC address for eth is random and not used by the CRUVI EEPROM	update of CPLD firmware is necessary	-

Known Issues

Requirements

Software

Software	Version	Note

Vitis	2021.2.1	needed Vivado is included into Vitis installation
PetaLinux	2021.2	needed
SI ClockBuilder Pro	---	optional

Software

Hardware

Design supports following modules:

Module Model	PCB Revision Support	Notes
CR00202-01*	REV01	NA

*used as reference

Hardware Modules

Design supports following carriers:

Carrier Model	Notes
TE0821-01-3BE21ML*	• See TE0821 Test Board
TEB0707-02*	• See TEB0707 TRM

*used as reference

Hardware Carrier

Additional HW Requirements:

Additional Hardware	Notes
USB Cable for JTAG/UART	Check Carrier Board and Programmer for correct type
XMOD Programmer	Carrier Board dependent, only if carrier has no own FTDI
Cooler	It's recommended to use cooler on ZynqMP device

*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
SI5338	<project folder>\misc\PLL\Si5338_B	SI5338 Project with current PLL Configuration
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:

- TE0821 "CR00202" Demonstration 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. Copy PetaLinux build image files to prebuilt folder

    - copy **u-boot.elf**, **system.dtb**, **image.ub, bl31.elf** 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>"

8. Generate Programming Files with Vitis

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

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

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

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

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

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

SD-Boot mode

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

2. Set Boot Mode to SD-Boot.
    - Depends on Carrier, see carrier TRM.
3. Insert SD-Card in SD-Slot.

JTAG

Not used on this Example.

Usage

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

> 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. ZynqMP Boot ROM loads PMU Firmware and FSBL from SD/QSPI Flash 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)
    a. Speed: 115200
    b. Select COM Port

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

2. Linux Console:

```
# password default disabled with 2021.2 petalinux release
petalinux login: root
Password: root
```

> Note: Wait until Linux boot finished

3. You can use Linux shell now.

```
Ethernet
        ifconfig                          (display all active interface details)
        ifconfig eth1 up     (activate the eth1 interface)
        udhcpc -i eth1       (negotiate an IP address for eth1)
```

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")

Vivado HW Manager

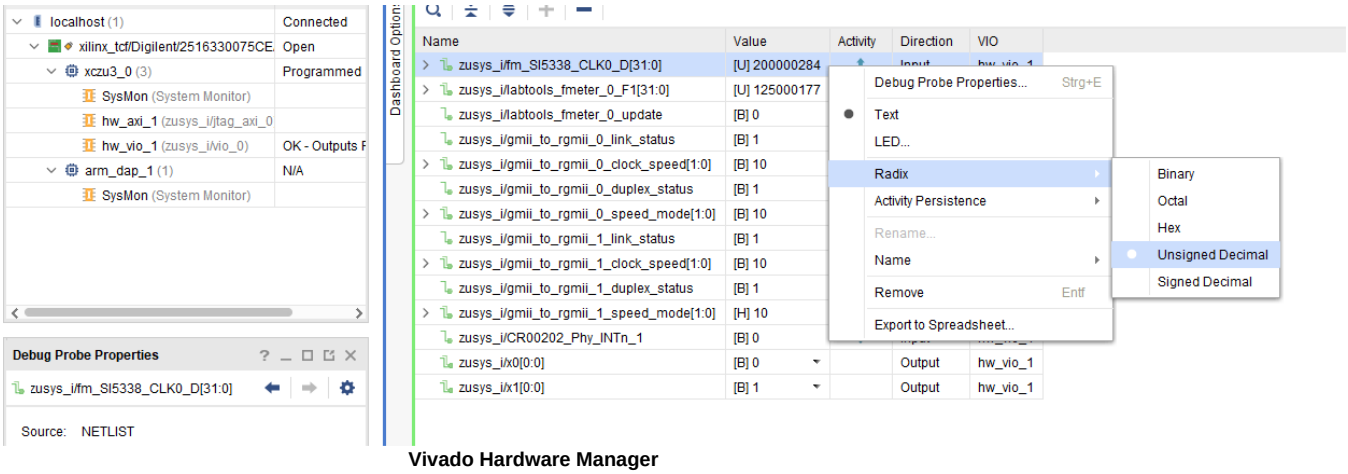
Open Vivado HW-Manager and add VIO signal to dashboard (*.ltx located on prebuilt folder)

Monitoring:

- SI5338_CLK Counter:
    - Set radix from VIO signals to unsigned integer.
      Note: Frequency Counter is inaccurate and displayed unit is Hz for CLK signals
- SI5338 CLK1 is configured to 200MHz by default and SI5338 CLK3 is configured to 125MHz by default.
- GMII_to_RGMII 0/1 output signals
    - link status
    - clock speed
    - duplex_status
    - speed_mode
- CR00202_Phy interrupt signal

Control:

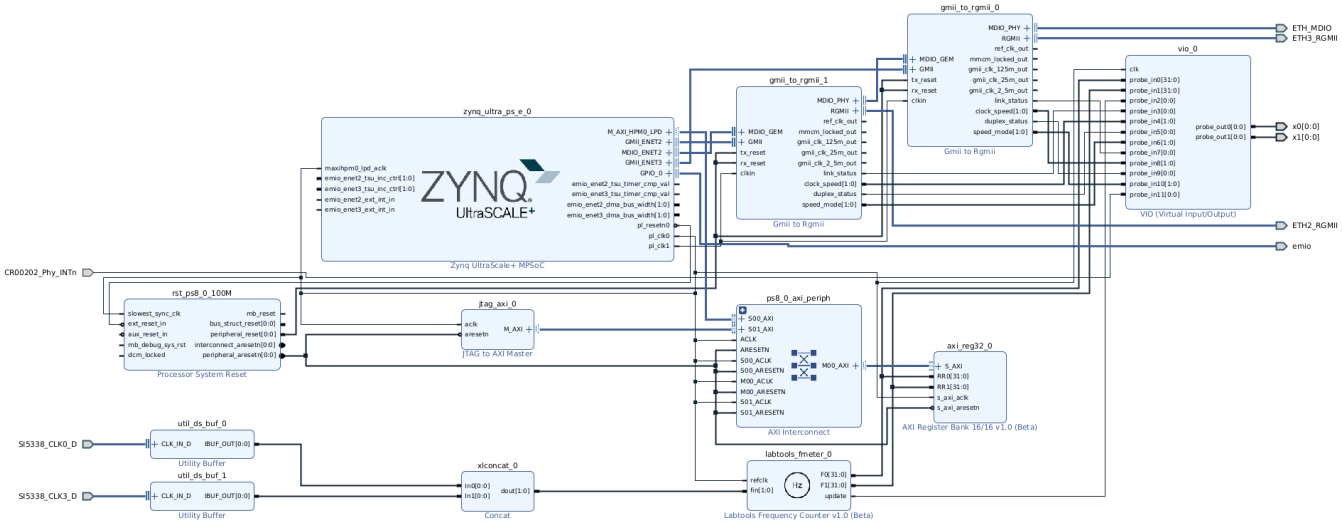
- LED over X0/X1 , see TE0821 CPLD#LED



Vivado Hardware Manager

System Design - Vivado

Block Design



Block Design

PS Interfaces

Activated interfaces:

Type	Note
DDR	
QSPI	MIO
SD0	MIO
SD1	MIO
I2C0	MIO
UART0	MIO
GPIO	EMIO (1)
GPIO0	MIO
SWDT0..1	
TTC0..3	
GEM2	EMIO
GEM3	EMIO
USB0	MIO, USB2 only

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

```
#SI5338
set_property PACKAGE_PIN E5 [get_ports {SI5338_CLK0_D_clk_p[0]}]
set_property IOSTANDARD LVDS [get_ports {SI5338_CLK0_D_clk_p[0]}]
set_property PACKAGE_PIN C3 [get_ports {SI5338_CLK3_D_clk_p[0]}]
set_property IOSTANDARD LVDS [get_ports {SI5338_CLK3_D_clk_p[0]}]


#CPLD
set_property PACKAGE_PIN B1 [get_ports {x0[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {x0[0]}]
set_property PACKAGE_PIN C1 [get_ports {x1[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {x1[0]}]


#Ethernet
#IO Placement
set_property PACKAGE_PIN R7 [get_ports {emio_tri_io[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {emio_tri_io[0]}]

set_property PACKAGE_PIN T7 [get_ports {CR00202_Phy_INTn}]
set_property IOSTANDARD LVCMOS18 [get_ports {CR00202_Phy_INTn}]

#CR00202 --> TEB0707 J11 --> TE0821
set_property PACKAGE_PIN N9 [get_ports {ETH2_RGMII_txc}]
set_property PACKAGE_PIN N8 [get_ports {ETH2_RGMII_tx_ctl}]
set_property PACKAGE_PIN M8 [get_ports {ETH2_RGMII_td[0]}]
set_property PACKAGE_PIN L8 [get_ports {ETH2_RGMII_td[1]}]
set_property PACKAGE_PIN K7 [get_ports {ETH2_RGMII_td[2]}]
set_property PACKAGE_PIN K8 [get_ports {ETH2_RGMII_td[3]}]
set_property PACKAGE_PIN K4 [get_ports {ETH2_RGMII_rxc}]
set_property PACKAGE_PIN K3 [get_ports {ETH2_RGMII_rx_ctl}]
set_property PACKAGE_PIN M6 [get_ports {ETH2_RGMII_rd[0]}]
set_property PACKAGE_PIN L5 [get_ports {ETH2_RGMII_rd[1]}]
set_property PACKAGE_PIN P7 [get_ports {ETH2_RGMII_rd[2]}]
set_property PACKAGE_PIN P6 [get_ports {ETH2_RGMII_rd[3]}]

set_property PACKAGE_PIN T8 [get_ports {ETH3_RGMII_txc}]
set_property PACKAGE_PIN R8 [get_ports {ETH3_RGMII_tx_ctl}]
set_property PACKAGE_PIN V9 [get_ports {ETH3_RGMII_td[0]}]
```

```
set_property PACKAGE_PIN U9 [get_ports {ETH3_RGMII_td[1]}]
set_property PACKAGE_PIN T6 [get_ports {ETH3_RGMII_td[2]}]
set_property PACKAGE_PIN R6 [get_ports {ETH3_RGMII_td[3]}]
set_property PACKAGE_PIN L3 [get_ports {ETH3_RGMII_rxc}]
set_property PACKAGE_PIN L2 [get_ports {ETH3_RGMII_rx_ctl}]
set_property PACKAGE_PIN L7 [get_ports {ETH3_RGMII_rd[0]}]
set_property PACKAGE_PIN L6 [get_ports {ETH3_RGMII_rd[1]}]
set_property PACKAGE_PIN V8 [get_ports {ETH3_RGMII_rd[2]}]
set_property PACKAGE_PIN U8 [get_ports {ETH3_RGMII_rd[3]}]


set_property PACKAGE_PIN Y8 [get_ports {ETH_MDIO_mdc}]
set_property PACKAGE_PIN W8 [get_ports {ETH_MDIO_mdio_io}]

set_property IOSTANDARD LVCMOS18 [get_ports {ETH2_RGMII_*}]
set_property IOSTANDARD LVCMOS18 [get_ports {ETH3_RGMII_*}]
set_property IOSTANDARD LVCMOS18 [get_ports {ETH_MDIO_*}]

set_property PULLTYPE PULLUP [get_ports {ETH2_RGMII_*}]
set_property PULLTYPE PULLUP [get_ports {ETH3_RGMII_*}]
set_property PULLTYPE PULLUP [get_ports {ETH_MDIO_*}]

#set_property slew FAST [get_ports {ETH2_RGMII_*}]
#set_property slew FAST [get_ports {ETH3_RGMII_*}]
#set_property slew FAST [get_ports {ETH_MDIO_*}]


# Clock Period Constraints
create_clock -period 8.000 -name ETH2_RGMII_rxc -add [get_ports ETH2_RGMII_rxc]
create_clock -period 8.000 -name ETH3_RGMII_rxc -add [get_ports ETH3_RGMII_rxc]

## Use these constraints to modify output delay on RGMII signals if 2ns delay is
added by external PHY
#set_output_delay -clock [get_clocks ETH2_RGMII_txc] -max -1.0 [get_ports
{ETH2_RGMII_td[*] ETH2_RGMII_txc}]
#set_output_delay -clock [get_clocks ETH2_RGMII_txc] -min -2.6 [get_ports
{ETH2_RGMII_td[*] ETH2_RGMII_txc}] -add_delay
#set_output_delay -clock [get_clocks ETH2_RGMII_txc] -clock_fall -max -1.0 [get_ports
{ETH2_RGMII_td[*] ETH2_RGMII_txc}]
#set_output_delay -clock [get_clocks ETH2_RGMII_txc] -clock_fall -min -2.6 [get_ports
{ETH2_RGMII_td[*] ETH2_RGMII_txc}]
#set_output_delay -clock [get_clocks ETH3_RGMII_txc] -max -1.0 [get_ports
{ETH3_RGMII_td[*] ETH3_RGMII_txc}]
#set_output_delay -clock [get_clocks ETH3_RGMII_txc] -min -2.6 [get_ports
{ETH3_RGMII_td[*] ETH3_RGMII_txc}] -add_delay
#set_output_delay -clock [get_clocks ETH3_RGMII_txc] -clock_fall -max -1.0 [get_ports
{ETH3_RGMII_td[*] ETH3_RGMII_txc}]
#set_output_delay -clock [get_clocks ETH3_RGMII_txc] -clock_fall -min -2.6 [get_ports
{ETH3_RGMII_td[*] ETH3_RGMII_txc}]

#clock setting
set_property UNAVAILABLE_DURING_CALIBRATION TRUE [get_ports ETH2_RGMII_td[1]]
set_property UNAVAILABLE_DURING_CALIBRATION TRUE [get_ports ETH3_RGMII_td[1]]
set_property UNAVAILABLE_DURING_CALIBRATION TRUE [get_ports ETH_MDIO_mdio_io]


##clock setting
##set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets zusys_i/gmii_to_rgmii_0/U0
/i_gmii_to_rgmii_block/rgmii_rxc_ibuf_i/O]
#set_property UNAVAILABLE_DURING_CALIBRATION TRUE [get_ports ETH2_RGMII_td[1]]
#set_property UNAVAILABLE_DURING_CALIBRATION TRUE [get_ports ETH3_RGMII_td[1]]
```

```
##False path constraints to async inputs coming directly to synchronizer
#set_false_path -to [get_pins -hier -filter {name =~ *idelayctrl_reset_gen
/*reset_sync*/PRE }]
#set_false_path -to [get_pins -of [get_cells -hier -filter { name =~ *i_MANAGEMENT
/SYNC_*/data_sync* }] -filter { name =~ *D }]
#set_false_path -to [get_pins -hier -filter {name =~ *reset_sync*/PRE }]

##False path constraints from Control Register outputs
#set_false_path -from [get_pins -hier -filter {name =~ *i_MANAGEMENT/DUPLEX_MODE_REG*
/C }]
#set_false_path -from [get_pins -hier -filter {name =~ *i_MANAGEMENT
/SPEED_SELECTION_REG*/C }]

## constraint valid if parameter C_EXTERNAL_CLOCK = 0
#set_case_analysis 0 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_clk/CE0}]
#set_case_analysis 0 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_clk/S0}]
#set_case_analysis 1 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_clk/CE1}]
#set_case_analysis 1 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_clk/S1}]

## constraint valid if parameter C_EXTERNAL_CLOCK = 0 and clock skew on TXC is
through MMCM
#set_case_analysis 0 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_90_clk/CE0}]
#set_case_analysis 0 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_90_clk/S0}]
#set_case_analysis 1 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_90_clk/CE1}]
#set_case_analysis 1 [get_pins -hier -filter {name =~ *i_bufgmux_gmii_90_clk/S1}]

##These constraints are for non-Versal devices
##To Adjust GMII Rx Input Setup/Hold Timing
#set_property IDELAY_VALUE 16 [get_cells *delay_rgmii_rx_ctl]
#set_property IDELAY_VALUE 16 [get_cells -hier -filter {name =~ *delay_rgmii_rxd*}]
#set_property IODELAY_GROUP gpr1 [get_cells *delay_rgmii_rx_ctl]
#set_property IODELAY_GROUP gpr1 [get_cells -hier -filter {name =~ *delay_rgmii_rxd*}]
#set_property IODELAY_GROUP gpr1 [get_cells *idelayctrl]

#set_property slew FAST [get_ports [list {ETH2_RGMII_td[3]} {ETH2_RGMII_td[2]}
{ETH2_RGMII_td[1]} {ETH2_RGMII_td[0]} ETH2_RGMII_txc ETH2_RGMII_tx_ctl]]
#set_property slew FAST [get_ports [list {ETH3_RGMII_td[3]} {ETH3_RGMII_td[2]}
{ETH3_RGMII_td[1]} {ETH3_RGMII_td[0]} ETH3_RGMII_txc ETH3_RGMII_tx_ctl]]

#clock setting
#set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets zusys_i/gmii_to_rgmii_0/U0
/i_gmii_to_rgmii_block/rgmii_rxc_ibuf_i/O]
```

Software Design - Vitis

For Vitis project creation, follow instructions from:

Vitis

Application

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

zynqmp_fsbl

TE modified 2021.2 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_fsbl_flash

TE modified 2021.2 FSBL

General:

- Modified Files: xfsbl_initialisation.c, xfsbl_hw.h, xfsbl_handoff.c, xfsbl_main.c
- General Changes:
    - Display FSBL Banner
    - Set FSBL Boot Mode to JTAG
    - Disable Memory initialisation

zynqmp_pmufw

Xilinx default PMU firmware.

hello_te0821

Hello TE0821 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:

- select SD default instead of eMMC:
    - CONFIG_SUBSYSTEM_PRIMARY_SD_PSU_SD_1_SELECT=y
- add new flash partition for bootscr and sizing
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART0_SIZE=0x2000000

    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART2_SIZE=0x2000000
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART3_NAME="bootscr"
    - CONFIG_SUBSYSTEM_FLASH_PSU_QSPI_0_BANKLESS_PART3_SIZE=0x80000
- select PSU_ethernet_2 instead of PSU_ethernet_3
    - CONFIG_SUBSYSTEM_ETHERNET_PSU_ETHERNET_2_SELECT=y

U-Boot

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

- MAC from eeprom together with uboot and device tree settings:
    - CONFIG_ENV_OVERWRITE=y
- Boot Modes:
    - CONFIG_QSPI_BOOT=y
    - CONFIG_SD_BOOT=y
    - # CONFIG_ENV_IS_IN_NAND is not set
    - CONFIG_BOOT_SCRIPT_OFFSET=0x4040000

Change platform-top.h:

```
#include <configs/xilinx_zynqmp.h>
#no changes
```

Device Tree

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

```
/include/ "system-conf.dtsi"
#include <dt-bindings/gpio/gpio.h>
/ {
chosen {
xlnx,eeprom = &eeprom;
};
};

/*------------------------ QSPI ------------------------ */
&qspi {
#address-cells = <1>;
#size-cells = <0>;
status = "okay";
flash0: flash@0 {
//compatible = "flash name, "micron,m25p80";
compatible = "jedec,spi-nor";
reg = <0x0>;
#address-cells = <1>;
#size-cells = <1>;
};
};


/*---------------------- SD1 sd2.0 ----------------------*/
&sdhci1 {
disable-wp;
no-1-8-v;
};
```

```
/*------------------------- ETH PHY -----------------------*/
/delete-node/ &psu_ethernet_2_mdio;
&gem2 {
phy-handle = <&phy1_gem2>;
phy-mode = "rgmii-id";
status = "okay";

reset-names = "ETH_RST";
reset-gpios = <&gpio 78 GPIO_ACTIVE_LOW>;

phy1_gem2: ethernet-phy@1 {
device_type = "ethernet-phy";
reg = <1>;
};

phy0_gem3: ethernet-phy@0 {
device_type = "ethernet-phy";
reg = <0>;
};

rgmii_1: rgmii_1@4 {
phy-handle = <&phy1_gem2>;
compatible = "xlnx,gmii-to-rgmii-1.0";
reg = <4>;
};

rgmii_0: rgmii_0@5 {
phy-handle = <&phy0_gem3>;
compatible = "xlnx,gmii-to-rgmii-1.0";
reg = <5>;
};

};

/delete-node/ &psu_ethernet_3_mdio;
&gem3 {
phy-handle = <&phy0_gem3>;
phy-mode = "rgmii-id";
status = "okay";

};


/*----------------------- USB 2.0 only --------------------*/
&dwc3_0 {
status = "okay";
dr_mode = "host";
maximum-speed = "high-speed";
/delete-property/phy-names;
/delete-property/phys;
/delete-property/snps,usb3_lpm_capable;
snps,dis_u2_susphy_quirk;
snps,dis_u3_susphy_quirk;
};

&usb0 {
status = "okay";
/delete-property/ clocks;
/delete-property/ clock-names;
clocks = <0x3 0x20>;
```

```
clock-names = "bus_clk";
};

/*---------------------------- I2C ------------------------*/
&i2c0 {
eeprom: eeprom@50 {
compatible = "microchip,24aa025", "atmel,24c02";
reg = <0x50>;
};
};
```

Kernel

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

Changes:

- Only needed to fix JTAG Debug issue:
    - CONFIG_CPU_IDLE is not set
    - CONFIG_CPU_FREQ is not set
    - CONFIG_EDAC_CORTEX_ARM64=y

Rootfs

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

Changes:

- For web server app:
    - CONFIG_busybox-httpd=y
    - CONFIG_init-ifupdown=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 web server app:
    - CONFIG_busybox-httpd=y
    - CONFIG_init-ifupdown=y
    - added folder 'recipes-core/init-ifupdown' in "<project folder>\os\petalinux\project-spec\meta-user\"

Change interfaces:

```
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

# Wireless interfaces
iface wlan0 inet dhcp
        wireless_mode managed
        wireless_essid any
        wpa-driver wext
        wpa-conf /etc/wpa_supplicant.conf

iface atml0 inet dhcp

# Wired or wireless interfaces
auto eth0
auto eth1
iface eth0 inet dhcp
iface eth1 inet dhcp

# Ethernet/RNDIS gadget (g_ether)
# ... or on host side, usbnet and random hwaddr
iface usb0 inet static
        address 192.168.7.2
        netmask 255.255.255.0
        network 192.168.7.0
        gateway 192.168.7.1

# Bluetooth networking
iface bnep0 inet dhcp
```

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

SI5338

File location "<project folder>\misc\PLL\Si5338_B\Si5338-*.slabtimeproj"

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

Appx. A: Change History and Legal Notices

Document Change History

To get content of older revision got to "Change History" of this page and select older document revision number.

Date	Document Revision	Authors	Description
Error rendering macro 'page-info' Ambiguous method overloading for method jdk.proxy279.$Proxy4022#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes	**Error rendering macro 'page-info'** Ambiguous method overloading for method jdk.proxy279.$Proxy4022#hasContentLevelPermission. Cannot resolve which method to invoke for [null, class java.lang.String, class com.atlassian.confluence.pages.Page] due to overlapping prototypes	**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	• initial release

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]	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]	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]		
	All	**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. ContentEntit yObject]	

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]