# AM0010 Test Board

Revision v.10

Exported on 2023-08-25

Online version of this document:

https://wiki.trenz-electronic.de/display/PD/AM0010+Test+Board

# 1 Table of Contents

# 2 Table of Figures

# 3 Table of Tables

# 4 Overview

Refer to http://trenz.org/am0010-info for the current online version of this manual and other available documentation.

## 4.1 Key Features

- Vitis/Vivado 2020.2
- PetaLinux
- SD(2.0)
- ETH
- USB(2.0)
- I2C
- eMMC
- FMeter
- Special FSBL for QSPI programming

## 4.2 Revision History

| Date | Vivado | Project Built | Authors | Description |
|---|---|---|---|---|
| 2021-11-19 | 2020.2 | AM0010-test_board-vivado_2020.2-build_9_20211119071538.zip<br>AM0010-test_board_noprebuilt-vivado_2020.2-build_9_20211119072230.zip | Mohsen Chamanbaz / John Hartfiel | • initial release |

**Table 1: Design Revision History**

## 4.3 Release Notes and Know Issues

| Issues | Description | Workaround | To be fixed version |
|---|---|---|---|
| No known issues | USB 3 Stick does not work on USB2 Interface, only USB 2 Stick | --- | --- |

**Table 2: Known Issues**

## 4.4 Requirements

### 4.4.1 Software

| Software | Version | Note |
|---|---|---|
| Vitis | 2020.2 | needed, Vivado is included into Vitis installation |
| PetaLinux | 2020.2 | needed |

**Table 3: Software**

### 4.4.2 Hardware

Basic description of TE Board Part Files is available on TE Board Part Files.[1]

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 |
|---|---|---|---|---|---|---|---|
| AM0010-01-3BI21FA | 3eg_1i_4gb | REV01 | 4GB | 128MB | 8GB | NA | NA |
| AM0010-01-3BI21MA* | 3eg_1i_4gb | REV01 | 4GB | 128MB | 8GB | NA | NA |
| AM0010-01-4DE21MA | 4ev_1e_4gb | REV01 | 4GB | 128MB | 8GB | NA | NA |
| AM0010-01-S001 | 4ev_1e_4gb | REV01 | 4GB | 128MB | 8GB | NA | NA |

1 https://wiki.trenz-electronic.de/display/PD/TE+Board+Part+Files

| Module Model | Board Part Short Name | PCB Revision Support | DDR | QSPI Flash | EMMC | Others | Notes |
|---|---|---|---|---|---|---|---|
| AM0010-01-S002 | 4ev_1e_4gb | REV01 | 4GB | 128MB | 8GB | NA | NA |

**Table 4: Hardware Modules**
*used as reference

Design supports following carriers:

| Carrier Model | Notes |
|---|---|
| AMB0010-01* | |

**Table 5: Hardware Carrier**
*used as reference

Additional HW Requirements:

| Additional Hardware | Notes |
|---|---|
| TE0790 (XMOD FTDI JTAG Adapter) | |
| Heat sink | |
| Mini-USB cable | |
| 12V Power supply | |
| SD card | |

**Table 6: Additional Hardware**
*used as reference

## 4.5 Content

For general structure and usage of the reference design, see Project Delivery - AMD devices[2]

### 4.5.1 Design Sources

2 https://wiki.trenz-electronic.de/display/PD/Project+Delivery+-+AMD+devices

| Type | Location | Notes |
|---|---|---|
| Vivado | <project folder>\block_design<br><project folder>\constraints<br><project folder>\ip_lib<br><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 |

**Table 7: Design sources**

### 4.5.2 Additional Sources

| Type | Location | Notes |
|---|---|---|
| init.sh | <project folder>\misc\sd\ | Additional Initialization Script for Linux |

**Table 8: Additional design sources**

### 4.5.3 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 Source | *.scr | Distro Boot file |

| File | File-Extension | Description |
|---|---|---|
| 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 |

**Table 9: Prebuilt files (only on ZIP with prebuilt content)**

### 4.5.4 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:

- AM0010 "Test Board" Reference Design[3]

3 https://shop.trenz-electronic.de/Download/?path=Trenz_Electronic/Modules_and_Module_Carriers/4x5.64/AM0010/Reference_Design/2020.2/test_board

# 5 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[4]
- Vivado Projects - TE Reference Design[5]
- Project Delivery.[6]

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[7]

⚠ **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.

4 https://wiki.trenz-electronic.de/display/PD/AMD+Development+Tools#AMDDevelopmentTools-XilinxSoftware-BasicUserGuides
5 https://wiki.trenz-electronic.de/display/PD/Vivado+Projects+-+TE+Reference+Design
6 https://wiki.trenz-electronic.de/display/PD/Project+Delivery+-+AMD+devices
7 https://wiki.trenz-electronic.de/display/PD/Project+Delivery+-+AMD+devices#ProjectDeliveryAMDdevices-Currentlylimitationsoffunctionality

- 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[8]

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 "\prebuilt\hardware\")**

```
\prebuilt\hardware\")">
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[9]
   - 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[10]
7. Copy PetaLinux build image files to prebuilt folder
   - copy **u-boot.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>"

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[11]

8 https://wiki.trenz-electronic.de/display/PD/Vivado+Board+Part+Flow
9 https://wiki.trenz-electronic.de/display/PD/PetaLinux+KICKstart
10 https://wiki.trenz-electronic.de/display/PD/Distro+Boot+with+Boot.scr
11 https://wiki.trenz-electronic.de/display/PD/Vitis

# 6 Launch

## 6.1 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[12]

### 6.1.1 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

### 6.1.2 QSPI-Boot mode

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

1. Connect **JTAG** and power on carrier with module
2. Open Vivado Project with "vivado_open_existing_project_guimode.cmd" or if not created, create with "vivado_create_project_guimode.cmd"

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

   ```
   TE::pr_program_flash -swapp u-boot
   TE::pr_program_flash -swapp hello_am0010 (optional)
   ```

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

3. Copy **image.ub** and **boot.scr** on **SD** or **USB**
   - use files from "<project folder>_binaries_<Article Name>\boot_linux" from generated binary folder,see: Get prebuilt boot binaries(see page 14)
   - or use prebuilt file location, see "<project folder>\prebuilt\file_location.txt"
4. Set Boot Mode to **QSPI-Boot** and insert **SD** or **USB**.
   - Depends on Carrier, see carrier TRM.

12 https://wiki.trenz-electronic.de/display/PD/AMD+Development+Tools#AMDDevelopmentTools-XilinxSoftwareProgrammingandDebugging

### 6.1.3 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(see page 14)
   - 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.

### 6.1.4 JTAG

Not used on this example.

## 6.2 Usage

1. Prepare HW like described on section Programming(see page 14)
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[13]

4. Power On PCB
   **boot process**

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

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

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

### 6.2.1 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:

13 https://wiki.trenz-electronic.de/display/PD/Distro+Boot+with+Boot.scr

```
petalinux login: root
Password: root
```

> Note: Wait until Linux boot finished

3. You can use Linux shell now.

```
i2cdetect -y -r 0   (check I2C 0 Bus)
i2cdetect -y -r 1   (check I2C 1 Bus)
udhcpc              (ETH0 check)
lsusb               (USB check)
```

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")

## 6.2.2 Vivado HW Manager

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

- Control: Dip switches and LEDs
- Monitoring: Output clock of SI clock buffer

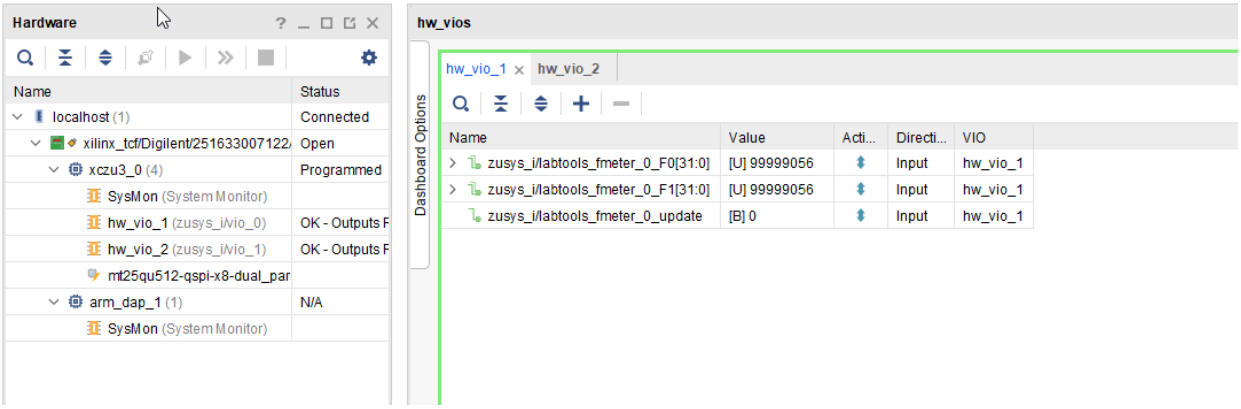


**Figure 1: Vivado Hardware Manager**

# 7 System Design - Vivado

## 7.1 Block Design

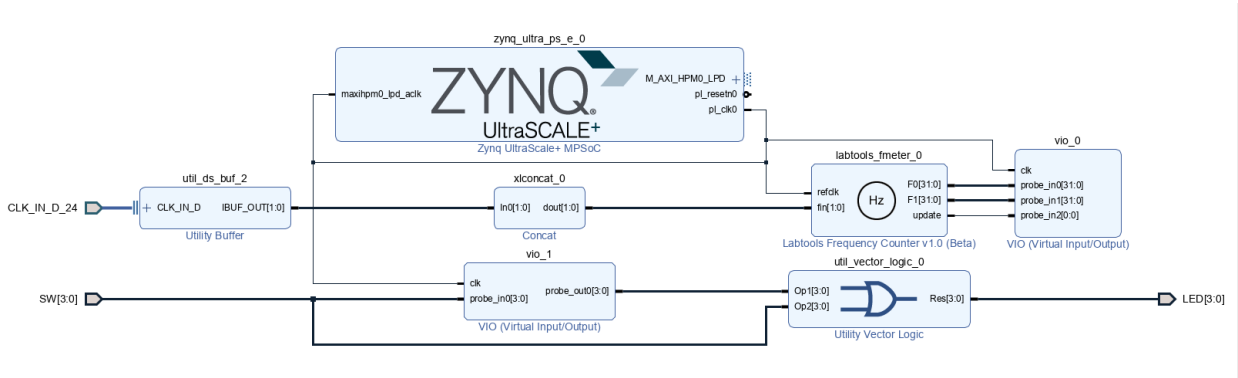


**Figure 2: Block Design**

### 7.1.1 PS Interfaces

Activated interfaces:

| Type | Note |
|---|---|
| DDR | |
| QSPI | MIO |
| SD0 (eMMC) | MIO |
| SD1 (as SD2.0) | MIO |
| I2C0 | MIO |
| I2C1 | MIO |
| UART0 | MIO |
| UART1 | MIO |
| GPIO0..2 | MIO |

| Type | Note |
|---|---|
| SWDT0..1 | |
| TTC0..3 | |
| GEM3 | MIO |
| USB0 (as USB2.0) | MIO |

**Table 10: PS Interfaces**

## 7.2 Constrains

### 7.2.1 Basic module constrains

**_i_bitgen_common.xdc**

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

### 7.2.2 Design specific constrain

**_i_io.xdc**

```
##############################################
#CLOCKs
##############################################
#   Y6       B224_CLK0_P
#   Y5       B224_CLK0_N
#   V6       B224_CLK1_P
#   V5       B224_CLK1_N
#set_property -dict { IOSTANDARD LVDS_25 PACKAGE_PIN Y6 } [get_ports
{CLK_IN_D_224_clk_p[0]}]
#set_property -dict { IOSTANDARD LVDS_25 PACKAGE_PIN V6 } [get_ports
{CLK_IN_D_224_clk_p[1]}]
#   AA13     B24_L7_P
#   AB13     B24_L7_N
#   AC14     B24_L6_P
#   AC13     B24_L6_N
set_property -dict { IOSTANDARD LVDS_25 PACKAGE_PIN AA13 } [get_ports
{CLK_IN_D_24_clk_p[0]}]
set_property -dict { IOSTANDARD LVDS_25 PACKAGE_PIN AC14 } [get_ports
{CLK_IN_D_24_clk_p[1]}]
```

```
##############################################
#LED and DIP Switch
##############################################
#   D15      USER_LED[0]
#   D14      USER_LED[1]
#   G15      USER_LED[2]
#   G14      USER_LED[3]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN D15 } [get_ports {LED[0]}]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN D14 } [get_ports {LED[1]}]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN G15 } [get_ports {LED[2]}]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN G14 } [get_ports {LED[3]}]
#   F13      USER_SW[0]
#   G13      USER_SW[1]
#   E15      USER_SW[2]
#   F15      USER_SW[3]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN F13 } [get_ports {SW[0]}]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN G13 } [get_ports {SW[1]}]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN E15 } [get_ports {SW[2]}]
set_property -dict { IOSTANDARD LVCMOS33 PACKAGE_PIN F15 } [get_ports {SW[3]}]
##############################################
#HYPERRAM
##############################################
# #CK
# set_property PACKAGE_PIN AG10 [get_ports CLK_P]
# #CKN/RFU
# set_property PACKAGE_PIN AH10 [get_ports CLK_N]
# #DQ0..7
# set_property PACKAGE_PIN AB9  [get_ports {D[0]}]
# set_property PACKAGE_PIN AC11 [get_ports {D[1]}]
# set_property PACKAGE_PIN Y10  [get_ports {D[2]}]
# set_property PACKAGE_PIN AA8  [get_ports {D[3]}]
# set_property PACKAGE_PIN Y9   [get_ports {D[4]}]
# set_property PACKAGE_PIN AD11 [get_ports {D[5]}]
# set_property PACKAGE_PIN AB10 [get_ports {D[6]}]
# set_property PACKAGE_PIN AF10 [get_ports {D[7]}]
# #RWDS/RDS
# set_property PACKAGE_PIN AA10 [get_ports RWDS]
# #CSN
# set_property PACKAGE_PIN AD10 [get_ports CS0_N ]
# #RFU
# set_property PACKAGE_PIN AE10 [get_ports CS1_N]
# #RESETN
# set_property PACKAGE_PIN AB11 [get_ports RESET_N]
# #INT
# set_property PACKAGE_PIN AA11 [get_ports INT_N ]
```

# 8 Software Design - Vitis

For Vitis project creation, follow instructions from:

Vitis[14]

## 8.1 Application

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

### 8.1.1 zynqmp_fsbl

TE modified 2020.2 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_*
    - ETH+OTG Reset over MIO
    - USB Reset over MIO
    - eMMC Reset over MIO

### 8.1.2 zynqmp_fsbl_flash

TE modified 2020.2 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

### 8.1.3 zynqmp_pmufw

Xilinx default PMU firmware.

---

General Example:

### 8.1.4 hello_am0010

Hello AM0010 is a Xilinx Hello World example as endless loop instead of one console output.

14 https://wiki.trenz-electronic.de/display/PD/Vitis

### 8.1.5 u-boot

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

# 9 Software Design - PetaLinux

For PetaLinux installation and project creation, follow instructions from:

- PetaLinux KICKstart[15]

## 9.1 Config

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

Changes:

- CONFIG_SUBSYSTEM_PRIMARY_SD_PSU_SD_1_SELECT=y
- CONFIG_SUBSYSTEM_ETHERNET_PSU_ETHERNET_3_MAC=""

## 9.2 U-Boot

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

Changes:

- CONFIG_I2C_EEPROM=y
- CONFIG_ZYNQ_GEM_I2C_MAC_OFFSET=0xFA
- CONFIG_SYS_I2C_EEPROM_ADDR=0x53
- CONFIG_SYS_I2C_EEPROM_BUS=0
- CONFIG_SYS_EEPROM_SIZE=256
- CONFIG_SYS_EEPROM_PAGE_WRITE_BITS=0
- CONFIG_SYS_EEPROM_PAGE_WRITE_DELAY_MS=0
- CONFIG_SYS_I2C_EEPROM_ADDR_LEN=1
- CONFIG_SYS_I2C_EEPROM_ADDR_OVERFLOW=0
- CONFIG_SD_BOOT=y

Change platform-top.h:

## 9.3 Device Tree

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

/*------------------ QSPI -------------------- */
&qspi {
    #address-cells = <1>;
    #size-cells = <0>;
    status = "okay";
    flash0: flash@0 {
```

15 https://wiki.trenz-electronic.de/display/PD/PetaLinux+KICKstart

```
        //compatible = "flash name, "micron,m25p80";
        compatible = "jedec,spi-nor";
        reg = <0x0>;
        #address-cells = <1>;
        #size-cells = <1>;
    };
};
/*------------------ I2C --------------------- */

&i2c0 {
    //optiga: optiga@30 {
    //    #address-cells = <1>;
    //    #size-cells = <0>;
    //    compatible = "atmel,24c08";
    //    reg = <0x30>;
    //};
    eeprom: eeprom@53 {
        #address-cells = <1>;
        #size-cells = <0>;
        compatible = "atmel,24c08";
        reg = <0x53>;
    };
    //crypto: crypto@60 {
    //    #address-cells = <1>;
    //    #size-cells = <0>;
    //    compatible = "atmel,24c08";
    //    reg = <0x60>;
    //};
};

&i2c1 {
  extern: extern@20 {
     compatible = "atmel,24c08";
     reg = <0x20>;
  };
};

/* ------------------ GEM3 ------------------- */

&gem3 {
    status = "okay";
    phy-mode = "rgmii-id";
    phy-handle = <&ethernet_phy0>;
    ethernet_phy0: ethernet-phy@0x3 {
            compatible = "marvell,88e1510";
            reg = <0x3>;
            #address-cells = <0x1>;
            #size-cells = <0x1>;
    };
};

/*------------------ USB0 ---------------------*/
/*
&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;
};
*/


/* USB  */
&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";
};


/*------------------- SD1 ------------------*/

&sdhci1 {
    pinctrl-names = "default";
    pinctrl-0 = <&pinctrl_sdhci1_default>;
    disable-wp;
    no-1-8-v;
};

&pinctrl0 {
    status = "okay";
    pinctrl_sdhci1_default: sdhci1-default {
        mux {
            groups = "sdio1_0_grp";
            function = "sdio1";
        };

        conf {
            groups = "sdio1_0_grp";
            slew-rate = <1>;
            io-standard = <1>;
            bias-disable;
        };
    /*
            mux-cd {
```

```
                groups = "sdio1_cd_0_grp";
                function = "sdio1_cd";
            };

            conf-cd {
                groups = "sdio1_cd_0_grp";
                bias-high-impedance;
                bias-pull-up;
                slew-rate = <1>;
                io-standard = <1>;
            };

            mux-wp {
                groups = "sdio1_wp_0_grp";
                function = "sdio1_wp";
            };

            conf-wp {
                groups = "sdio1_wp_0_grp";
                bias-high-impedance;
                bias-pull-up;
                slew-rate = <1>;
                io-standard = <1>;
            };
    */

    };
};

/*-------------------- SD0 eMMC ----------------*/
&sdhci0 {
    // disable-wp;
    no-1-8-v;
};
```

## 9.4 FSBL patch

Must be add manually, see template

## 9.5 Kernel

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

Changes:

- # CONFIG_CPU_IDLE is not set (Bugfix for UART and JTAG problem)
- # CONFIG_CPU_FREQ is not set (Bugfix for UART and JTAG problem)
- CONFIG_EDAC_CORTEX_ARM64=y

## 9.6 Rootfs

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

Changes:

- CONFIG_i2c-tools=y (For i2c debugging)
- CONFIG_busybox-httpd=y (for web server app)

## 9.7 Applications

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

### 9.7.1 startup

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

### 9.7.2 webfwu

Webserver application suitable for Zynq access. Need busybox-httpd

# 10 Additional Software

No additional software is needed.

# 11 App. A: Change History and Legal Notices

## 11.1 Document Change History

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

| Date | Document Revision | Authors | Description |
|---|---|---|---|
| 2021-11-19 | v.10 (see page 6) | John Hartfiel[16] | • initial release |
| -- | all | Mohsen Chamanbaz[17] , John Hartfiel[18] | -- |

**Table 11: Document change history.**

## 11.2 Legal Notices

## 11.3 Data Privacy

Please also note our data protection declaration at https://www.trenz-electronic.de/en/Data-protection-Privacy

## 11.4 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.

## 11.5 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

16 https://wiki.trenz-electronic.de/display/~j.hartfiel
17 https://wiki.trenz-electronic.de/display/~M.Chamanbaz
18 https://wiki.trenz-electronic.de/display/~j.hartfiel

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.

## 11.6 Copyright Notice

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Trenz Electronic.

## 11.7 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.

## 11.8 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.

## 11.9 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[19]. 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[20] 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)[21].

**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

19 http://guidance.echa.europa.eu/
20 https://echa.europa.eu/candidate-list-table
21 http://www.echa.europa.eu/

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.

2019-06-07