TE0813 StarterKit

Revision v.4

Exported on 2022-09-22

Online version of this document:

https://wiki.trenz-electronic.de/display/PD/TE0813+StarterKit

1 Table of Contents

2 Table of Figures

3 Table of Tables

4 Overview

Linux with basic periphery of TE0818 StarterKit (TEBF0818 Carrier).

Refer to http://trenz.org/te0813-info for the current online version of this manual and other available documentation.

4.1 Key Features

- Vitis/Vivado 2020.2
- TEBF0818
- PetaLinux
- USB
- ETH
- MAC from EEPROM
- PCIe
- SATA
- SD
- I2C
- RGPIO
- Display Port (DP)
- user LED access
- Modified FSBL for Si5338 programming/ petalinux patch
- Special FSBL for QSPI Programming

4.2 Revision History

Date	Vivado	Project Built	Authors	Description
2021-11-16	2020.2	TE0813-StarterKit_noprebuilt-vivado_2020.2-build_9_20211116073800.zip TE0813-StarterKit-vivado_2020.2-build_9_20211116073742.zip	John Hartfiel	• new variants
2021-10-28	2020.2	TE0813-StarterKit-vivado_2020.2-build_8_20211028142542.zip TE0813-StarterKit_noprebuilt-vivado_2020.2-build_8_20211028142614.zip	Manuela Strücker	• initial release

Table 1: Design Revision History

4.3 Release Notes and Know Issues

Issues	Description	Workaround/ Solution	To be fixed version
Xilinx Software	Incompatibility of board files for ZynqMP with eMMC activated between 2021.2 and 2021.2.1 patch, see Xilinx Forum Request[1]	use corresponding board files for the Vivado versions	--
QSPI Flash	Programming QSPI flash fails sometimes	use Vivado 2019.2 for programming	--

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
SI ClockBuilder Pro	---	optional

Table 3: Software

4.4.2 Hardware

Basic description of TE Board Part Files is available on TE Board Part Files.[2]

Complete List is available on "<project folder>\board_files*_board_files.csv"

Design supports following modules:

1 https://support.xilinx.com/s/feed/0D54U00005Wbon6SAB?language=en_US
2 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
TE0813-01-4BE11-A*	4eg_2gb	REV01	2GB	128MB	NA	NA	NA
TE0813-01-2AE11-A	2cg_2gb	REV01	2GB	128MB	NA	NA	NA
TE0813-01-2BE11-A	2eg_2gb	REV01	2GB	128MB	NA	NA	NA
TE0813-01-3AE11-A	3cg_2gb	REV01	2GB	128MB	NA	NA	NA
TE0813-01-4AE11-A	4cg_2gb	REV01	2GB	128MB	NA	NA	NA
TE0813-01-5DE11-A	5ev_2gb	REV01	2GB	128MB	NA	NA	NA
TE0813-01-3BE11-A	3eg_2gb	REV01	2GB	128MB	NA	NA	NA
TE0813-01-4DE11-A	4ev_2gb	REV01	2GB	128MB	NA	NA	NA

Table 4: Hardware Modules

*used as reference

Note: Design contains also Board Part Files for TE0818 only configuration, this board part files are not used for this reference design.

Design supports following carriers:

Carrier Model	Notes
TEBF0818*	Used as reference carrier.

Table 5: Hardware Carrier

*used as reference

Additional HW Requirements:

Additional Hardware	Notes
DP Monitor	Optional HW Not all monitors are supported, also Adapter to other Standard can make trouble. Design was tested with **DELL P2421**
USB Keyboard	Optional HW Can be used to get access to console which is show on DP
USB Stick	Optional HW USB was tested with USB memory stick
Sata Disk	Optional HW
PCIe Card	Optional HW
ETH cable	Optional HW Ethernet works with DHCP, but can be setup also manually
SD card	with fat32 partition

Table 6: Additional Hardware

4.5 Content

For general structure and usage of the reference design, see Project Delivery - Xilinx devices[3]

4.5.1 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

3 https://wiki.trenz-electronic.de/display/PD/Project+Delivery+-+Xilinx+devices

Type	Location	Notes
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
SI5338	<project folder>\misc\Si5338	SI5338 Project with current PLL Configuration
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
DebugProbes-File	*.ltx	Definition File for Vivado/Vivado Labtools Debugging Interface

File	File-Extension	Description
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 prebult 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:

- TE0813 "StarterKit" Reference Design[4]

4 https://shop.trenz-electronic.de/Download/?path=Trenz_Electronic/Modules_and_Module_Carriers/5.2x7.6/TE0813/Reference_Design/2020.2/StarterKit

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:

- Xilinx Development Tools[5]
- Vivado Projects - TE Reference Design[6]
- Project Delivery.[7]

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[8]

> ⚠ **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.

5 https://wiki.trenz-electronic.de/display/PD/Xilinx+Development+Tools#XilinxDevelopmentTools-XilinxSoftware-BasicUserGuides
6 https://wiki.trenz-electronic.de/display/PD/Vivado+Projects+-+TE+Reference+Design
7 https://wiki.trenz-electronic.de/display/PD/Project+Delivery+-+Xilinx+devices
8 https://wiki.trenz-electronic.de/display/PD/Project+Delivery+-+Xilinx+devices#ProjectDeliveryXilinxdevices-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[9]
> - Important: Use Board Part Files, which ends with *_tebf0818

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[10]
   - 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[11]
7. Copy PetaLinux build image files to prebuilt folder
   - copy **u-boot.elf**, **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>"

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[12]

9 https://wiki.trenz-electronic.de/display/PD/Vivado+Board+Part+Flow
10 https://wiki.trenz-electronic.de/display/PD/PetaLinux+KICKstart
11 https://wiki.trenz-electronic.de/display/PD/Distro+Boot+with+Boot.scr
12 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[13]

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_te0813 (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 15)
   - 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.
   - TEBF0818 automatically changes the boot mode to SD when the SD card is inserted. Optional CPLD firmware without boot mode change for microSD slot is available in the download area

13 https://wiki.trenz-electronic.de/display/PD/Xilinx+Development+Tools#XilinxDevelopmentTools-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 15)
   - 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 15)
2. Connect UART USB (JTAG XMOD)
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[14]

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
   boot process

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

   2. FSBL 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

6.2.1 Linux

1. Open Serial Console (e.g. putty)
   - Speed: 115200

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

- select COM Port

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

2. Linux Console:

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

> Note: Wait until Linux boot finished

3. You can use Linux shell now.

```
i2cdetect -y -r 0   (check I2C Bus)
dmesg | grep rtc    (RTC check)
udhcpc              (ETH0 check)
lsusb               (USB check)
lspci               (PCIe 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 Hardware Manager

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

- RGPIO Interface (**Important**: CPLD Firmware REV07 or newer is needed) for Control and Monitoring:
    - Set Enable to send Write date over RGPIO interface.
        - **Important use CPLD Firmware REV07 or newer:** https://wiki.trenz-electronic.de/display/PD/TEBF0818+CPLD
            - Buttons, LEDs, Status...
- Control:
    - LEDs: XMOD 2 (without green dot) and HD LED are accessible.
    - CAN_S

hw_vios

Dashboard Options

hw_vio_1 × hw_vio_2

Name	Value	Acti...	Directi...	VIO
zusys_i/RGPIO/io_rgpio_s_enable	[B] 1		Output	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_23dt12_PG[11:0]	[H] FFF		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_23dt8_unused[15:0]	[H] 0000		Output	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_11dt8_bootmode[3:0]	[H] 5		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_7dt6_ER_ERST[1:0]	[H] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_7dt0_data[7:0]	[H] 1F		Output	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_6dt5_SD_CD[1:0]	[H] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_3_unused	[B] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_2_xmod1_button	[B] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_1_S5_2_bootmode	[B] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_s_0_S5_1_bootmode	[B] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_enable	[B] 1		Output	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_23dt12_unused[11:0]	[H] 000		Output	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_23_PJTAG_SRST	[B] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_22_PJTAG_TRST	[B] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_21_FMC_CLKDIR	[B] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_20_SD_WP	[B] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_19_reserved	[B] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_18_S5_4_FMCVADJ	[B] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_17_S5_3_USER	[B] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_16_XMOD2BUTTON	[B] 1		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_15dt13_PHY_LEDS[2:0]	[H] 7		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_12_CAN_FAULT	[B] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_11dt8_muxsel[3:0]	[H] 0		Output	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_11dt8_MUX[3:0]	[H] 0		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_7dt6_unused[1:0]	[H] 0		Output	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_7dt0_data[7:0]	[H] 1F		Input	hw_vio_1
zusys_i/RGPIO/io_rgpio_m_5dt0_leds[5:0]	[H] 00		Output	hw_vio_1

hw_vios

Dashboard Options

hw_vio_1 hw_vio_2 ×

Name	Value	Acti...	Directi...	VIO
zusys_i/vio_CAN_0_S	[B] 0		Output	hw_vio_2
zusys_i/vio_LED_HD	[B] 0		Output	hw_vio_2
zusys_i/vio_LED_XMOD2	[B] 0		Output	hw_vio_2

Table 10: Vivado Hardware Manager

7 System Design - Vivado

7.1 Block Design

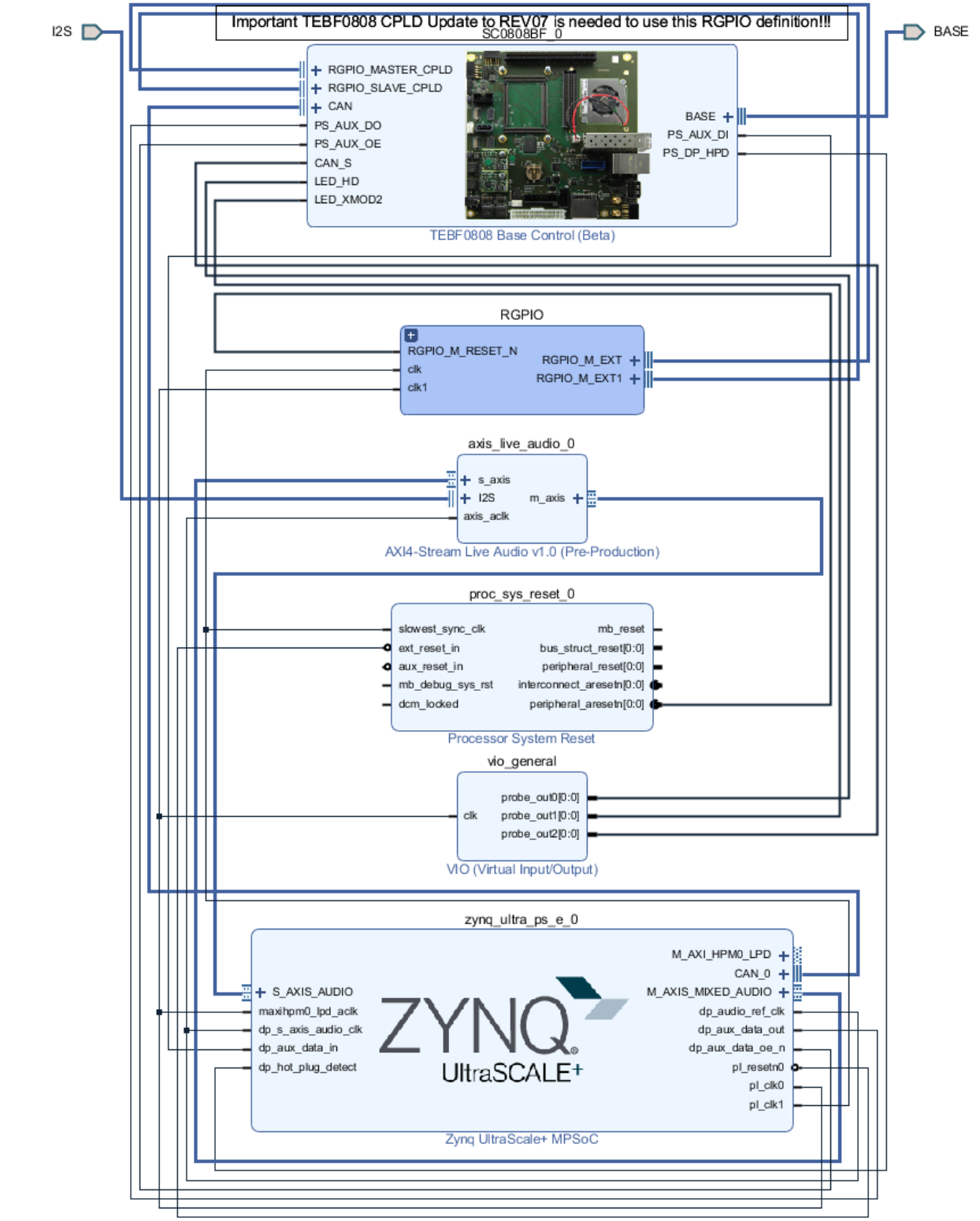

Figure 1: Block Design

7.1.1 PS Interfaces

Activated interfaces:

Type	Note
DDR	
QSPI	MIO
SD0	MIO
SD1	MIO
CAN0	EMIO
I2C0	MIO
PJTAG0	MIO
UART0	MIO
GPIO0	MIO
SWDT0..1	
TTC0..3	
GEM3	MIO
USB0	MIO/GTP
PCIe	MIO/GTP
SATA	GTP
DisplayPort	EMIO/GTP

Table 11: PS Interfaces

7.2 Constrains

7.2.1 Basic module constrains

_i_bitgen.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

```
#TEBF0818
# system controller ip
#LED_HD SC0 J3:C13
#LED_XMOD SC17 J3:B19
#CAN RX SC19 J3:B23 B26_L11_P
#CAN TX SC18 J3:B22 B26_L11_N
#CAN S SC16 J3:B18 B26_L1_N

set_property PACKAGE_PIN J14 [get_ports BASE_sc0]
set_property PACKAGE_PIN F15 [get_ports BASE_sc5]
set_property PACKAGE_PIN H13 [get_ports BASE_sc6]
set_property PACKAGE_PIN H14 [get_ports BASE_sc7]
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 A13 [get_ports BASE_sc16]
set_property PACKAGE_PIN B13 [get_ports BASE_sc17]
set_property PACKAGE_PIN A14 [get_ports BASE_sc18]
set_property PACKAGE_PIN B14 [get_ports BASE_sc19]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc0]
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]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc16]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc17]
```

```
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc18]
set_property IOSTANDARD LVCMOS18 [get_ports BASE_sc19]

# Audio Codec
#LRCLK J3:D22
#BCLK J3:D23
#DAC_SDATA J3:C21
#ADC_SDATA J3:C22
set_property PACKAGE_PIN G14 [get_ports I2S_lrclk ]
set_property PACKAGE_PIN G15 [get_ports I2S_bclk ]
set_property PACKAGE_PIN F13 [get_ports I2S_sdin ]
set_property PACKAGE_PIN G13 [get_ports I2S_sdout ]
set_property IOSTANDARD LVCMOS18 [get_ports I2S_lrclk ]
set_property IOSTANDARD LVCMOS18 [get_ports I2S_bclk ]
set_property IOSTANDARD LVCMOS18 [get_ports I2S_sdin ]
set_property IOSTANDARD LVCMOS18 [get_ports I2S_sdout ]
```

8 Software Design - Vitis

For Vitis project creation, follow instructions from:

Vitis[15]

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_*
    - Si5338 Configuration
    - OTG+PCIe Reset over MIO
    - I2C MUX for EEPROM MAC

8.1.2 zynqmp_fsbl_flash

TE modified 2019.2 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 hello_te0813

Hello TE0813 is a Xilinx Hello World example as endless loop instead of one console output.

8.1.4 u-boot

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

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

9 Software Design - PetaLinux

For PetaLinux installation and project creation, follow instructions from:

- PetaLinux KICKstart[16]

9.1 Config

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

Activate:

- 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=0x50
- CONFIG_SYS_I2C_EEPROM_BUS=2
- 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;
  };
};

/* notes:
serdes:
https://patchwork.kernel.org/project/linux-arm-kernel/patch/1536769366-31398-2-
git-send-email-anurag.kumar.vulisha@xilinx.com/
https://github.com/Xilinx/linux-xlnx/blob/master/include/dt-bindings/phy/phy.h
```

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

```
*/

/* default */

/* sata */

&sata {
phy-names = "sata-phy";
phys = <&lane2 1  0 0 150000000>;

//+phys = <PHANDLE CONTROLLER_TYPE CONTROLLER_INSTANCE LANE_REF_CLK LANE_FREQ>;
//+
//+PHANDLE                  = &lane0 or &lane1 or &lane2 or &lane3
//+CONTROLLER_TYPE          = PHY_TYPE_PCIE or PHY_TYPE_SATA or PHY_TYPE_USB
//+           or PHY_TYPE_DP or PHY_TYPE_SGMII
//+CONTROLLER_INSTANCE      = Depends on controller type used, can be any of
//+             PHY_TYPE_PCIE : 0 or 1 or 2 or 3
//+             PHY_TYPE_SATA : 0 or 1
//+             PHY_TYPE_USB  : 0 or 1
//+             PHY_TYPE_DP   : 0 or 1
//+             PHY_TYPE_SGMII: 0 or 1 or 2 or 3
//+LANE_REF_CLK             = Depends on which lane clock is used as ref clk, can
be
//+           0 or 1 or 2 or 3
//+LANE_FREQ                = Frequency of the reference clock, can be any of the
//+           below mentioned based on the phy type used
//+- PHY_TYPE_PCIE          = 100Mhz
//+- PHY_TYPE_SGMII         = 125Mhz
//+- PHY_TYPE_SATA          = 125Mhz, 150Mhz
//+- PHY_TYPE_USB           = 26Mhz, 52Mhz, 100Mhz
//+- PHY_TYPE_DP            = 27Mhz, 108Mhz, 135Mhz
 };

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

};

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

};

/* USB  */

&dwc3_0 {
    status = "okay";
    dr_mode = "host";
    snps,usb3_lpm_capable;
    snps,dis_u3_susphy_quirk;
    snps,dis_u2_susphy_quirk;
    phy-names = "usb2-phy","usb3-phy";
    phys = <&lane1 4 0 2 100000000>;
```

```
    maximum-speed = "super-speed";
};

/* 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 = "jedec,spi-nor";
        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@0 { // MCLK TEBF0818 SI5338A, 570FBB000290DG_unassembled
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <0>;
        };
        i2c@1 { // SFP TEBF0818 PCF8574DWR
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <1>;
        };
        i2c@2 { // PCIe
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <2>;
        };
        i2c@3 { // SFP1 TEBF0818
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <3>;
        };
```

```
        i2c@4 {// SFP2 TEBF0818
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <4>;
        };
        i2c@5 { // TEBF0818 EEPROM
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <5>;
            eeprom: eeprom@50 {
                compatible = "atmel,24c08";
                reg = <0x50>;
              };
        };
        i2c@6 { // TEBF0818 FMC
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <6>;
        };
        i2c@7 { // TEBF0818 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 { // TEBF0818 PMOD P1
            #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 { // TEBF0818 Firefly A
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <2>;
        };
        i2c@3 { // TEBF0818 Firefly B
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <3>;
```

```
        };
        i2c@4 { //Module PLL Si5338 or SI5345
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <4>;
        };
        i2c@5 { //TEBF0818 CPLD
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <5>;
        };
        i2c@6 { //TEBF0818 Firefly PCF8574DWR
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <6>;
        };
        i2c@7 { // TEBF0818 PMOD P3
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <7>;
        };
    };
};
```

9.4 Kernel

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

Changes:

- CONFIG_CPU_IDLE is not set (only needed to fix JTAG Debug issue)
- CONFIG_CPU_FREQ is not set (only needed to fix JTAG Debug issue)
- CONFIG_NVME_CORE=y
- CONFIG_BLK_DEV_NVME=y
- CONFIG_NVME_TARGET=y
- CONFIG_SATA_AHCI=y
- CONFIG_SATA_MOBILE_LPM_POLICY=0
- CONFIG_NVM=y
- CONFIG_NVM_PBLK=y
- CONFIG_NVM_PBLK_DEBUG=y
- CONFIG_EDAC_CORTEX_ARM64=y

9.5 Rootfs

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

Changes:

- CONFIG_i2c-tools=y
- CONFIG_busybox-httpd=y (for web server app)
- CONFIG_packagegroup-petalinux-utils=y (util-linux,cpufrequtils,bridge-utils,mtd-utils,usbutils,pciutils,canutils,i2c-tools,smartmontools,e2fsprogs)

9.6 Applications

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

9.6.1 startup

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

9.6.2 webfwu

Webserver application suitable for Zynq access. Need busybox-httpd

10 Additional Software

10.1 SI5338

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

General documentation how you work with this project will be available on Si5338[17]

17 https://wiki.trenz-electronic.de/display/PD/Si5338

11 Appx. 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
2022-09-06	v.4(see page 6)	@ Manuela Strücker[18]	• new variants
2021-10-28	v.2	Manuela Strücker	• Release 2020.2
	All	@ John Hartfiel[19], Manuela Strücker[20]	

Table 12: 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.

18 https://wiki.trenz-electronic.de/display/~m.struecker
19 https://wiki.trenz-electronic.de/display/~j.hartfiel
20 https://wiki.trenz-electronic.de/display/~m.struecker

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 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[21]. 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[22] 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)[23].

RoHS

Trenz Electronic GmbH herewith declares that all its products are developed, manufactured and distributed RoHS compliant.

WEEE

21 http://guidance.echa.europa.eu/
22 https://echa.europa.eu/candidate-list-table
23 http://www.echa.europa.eu/

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.

2019-06-07