Application Note

Akademie věd České republiky
Ústav teorie informace a automatizace AV ČR, v.v.i.

# Smart Oscilloscope Based on TEBF0808 and TE0808-04-6EB21A SoM with Analog Devices AD-FMCDAQ2-EBZ FMC Card

Lukas Kohout
kohoutl@utia.cas.cz

## Revision history

| Rev. | Date | Author | Description |
|---|---|---|---|
| 0 | 01.17.2022 | L. Kohout | Initial Draft |
| 1 | | | |
| 2 | | | |
| | | | |

# Contents



# Acknowledgement

This work has been supported from project Arrowhead Tools, project number ECSEL 826452 and MSMT 8A19009.

# 1 Introduction

This document describes a smart oscilloscope based on Trenz Electronic TEBF0808 carrier board [1] with Trenz Electronic TE0808-04-6EB21A System on Module (SoM) [2] and with Analog Devices AD-FMCDAQ2-EBZ evaluation board [3].

# 2 Used Tools and Resources

- Trenz Electronic TEBF0808 carrier board [1].
- Trenz Electronic TE0808-04-6EB21A SoM [2]. The module contains Xilinx Zynq UltraScale+ device with 4GB DDR4 memory. It is recommended to have an active cooler installed on the SoM.
- Analog Devices AD-FMCDAQ2-EBZ evaluation board [3]. Be careful during the FMC board installation, some components are too close to distance columns and they can be easily damaged. It is recommended to use the distance columns enclosed with the FMC card.

# 3 Description

The system is based on the example in HDL package provided by Analog Devices for Xilinx tools in version 2018.2. The package is downloadable from Analog Devices GitHub (https://github.com/analogdevicesinc/hdl/tree/hdl_2018_r2). The sample rate of the system is 1 GS/s. It can capture up to 939 000 000 samples that corresponds to 939 ms. If the system is configured to capture one input channel, it can store up to the maximal number of samples. But when the oscilloscope is set to capture two input channels at the same time, of course, it can store only a half of the maximum number of samples per one channel.

The original design is modified to be able to run continuously. It supports a hardware trigger and the system can pre-buffer samples before the trigger is activated. The size of this buffer is configurable; the maximal number of samples stored in this buffer is 524288. The oscilloscope is extended with the hardware accelerated signal measurement unit (SMU). Block design of the smart oscilloscope is shown in Figure 1.

## 3.1 Signal Measurement Unit

Its purpose is to perform measurements in captured samples for the power source quality evaluation. The core reduces evaluation time needed by ARM Cortex-A53 by parallel processing of data intensive operations. After configuring SMU core trigger levels, source data and destination address for measurements, it searches for periods in signal similarly to normal digital scope and evaluates the following parameters for the first found period:

- Period length in number of samples.
- Period start time. It is a pointer to memory.
- Minimal value time.
- Maximal value time.
- Mean value.
- RMS value.

As the oscilloscope can capture two input channels at the same time, the SMU can also process two channels in parallel.

One sample is a 14-bit signed integer. As the samples are stored in byte-oriented memory, the samples are represented as 2B signed integers in the memory. All captured data take 1791 MB of the memory space. SMU core is clocked at 200 MHz. The analysis computed by ARM Cortex-A53 processor takes 37.571 s. The analysis performed by SMU core takes 1.113s. It gives us a speedup 33x.

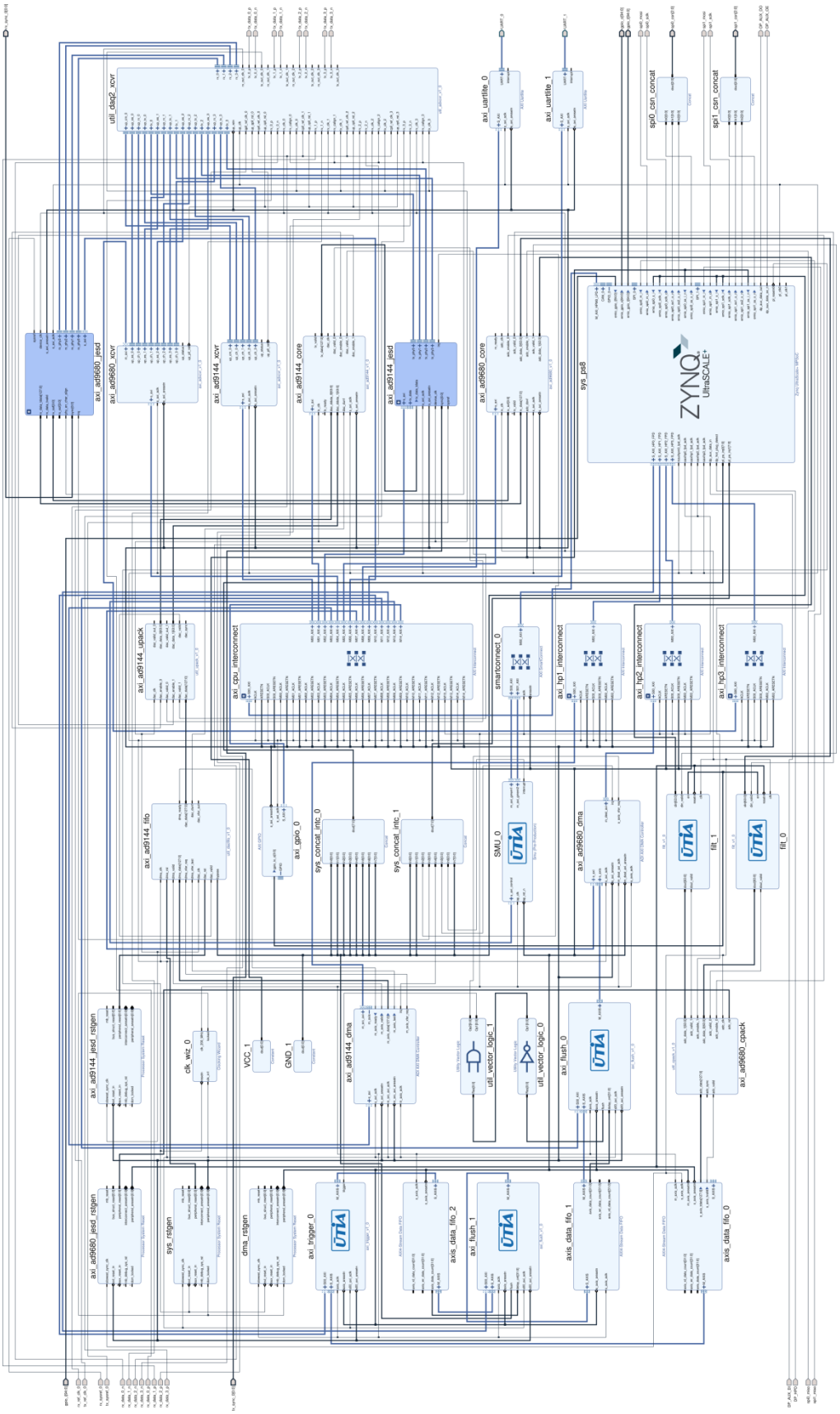


**Figure 1: Vivado 2018.2 Block Design.**

The system runs on Petalinux kernel in version 2018.2 with Debian Buster (version 10.10) file system. The smart oscilloscope is controlled by a set of software tools. All of them except one run on the system OS. The main control GUI application runs on extra PC. Next list summarizes these software applications:

- GUI application written in Python language. This application is executed on extra PC that is connected to the same Ethernet local network as the smart oscilloscope device is. It communicates with the oscilloscope device via IP sockets and it controls all the steps that are needed to capture an input signal and analyze it. It also provides the result to the user via its graphical interface.
- Main capturing SW application running on Zynq UltraScale+ device. This application controls Analog Devices AD-FMCDAQ2-EBZ card and internal data path in FPGA.
- Bridge SW application. As the name of this application indicates, this application serves as a kind of bridge between GUI application, the main capturing SW application and externally connected NUCLEO board. For the NUCLEO controller is responsible Arcelik partner from Arrowhead Tools project and the device is able to run without it.
- Calibration tool is an auxiliary tool that is used to determine the correct scale to obtain the actual sample values. The scale needs to be determined once at the beginning.
- Arrowhead provider of service. It is a tool that provides the analysis of the captured input signal to the Arrowhead framework.

The relationships between all software parts of the system can be seen in Figure 2.

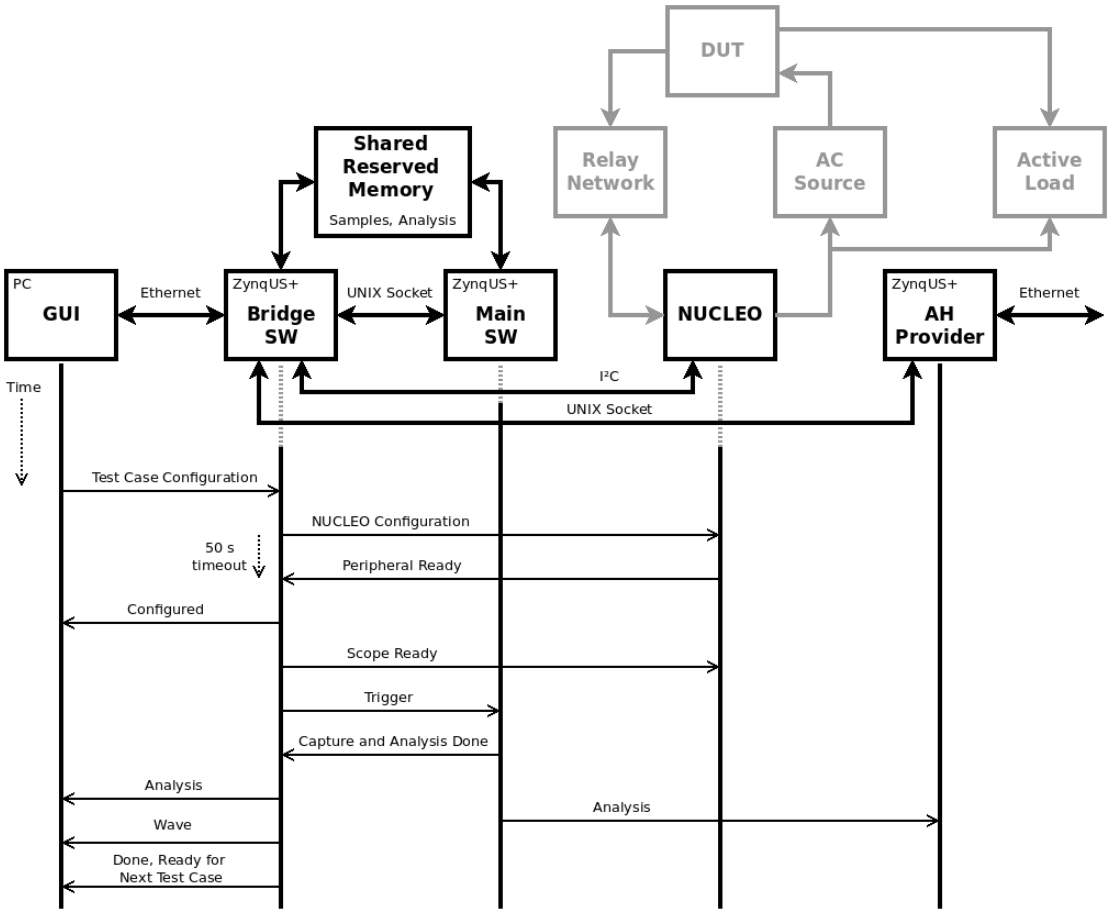


**Figure 2: Communication timeline.**

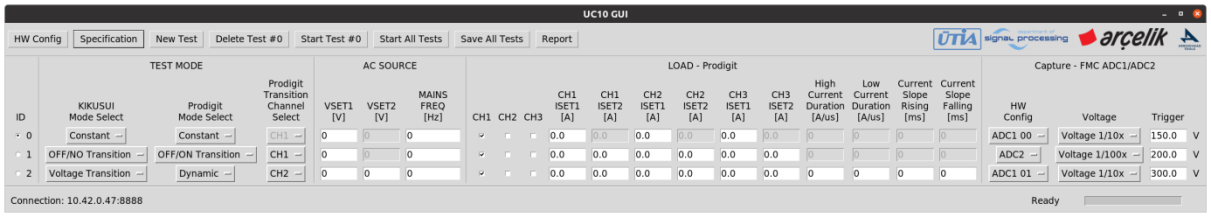


**Figure 3: GUI application – main window.**

# 4 User Guide

This section describes all software components in more details. Its purpose is to summarize all possible options of the software applications.

## 4.1 Python GUI Application

GUI application runs on extra PC. It is assumed that this PC is a member of the same local network as the smart oscilloscope device is. To communicate with the device it uses IP sockets. It allows the user to configure different test cases, send them to the oscilloscope device and display measurement results. This application is written in Python Language. It has been tested on Python 3.8.10 on Ubuntu 20.04 LTS and Python 3.8.10 running on Windows 10. The package attached with this application note includes pre-compiled executable binary files of the application:

- 64b Linux - gui_eth/lin64/gui_eth.
- 64b Windows 10 - gui_eth/win64/gui_eth.exe.

The text describes the version of the application for Linux. For Windows version it is analogous.

The GUI application should be configured before running it. Open its configuration file *gui_eth.cfg* located in folder *gui_eth/lin64* and modify lines number:

1. **Destination_IP** – It is an IP address of the oscilloscope (Zynq) device.
2. **Destination_PORT** – It is an IP port dedicated to communication with the bridge SW application. Check the bridge SW application, the port has to be the same.
3. **Timeout** – It is a timeout value in seconds defined for one IP operation (60 seconds).

To start the GUI application, execute gui_eth/lin64/gui_eth binary file. The main window of the application can be seen in Figure 3. The list bellow describes buttons of the main window:

- **HW Config** – clicking on this button opens a hardware configuration table (Figure 5). In this table, it is possible to set a name of the hardware configuration. This label is used as the identifier of the hardware setup processed by the NUCLEO board and it is also shared within the GUI application components as the reference.
    - **Save** – it saves current table to *hw_config.csv* file. This file is authomaticaly loaded on the application startup. If the file is missing, it will be created.
    - **Close** – it closes the table without saving.
- **Specification** – this button opens a specification table, where the user can define expected values of the measurement and the tolerance of these values. Each hardware configuration has its own specification in the table (see Figure 4).
    - **Save** – it saves the current table to *specification.csv* file. This file is authomaticaly loaded on the application startup. If the file is missing, it will be created.
    - **Close** – it closes the table without saving.

- **New Test** – it adds a new test case to the list of test cases on the main window.
- **Delete Test #n** – it deletes a currently selected test case. The selection is done by clicking on the test case ID on the left side of the main window. The selected test case is also directly indicated on the button label.
- **Start Test #n** – it starts a currently selected test case. The selection is done by clicking on the test case ID on the left side of the main window. The selected test case is also directly indicated on the button label. After the test is finished, a report window will be opened.
- **Start All Tests** – it starts all tests from the list on the main window one by one. After all tests are finished, a report window will be opened.

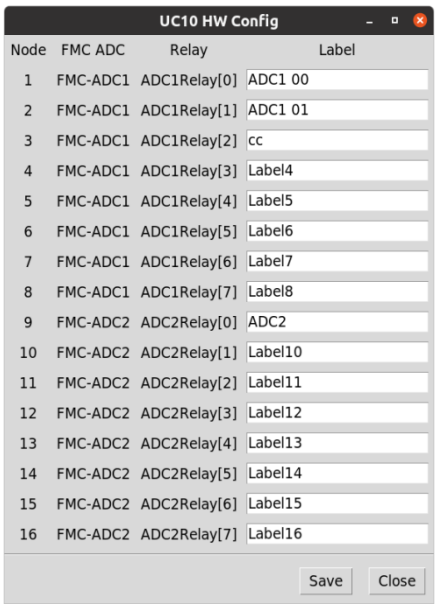


**Figure 5: GUI application – hardware configuration table.**

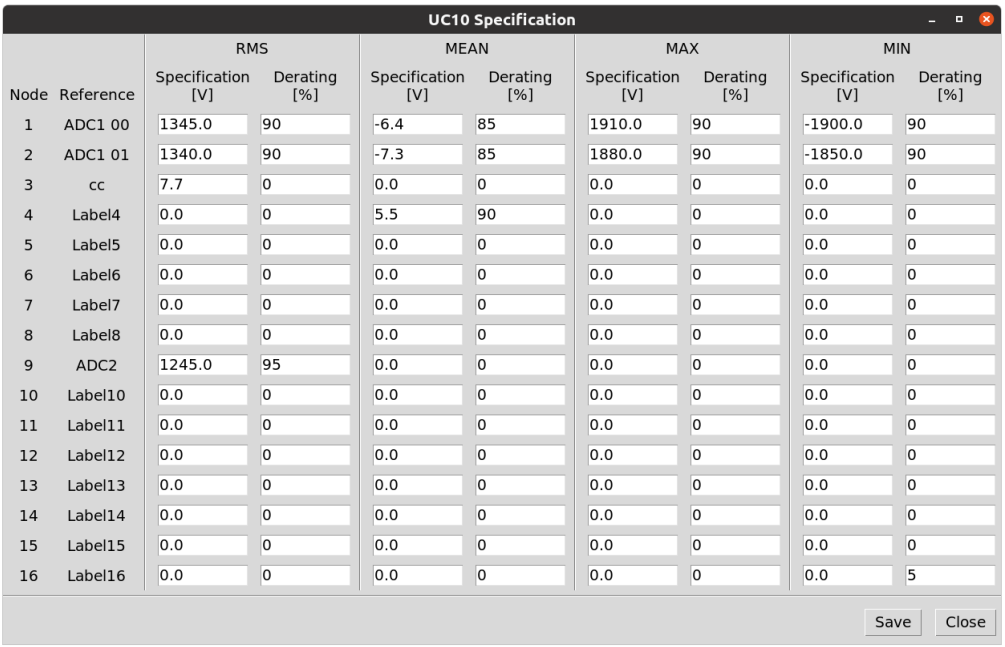


**Figure 4: GUI application – specification table.**

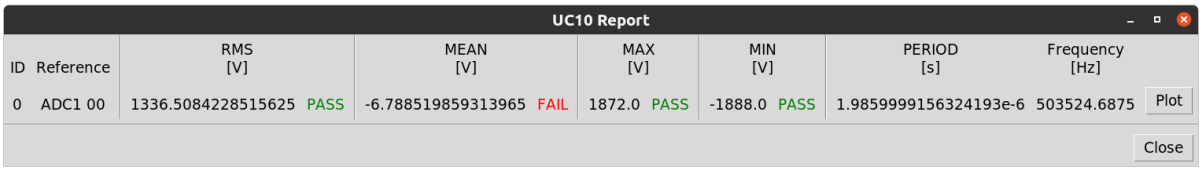


**Figure 6: GUI application – report window.**

- **Save All Tests** – it saves all tests from the list on the main window to *test_case.csv* file. This file is authomaticaly loaded on the application startup. If the file is missing, it will be created.
- **Report** – it opens a report window with the measurement results and PASS/FAIL decision made based on the specification table. Figure 6 shows an example of the report window for one test. In case of starting all tests, this window will contain report of all tests.
    - **Plot** – it plots the waveform of the captured signal during the test with measured values (Figure 7). Cursors in the plot window are interactive.
        - **X1** – select the first X axis cursor. Click to change its position.
        - **X2** – select the second X axis cursor. Click to change its position.
        - **Y1** – select the first Y axis cursor. Click to change its position.
        - **Y2** – select the second X axis cursor. Click to change its position.
        - **Reset** – it resets cursors to pre-calculated values.
        - **Close** – it closes the plot window.

      The wave within the window can be moved, zoomed, or saved as the image (PNG, JPG, …). After the test is finished, the waveform data are stored in a file with extension *rb*. The file contents raw binary data in form of 2B signed

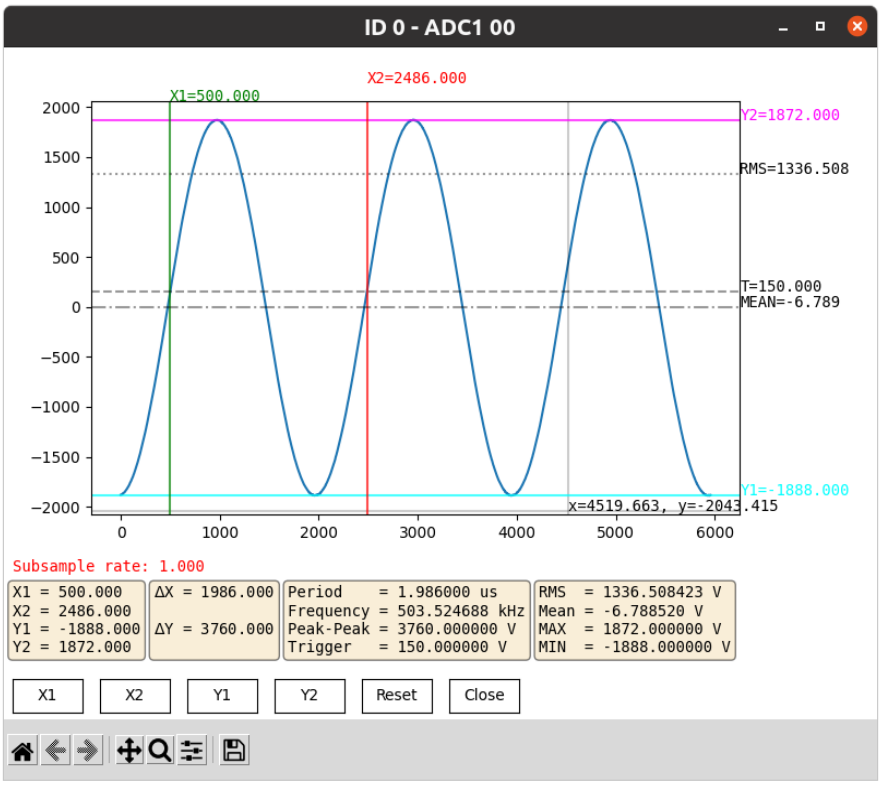


**Figure 7: GUI application – report – plot waveform.**

integers representing samples. When the test case is run again, the content of the file will be overwritten. Due to performance, the number of samples displayed in the plot window is limited. If the number of samples stored in the wave file exceeds 166666 samples, the wave will be downsampled.

- **Close** – it closes the report window. When the plot window is opened it will be closed as well.

## 4.2 Main Capturing Application

The main SW application handles the FMC card. It captures data from ADCs to the shared reserved memory space located in the DDR3 memory. It performs an analysis of the captured data. The result can be accessed by another application to perform any other post processing operations; the SW bridge, for instance. To provide this, it communicates via UNIX socket. Another UNIX socket is dedicated to send the analysis to the Arrowhead provider of service. The application is parameterizable from the command line on its startup. The application is located in the */root/smart-osc* folder of the Zynq device. To print help, execute the following commands from the Zynq device terminal:

```
cd /root/smart-osc
./sw.elf -h
Usage: ./sw.elf [-t NO_OF_US] [-T] [-c NO_OF_CHANNELS] [-s SCALE_DIV_1] [-S
SCALE_DIV_2] [-l ITERATIONS] [-y YRANGE_MIN -Y YRANGE_MAX] [-g] [-G] [-C
CAP_CHAN]
  -t NO_OF_US : Number of us to be captured per one channel
                1 channel  : 1-939000 (1 - default)
                2 channels : 1-469000 (1 - default)
                1 us ~ 1000 samples (1GBS/s)
  -T : Wait for external trigger
  -c NO_OF_CHANNELS : Set number of channels [1-2] (2 default)
  -s SCALE_DIV_1 : floating point divider for channel 1, default 1.0
  -S SCALE_DIV_2 : floating point divider for channel 2, default 1.0. If
                   SCALE_DIV_2 is not defined, SCALE_DIV_1 will be used for
                   channel 2
  -l ITERATIONS : 0 = infinite loop (default), 1-xxx
  -y YRANGE_MIN : Set minimum of the Y range, -Y YRANGE_MAX also required
  -Y YRANGE_MAX : Set maximum of the Y range, -y YRANGE_MIN also required
  -C CAP_CHAN : Select the channel to be analyzed [1-4] (4 default). 1 -
                CH1, 2 - CH2, 3 - CH1 and CH2 at the same time,
                4 - CH1 or CH2
                For choice 2, 3 or 4 parameter -c NO_OF_CHANNELS has to be
                set to 2
  -g : Generate file for GnuPlot
  -G : Execute GnuPlot
  -h : Print this help
```

Table 1 shows detailed input arguments description.

**Table 1: Main capturing application arguments.**

| Option | Value | Description |
|---|---|---|
| -t | unsigned integer | Number of microseconds to be captured per one channel<br>1 channel : 1-939000 (1 - default)<br>2 channels at the same time: 1-469000 (1 - default)<br>1 µs ~ 1000 samples (1GBS/s) |
| -T | | Enables waiting for an external trigger delivered by UINIX socket from the GUI application. It automatically sets argument –C to 4. |

**Table 1: Main capturing application arguments.**

| Option | Value | Description |
|---|---|---|
| -c | unsigned integer | Number of channels [1- 2] (2 default) |
| -s | float | Scale divider of captured samples, each sample is divided by this value (1.0 default). By default, this value is used primarily for channel 1 or for both input channels when argument -S SCALE_DIV_2 is not set. |
| -S | float | Scale divider of captured samples for channel 2, each sample is divided by this value (1.0 default). In case this argument is not set, scale divider for channel 1 is used also for channel 2 |
| -I | unsigned integer | Number of iterations. 0 means infinite loop, otherwise 1-$2^{32}$ (0 - default) |
| -y | signed integer | Set minimum of the Y range of the GnuPlot tool. This option also requires *-g -G -Y YRANGE_MAX* options to be enabled. |
| -Y | signed integer | Set maximum of the Y range of the GnuPlot tool. This option also requires *-g -G -Y YRANGE_MIN* options to be enabled. |
| -C | unsigned integer | Selects the channel to be captured [1-4]. 1 – store only channel 1 (default), 2 – store only channel 2, 3 – store both channels at the same time, 4 – store channel 1 or channel 2. In case of selecting channel number 2, 3 or 4, argument -c NO_OF_CHANNELS has to be set to 2. Channel 4 is also automatically set by –T argument. |
| -g | | Write captured samples to the file *plot.txt* in ASCII. This file is processed by the GnuPlot tool. The file remains in the file system after the application is stopped; it contains the data of the latest capturing iteration. |
| -G | | Execute the GnuPlot tool internally. Option *-g* is *required* |
| -h | | Print help |

### 4.2.1 Examples

The main capturing application is located in the */mnt* folder on the Zynq device.

**Start all in default**

```
./sw.elf
```

**Set 20 microseconds to be captured**

```
./sw.elf -t 20
```

**Enable generating of the file with samples**

```
./sw.elf -t 20 -g
```

**Start with GnuPlot and set Y range [-2, 3]**

```
./sw.elf -t 20 -g -G -y -2 -Y 3
```

**Start with GnuPlot, set Y range [-2, 3] and display channel 2**

```
./sw.elf -t 20 -g -G -y -2 -Y 3 -C 2
```

**Start for UC10 demo**

```
./sw.elf – t 100000 –s 1.0 -T
```

**Note**

Writing *plot.txt* file to the SD card is slow. The size of the file depends on a number of microseconds to be captured

## 4.3 Bridge Application

The bridge application serves as a bridge between GUI, the main capturing application and the NUCLEO board. The application uses four different interfaces. GUI and the bridge application communicate via Ethernet. This communication is based on IP sockets. To communicate with the NUCLEO board, $I^2C$ interface is utilized. To send commands and receive response from the main capturing application, the bridge uses UNIX sockets. The bridge also accesses a shared reserved memory to obtain the results of analysis of the captured data by the main capturing SW. The application is located in the */root/smart-osc* folder of the Zynq device. To print help, execute the following commands from the Zynq device terminal:

```
cd /root/smart-osc
./bridge_sw.elf -h
Usage: ./bridge_sw.elf [-s I2C_SPEED] [-a I2C_ADDRESS] [-p IP_PORT] [-f] [-
h]
  -s I2C_SPEED - in kHz, valid values 100 (default) and 400
  -a I2C_ADDRESS - 1 byte in HEX, default value: 0B
  -p IP_PORT - set IP port, default is 8888
  -f force ACK on I2C failure for debug purpose
  -S Skip socket reading for debug purpose
  -h Print this help
```

All parameters are described in Table 2.

**Table 2: Bridge application input arguments.**

| Option | Value | Description |
|---|---|---|
| -s | unsigned integer | Set I2C clock in kHz [100, 400], 100 is default |
| -a | 2-char HEX | 7-bit I2C address, 0B is default |
| -p | unsigned integer | IP port, default is 8888 |
| -f | | Force acknowledge to the GUI application on I2C failure. This option is for debug purpose. It can be used when NUCLEO board is not connected. |
| -S | | Skip reading of the UNIX socket from the main SW application. This option is for debug purpose. |
| -h | | Print help |

### 4.3.1 Examples

The bridge application is located in the */mnt* folder on the Zynq device.

**Start in default (UC10 demo)**

```
./bridge_sw.elf
```

**Set I2C clock to 400 kHz**

```
./bridge_sw.elf -s 400
```

**Set 7-bit I2C address to 0x16 (0x2C for reading, 0x2D for writing)**

```
./bridge_sw -a 16
```

**Set full debug mode**

```
./bridge_sw.elf -f -S
```

## 4.4 Calibration Tool

AD-FMCDAQ2-EBZ FMC card uses AD9680-1000 analog to digital converter, which accepts input signal voltage in range 1.70 V peak to peak. The input pin is equipped with a transformer that adjusts the input signal voltage level to the given range. Therefore, it is necessary to obtain a conversion constant that allows the sampled values to be converted to the actual values. This value is then used as an input argument of the main capturing application. Examine *-s SCALE_DIV_1* and *-S SCALE_DIV_2* arguments in Section 4.2. The calibration tool is an auxiliary tool and the scale needs to be determined only once at the beginning. To get the scaling constant, set the input signal voltage level to a known value. 1 V, for instance. As the tool searches for the maximum value in the input signal, the shape of the input signal should be a constant value or sinus. The application is located in the */root/smart-osc* folder of the Zynq device. To start the calibration tool, execute the following commands from the Zynq device terminal:

```
cd /root/smart-osc
./calib.elf
 ...
+ increment | - decrement | v val | s stop | any key to repeat
```

The tool prints the maximum value of the captured input signal and divides this value by the scaling constant. The constant starts at value 1.0. The value can be incremented, decremented or set directly. The increase/decrease step is equal to 0.01. When the value is changed, the tool captures the new data, obtains the maximum value and divides the maximum by the new scaling constant. Continue setting the scale until the maximum value is equal 1 V.

## 4.5 Arrowhead Provider of Service

The Arrowhead provider of service is a tool that provides the analysis of the captured input signal to the Arrowhead framework in version 4.1.3. It reads the UNIX socket containing the analysis of the latest measurement. This UNIX socket is written by the main capturing application. The precompiled application is located in the */root/smart-osc/provider* folder of the Zynq device. To run the provider application, execute the following commands from the Zynq device terminal:

```
cd /root/smart-osc/provider
./ProviderExample --secureArrowheadInterface --secureProviderInterface
```

The full source code of the provider application is located in */root/ ProviderUTIASecure*. If the provider certificates and password are changed, it is necessary to recompile it. The details about the Arrowhead provider of service compilation procedure can be found in the application note called *Arrowhead 4.1.3 Client on Trenz TEBF0808 + TE0808 04 6EB21A SoM Running Petalinux 2018.2 Kernel with Debian Buster File System* [4].

To install the Arrowhead framework, follow the application note called *Arrowhead Core System on Ubuntu 18.04* [5].

# 5 Smart Oscilloscope Installation

In this section it is assumed that the hardware components TEBF0808 carrier board, TE0808-04-6EB21A SoM and AD-FMCDAQ2-EBZ are already assembled. The application note package contains an image of the SD card for the Zynq device (examine Section 6). The image includes all required components:

- *BOOT.bin* file. It contains the FPGA bitstream, the first stage bootloader and u-boot.
- Petalinux 2018.2 kernel (*image.ub* file).
- Debian Buster 10.10 file system.
- Precompiled smart oscilloscope software components. These are the main capturing application, bridge application, calibration tool and Arrowhead provider of the service.

To write the image to the SD card, follow the steps described in the application note *Trenz TEBF0808 + TE0808-04-6EB21A SoM Running Petalinux 2018.2 Kernel with Debian Buster File System* [6].

# 6 Package Content

```
.
├── gui_eth
│   ├── lin64
│   │   ├── ...
│   │   ├── gui_eth
│   │   ├── gui_eth.cfg
│   │   ├── hw_config.csv
│   │   ├── logo_aht.png
│   │   ├── logo_arcelik.png
│   │   ├── logo_utia.png
│   │   ├── specification.csv
│   │   ├── test_case.csv
│   │   └── ...
│   └── win64
│       ├── ...
│       ├── gui_eth.cfg
│       ├── gui_eth.exe
│       ├── hw_config.csv
│       ├── logo_aht.png
│       ├── logo_arcelik.png
│       ├── logo_utia.png
│       ├── specification.csv
│       ├── test_case.csv
│       └── ...
├── sd
│   └── aht-demo-2022-01-21.zip
└── uc10-appnote_v2.pdf
```

# 7 References


[1] Trenz Electronic, UltraITX+ Baseboard for Trenz Electronic TE080X UltraSOM+, [Online]. Available: https://shop.trenz-electronic.de/en/TEBF0808-04A-UltraITX-Baseboard-for-Trenz-Electronic-TE080X-UltraSOM.

[2] Trenz Electronic, UltraSOM+ MPSoC Module with Zynq UltraScale+ XCZU6EG-1FFVC900E, 4 GB DDR4, [Online]. Available: https://shop.trenz-electronic.de/en/TE0808-05-6BE21-A-UltraSOM-MPSoC-Module-with-Zynq-UltraScale-XCZU6EG-1FFVC900E-4-GB-DDR4.

[3] Analog Devices, AD-FMCDAQ2-EBZ Evaluation Board, [Online]. Available: https://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/eval-ad-fmcdaq2-ebz.html#eb-overview.

[4] L. Kohout, Arrowhead 4.1.3 Client on Trenz TEBF0808 + TE0808 04 6EB21A SoM Running Petalinux 2018.2 Kernel with Debian Buster File System, UTIA AV ČR, v.v.i., [Online]. Available: http://sp.utia.cz/index.php?ids=results&id=ah-cli-deb.

[5] L. Kohout, Arrowhead Core System on Ubuntu 18.04, UTIA AV ČR, v.v.i., [Online]. Available: http://sp.utia.cz/index.php?ids=results&id=aht-cs-on-ubuntu18-04.

[6] L. Kohout, Trenz TEBF0808 + TE0808-04-6EB21A SoM Running Petalinux 2018.2 Kernel with Debian Buster File System, UTIA AV ČR, v.v.i., [Online]. Available: http://sp.utia.cz/index.php?ids=results&id=te0808-debian-buster.

## Disclaimer

This disclaimer is not a license and does not grant any rights to the materials distributed herewith. Except as otherwise provided in a valid license issued to you by UTIA AV CR v.v.i., and to the maximum extent permitted by applicable law:

(1) THIS APPLICATION NOTE AND RELATED MATERIALS LISTED IN THIS PACKAGE CONTENT ARE MADE AVAILABLE "AS IS" AND WITH ALL FAULTS, AND UTIA AV CR V.V.I. HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and

(2) UTIA AV CR v.v.i. shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under or in connection with these materials, including for any direct, or any indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or UTIA AV CR v.v.i. had been advised of the possibility of the same.

Critical Applications:
UTIA AV CR v.v.i. products are not designed or intended to be fail-safe, or for use in any application requiring fail-safe performance, such as life-support or safety devices or systems, Class III medical devices, nuclear facilities, applications related to the deployment of airbags, or any other applications that could lead to death, personal injury, or severe property or environmental damage (individually and collectively, "Critical Applications"). Customer assumes the sole risk and liability of any use of UTIA AV CR v.v.i. products in Critical Applications, subject only to applicable laws and regulations governing limitations on product liability.