TPQA-Traceable Power & Power Quality Analyzer

GUIDE FOR SAMPLING POWER AND POWER QUALITY MEASUREMENTS

USER'S GUIDE FOR THE TPQA OPEN SOFTWARE TOOL

Bruno Trinchera

May/2019

Good practice guide
Chapter 3 – User's guide for the TPQA open software tool

BLANK PAGE

OVERVIEW

This document produces a user's guide for the TPQA open software tool which will help end-users to acquaint themselves with the tool. The software is described in terms of the user interface and the configuration of the digitizers. The guide gives general information how to integrate new digitizers using the concept of the virtual driver and the flow chart for its implementation. Further details about power and PQ tests included in the initial database of calculation algorithms and about the implementation of new algorithms are described in the similar guide produced for TWM open software tool.

This guide also covers at least partially following reports and activities:

- A2.1.1 Flow chart for the use of the modular system in LabWindows/CVI;
- A2.1.4 Concept for the interface between LabWindows/CVI and Matlab;
- A2.2.2 Integration of instrument drivers to create a virtual generic digitizer for LabWindows/CVI;
- A2.2.4 Developing, testing and providing control and acquisition module as unique GUI interface for Labwindows/CVI;
- A2.2.5 Testing of control and data acquisition of TPQA as an executable file on existing measurement setups;
- A2.3.6 Processing module description of TPQA for Labwindows/CVI;
- A2.4.1 & A2.4.3 Building and description of the TPQA;
- A3.3.2 Guidance documentation on TPQA configuration of the built-in digitizers;
- A.3.3.3 Guidance on the integration of new digitizers.

CONTENTS

FIGURES

A INTRODUCTION

CMI and INRIM produced guidance documentation on the start-up, including installation of the software, and the user interface for both LabVIEW and LabWindowsTM/CVI environments (the 2 GUIs) based on information from the report on the software tool from A2.4.5 [15] and [16].

A.1 TPQA MANUAL

Following chapters will describe installation and basic usage of the TPQA tool [1].

A.1.1 Installation of drivers and development environments

Before running and installing of TPQA the user must complete the following steps and install several components:

1) Download and unpacking of the TPQA tool itself from GitHub [1]:

2) Installation of the prerequisites for the running of TPQA open tool project:
   a. LabWindows 2013 Runtime Engine. This is needed to run any LabWindows application;
   b. Drivers of all integrated instruments, which are currently:
      i. NI VISA drivers for handling of DMMs via GPIB IEEE-488.2 bus;
      ii. niScope drivers for handling of NI PXI 5922 digitizers.

3) Installation of the Matlab for data processing.

4) After downloading of TPQA.Zip file, unpack it and run the file *reg_matlab_dlls.m* into the matlab consol, for registering \bin\win32 matlab dlls, o add manually the path "*C:\Program Files (x86)\MATLAB\R2013a\bin\win32"* to the environment variables.

Steps 1), 2), 3) and 4) are mandatory. Current version of TPQA is built for DMM HP3458 and NI-PXI5922 digitizers. It is necessary to install VISA drivers even for use with NI 5922 cards and "niScope" drivers and even for use only with Agilent 3458A multimeters.

Step 3) is needed when processing of the recorder waveforms is performed off-line but the Matalb must be installed on the PC. For on-line processing TPQA offers an addition GUI interface. In that case, the algorithms employed for waveform analysis are based on specific functions distributed under *Measurement* LabWindows library. The 15RPT04 TracePQM protocol establishes that only the algorithm contained into QWTB should be validated. So, the output data coming from TPQA on-line processing module have not been validated. Their validation might be performed as additional task from end-users.

A.1.1.a Installing TPQA tool

TPQA tool requires no installation. Its files just must be unpacked from the ZIP to any user folder, e.g.:

C:\TPQA\TPQA_1.1.0

It can be downloaded from GitHub webpage (see Figure A.1).

However, user may also download development version which will run only with development version of LabWindows installed on the computer. Note the TPQA was intentionally developed in LabWindows/CVI minimum version 2013. The development version can be obtained downloading the GitHub as a ZIP file (see Figure A.2).

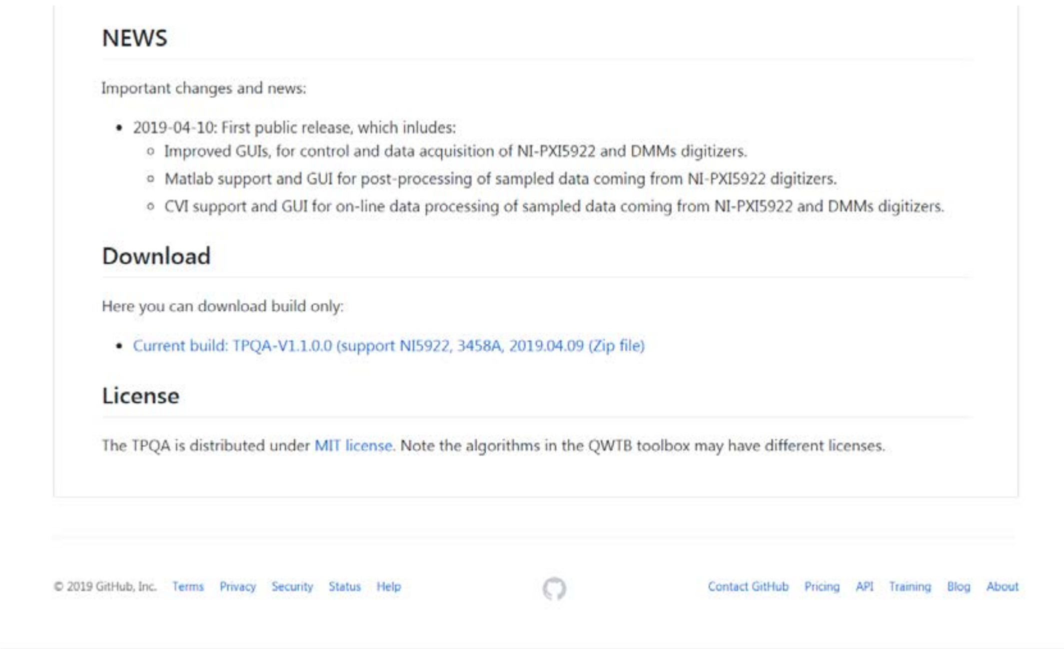


Figure A.1: Location of TPQA download on the TPQA GitHub webpage.

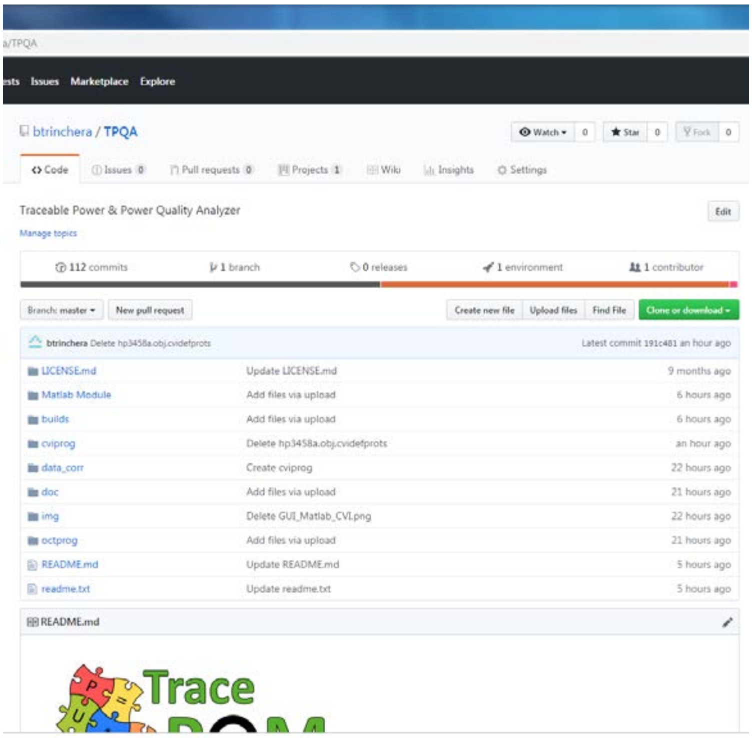


Figure A.2: Downloading development version of TPQA from GitHub webpage.

A.1.1.b Installing TPQA prerequisites

LabWindows applications in general requires large amount of external libraries and drivers to run. First required is LabWindows 2013 Runtime Engine 32bit [3]. The version must be 32bit as the TPQA is built as a 32bit application.

Next component are the VISA drivers, which are needed for communication via GPIB. At least version 5.4 should be installed. The newer versions are partially compatible, see [4] for selection.

Last needed component are niScope drivers, which are used to communicate with NI 5922 digitizers. These supported versions are 4.0.5 to 16.1 (see [5] for details).

A.1.1.c Installing Matlab

Matlab [2] is the only processing environment for the TPQA, however it should be totally compatible. TPQA was tested with versions 2007b and later. Typical Matlab installation should contain all required packages and they are loaded automatically, so no additional actions after installation should be required.

A.1.2 Startup

When all required components are installed, the TPQA can be started by its executable "TPQA_32bit.exe". When no component is missing, the front panel should appear with no error messages. If some driver is missing, LabWindows will show an error with explanation which component cannot be located. Typical missing components are "niScope.dll", "niTclk.dll" (part of "niScope" drivers) or VISA drivers. If TPQA requests the libraries, follow the installation guidance in section A.1.1. TPQA requires no other configurations prior starting the application itself.

A.1.3 User guide

Main panel of TPQA is shown in Figure A.3. All subpanels with particular configurations can be invoked from the main panel. E**rror** indicator at the bottom will show eventual error message of the TPQA. User must configure the system before any measurement can be taken. This is done by the main panel and from additional control buttons situated into panel **Acquisition and Processing Control.** The additional control buttons are as follows :

- **LF-Setup DMMs** for configuration of low frequecy setup based on HP 3458 digitizers ;
- **HW Corrections** for selection of the transducer and digitizer connection and correction files;
- **QWTB processing** for off-line data processing based on Matlab algorithms;
- **CVI Data processing** for on-line data processing based on LabWindows algorithms.

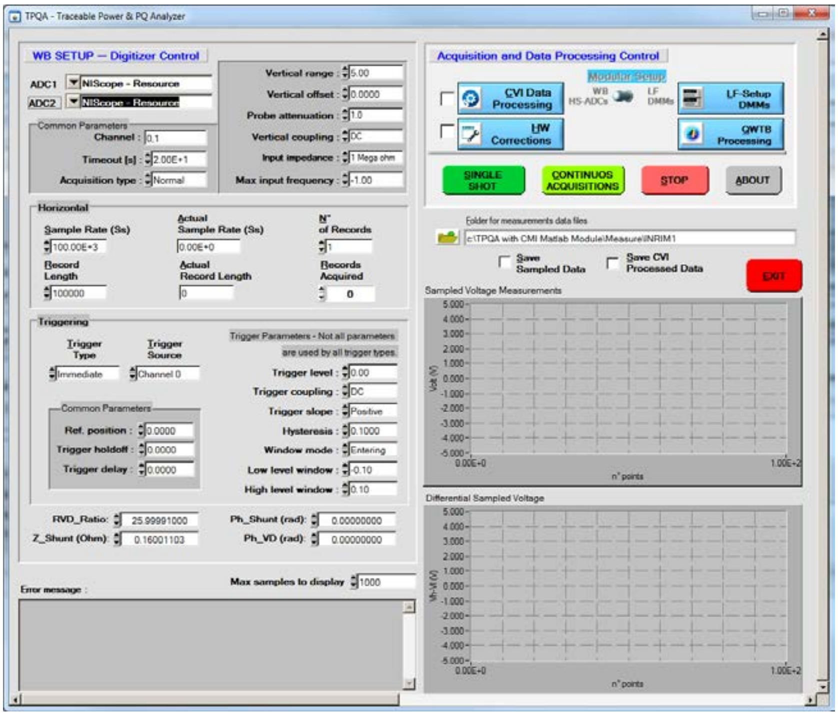


Figure A.3: Front panel of TPQA.

A.1.3.a Configuring digitizers for a new measurement setup

TPQA contains two distinct subpanels for the configuration of high precision digitizers and measurement setup.
The first subpanel concerns the configuration of high-speed digitizers employed for wideband power and power quality measurements, e.g. NI 5922, and is visible into TPQA main panel. The second one is focused on the configuration of high precision sampling DMMs mainly used for LF power and PQ measurements at power line frequency, e.g. HP-3458A.

I) Configuration subpanel for high-speed PXI-5922 digitizer

All the useful parameters for configuring high speed digitizers are inside the subpanel **WB SETUP – Digitizer Control,** as shown in Figure A.4. Below is reported a brief description of main control windings. Further explanations about the operation of the buttons and controls are reported in A.2.4.5 report describing the TPQA part of the open software project, located within \doc directory of TPQA project.

- The overall configuration of high speed digitizers is divided in three steps:
- Selecting digitizers and physical channels: using **ADC1** and **ADC2** it is possible to select all PXI-5922 resources recognized by niScope driver. Within **Common parameters** box it is possible to configure all physical channels of individual digitizing cards. Figure A.4 gives an example on how to configure digitizing cards for single-ended or differential voltage measurements according to power and PQ experimental setup specifications.

Figure A.4: Left single-ended measurements; b) right differential measurements.

- Configuring vertical and horizontal digitizer parameters: Figure A.5 shows the control parameters allowing the configuration of each physical channel of the digitizer. Vertical setting comprises common parameters as **Vertical range**, **offset and coupling**, **Input impedance** and additional parameters as **Probe attenuation** and **Max Input frequency**. Horizontal setting comprises common parameters as **Sample Rate** and **Record Length**, which must be set manually by users.

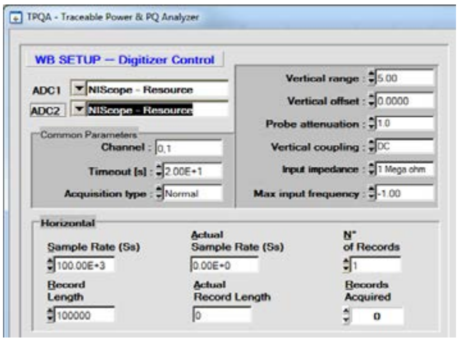


Figure A.5: TPQA configuration panel for vertical and horizontal setting of digitizer physical channels.

- Trigger setting: Figure A.6 shows the panel aimed at setting and arming digitizer trigger during the measurements. **Trigger Type** and **Trigger Source** are used to arm and set the input source of the trigger. For setting the parameters shown on the left, consult the PXI-5922 datasheet.

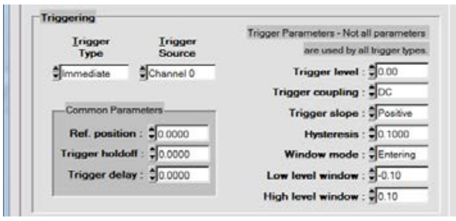


Figure A.6: TPQA configuration panel for trigger resources and setting.

II) Configuration subpanel for HP-3458 sampling DMMs

The subpanel developed to assist the user during the configuration low frequency macro setup based on high precision HP-3458 sampling DMMs is activated by pressing the button **LF-Setup DMMs** into **Acquisition and Data processing Control** box.
Figure A.7 shows the panel for HP-3458 DMMs digitizers configuration.

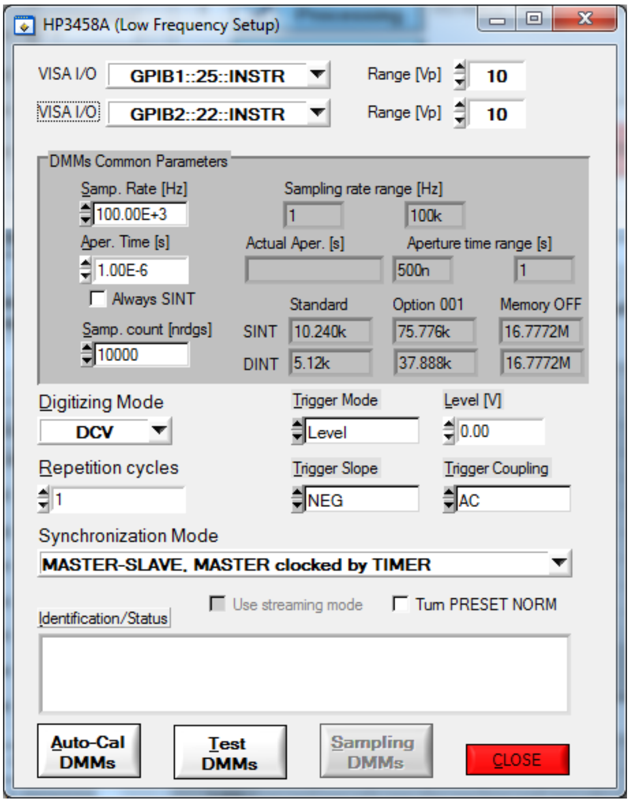


Figure A.7: TPQA configuration panel for HP-3458a DMMs digitizer.

It is composed of three main steps:

- *DMMs selecting:*
  - **VISA I/0** : Selects the GPIB address of the Master DMM;
  - **VISA I/O**: Selects the GPIB address of the Salve DMM.
- *DMMs common parameters settings:*
  - **Sampling Rate**: Reciprocal of the time interval state by the TIMER;
  - **Aperture Time [s]:** the DMMs A/D converter's integration time;
  - **Actual aperture**: reports the effective Aperture time value inserted;
  - **Apeture time range**: max and min value allowed according to sampling rate (please read the 3458A Manual);
  - **Sampling rate range**: it is made up all frequencies between the highest and lowest sampling frequency of DMMs;
  - **Sampling Count**: the total number of samples that the A/D must acquise;
  - **SINT (Single Integer)**: is one type of output formats for readings and it is composed by two bytes, is used during low-resolution measurement;

- **DINT (Double Integer)**: is one type of output formats for readings and it is composed by two bytes, is used during high-resolution measurement.
- **Standard, Option 001 and Memory OFF**: indicate three measurement mode (please read Sampling with 3458A [] for more details).

Note: The parameters placed in the zone of subpanel highlighted in grey are automatically calculated by software.

- *Sampling settings* :
  - **Digitizing mode** : Selects the digitizing method of DMMs;
  - **Trigger property** :
    - **Trigger mode**: the trigger mode indicates the signal condition that allows the representation of the acquired waveform starts. The user can choose three types of triggers (external, immediate and level). External means that there is another trigger of s different device. Immediate means that when the DMM starts acquiring, the waveform is also displayed. Finally, the level indicates the value of the trigger starting voltage;
    - **Trigger slope**: indicates if the trigger must be on negative or positive slope of waveform;
    - **Trigger coupling**: indicates the coupling type between trigger circuit and ADC channels. It can be DC or AC, without or with capacitive coupling;
    - **Level**: in this slash the User can put in the voltage value of trigger level if him chosen the level as trigger mode option.
  - **Synchronization Mode** [Stanislav]: indicates how multiple DMMs are synchronized. There are different type through them are synchronized: MASTER-SLAVE, MASTER clocked by TIMER, MASTER-SLAVE, MASTER clocked by AWG, All clocked by AWG e All clocked by TIMER. All this is explained better in Stanislav report [8].

At the bottom of the subpanel the user can choose to auto call or to test DMMs by pressing **Auto-Cal DMMs** or **Test DMMs**. When the user has finished to configure the HP3458 for sampling can to close the subpanel by pressing **CLOSE**.

A.1.3.b Configuration of measurement corrections

User must upload transducers corrections files before doing any meaningful measurement. This is done by pressing **HW Corrections** on the main panel, which will invoke panel shown in Figure A.8.

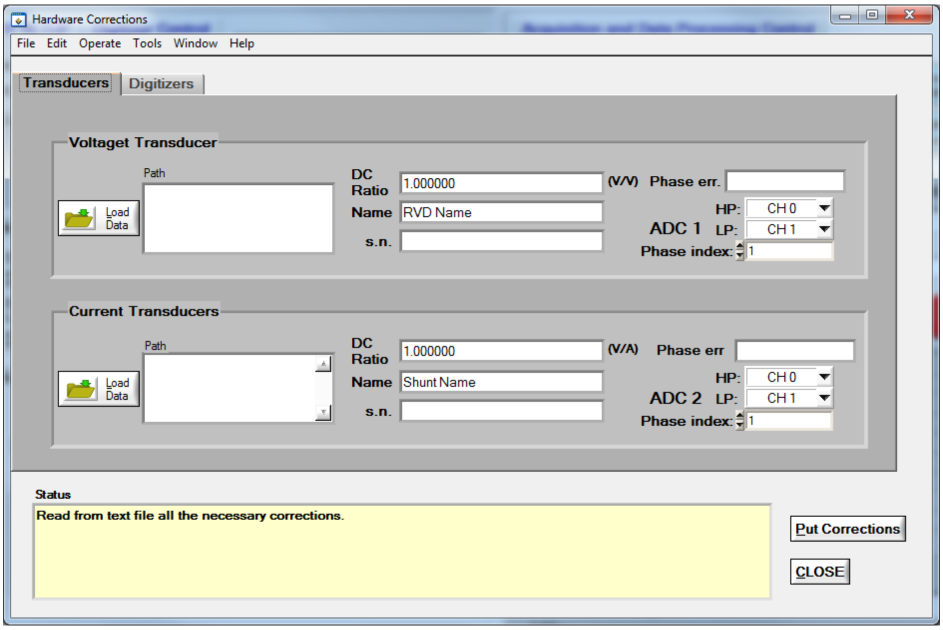


Figure A.8: TPQA hardware corrections configuration panel.

In the **HW Corrections** menu, there are two pages where to apply corrections: the first one is related to voltage and current transducers and the second one for Digitizer. The formats of the correction files is described in *A2.3.1 Corrections Files Reference Manual* [9]. Also in the same panel is possible to set how the transducers are connected to the experimental setup. Below it is given an example for single-ended and differential transducer configuration using the selectors **High (HP)** and **Low (LP)** to map the transducer outputs to particular digitizer virtual channels.

I) Transducers corrections

In the transducers page there are two sections where it is possible to upload the corrections for Voltage and Current transducers for both single-phase power and power quality measurements. The corrections for each transducer are loaded by pressing the button **Load Data**. Note that TPQA installation contains examples including "dummy" divider and shunt correction with unity transfers in the TPQA folder:

./data/corrections/transducers.

Once uploaded the respective files, the following boxes will be automatically filled with the related data: **DC Ratio**, **Name**, **Serial number** and **Phase error**.

For the ADC samplers it is instead required the channel selection depending on the type of measurement.

If the measurement is single-ended, the transducers are connected to the same ADC module, so **ADC1** and **ADC2** must be set only for the high potential as shown in Figure A.9. The low potential **LP** box must be set as not used.

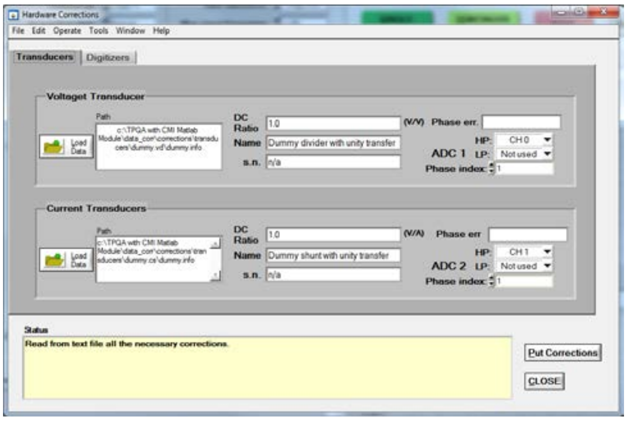


Figure A.9: Example of transducers configuration when ADC1 and ADC2 are set Single-ended mode.

If the measurement is differential, the transducers are connected to different ADC boards and the channels parameters must be set as shown in Figure A.10. Both **ADC1** and **ADC2** use the **HP** and **LP** boxes, set either on channel zero and channel one.

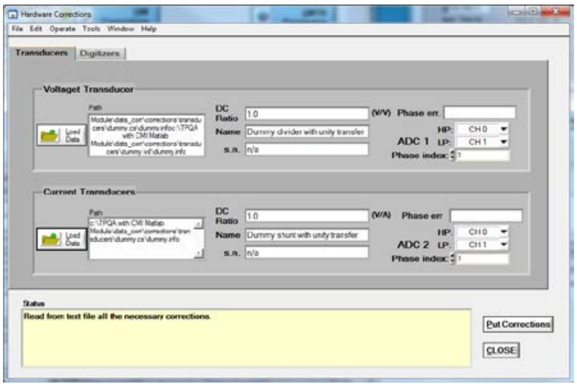


Figure A.10: Example of transducers configuration when ADC1 and ADC2 are set in Differential mode.

II) Digitizer corrections

In the second page it is possible to upload the digitizer corrections as shown in Figure A.11.

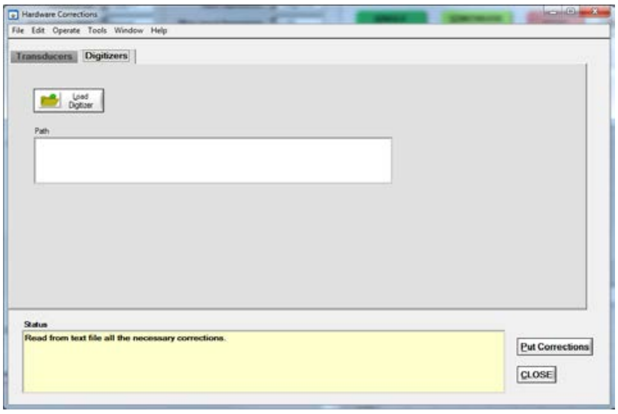


Figure A.11: Example of digitizer configuration panel.

For further information about the digitizer correction refer to section A.1.3.b of A3.3.1 Installation and Guide TWM [13].

A.1.3.c TPQA data processing

TPQA data processing is performed using two approaches: i) QWTB Processing for post-processing of raw data coming from ADCs; ii) CVI data processing for quasi real-time processing. Both modalities can performed using the same set of data. Below is reported a description of both approaches.

A.1.3.c.1 TPQA – QWTB Processing (post-processing)

This method of data processing is described in the activity A2.1.4 [10] “The concept for the interface between the data processing module and the control and data acquisition module”, of TracePQM-15RPT04, and only describes the concept for interfacing LabWindows/CVI to MATLAB tool. The report describes the off-line method, which performs the calculation of power and power quality parameters, according to the algorithms, using a set of static sampled data.
Here we report a brief description on how to use the same interface for the post-processing of sampled data coming from a single measurement session. All the consideration about this post-processing approach with respect to the data exchange are identical to those described for TWM open software tool project. For more details refer to the following reports:

- *A2.3.1 – Corrections Files Reference Manual*, [9];
- *A2.3.1 – Data exchange format and file formats*, [11];
- *A2.3.2 – Algorithm Exchange Format*, [12];
- *A2.4.4 – TWM algorithms description*, [6].

Once the measurement session has been configured, see section “B.2 Measurement configuration for LabWindowsTM/CVI environment (TPQA)” for an example of configuration, where is described as the user can use TWM algorithm on TPQA. This is done by pressing the button **QWTB Processing** on the main TPQA panel, which will invoke panel shown in Figure A.12 and the user must wait until the message (Please wait, starting Matlab interface … done) will not appear on the box Processing status (see Figure A.12).

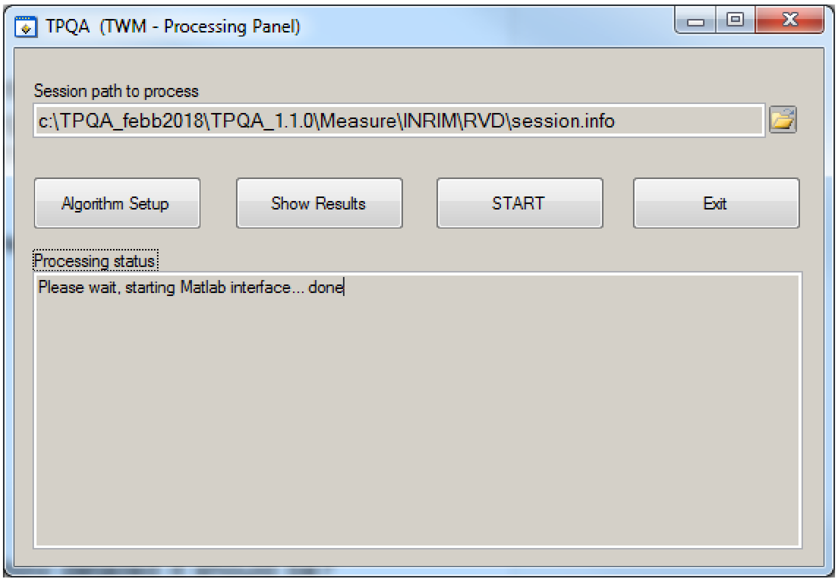


Figure A.12: TPQA (TWM processing panel).

The user must check if the **Session path to process** is correctly linked to the directory containing the session.info file generated during the measurements.
The User must press the button **Algorithm Setup** and the new panel **QWTB processing algorithms**, shown in Figure A.13, will be invoked.

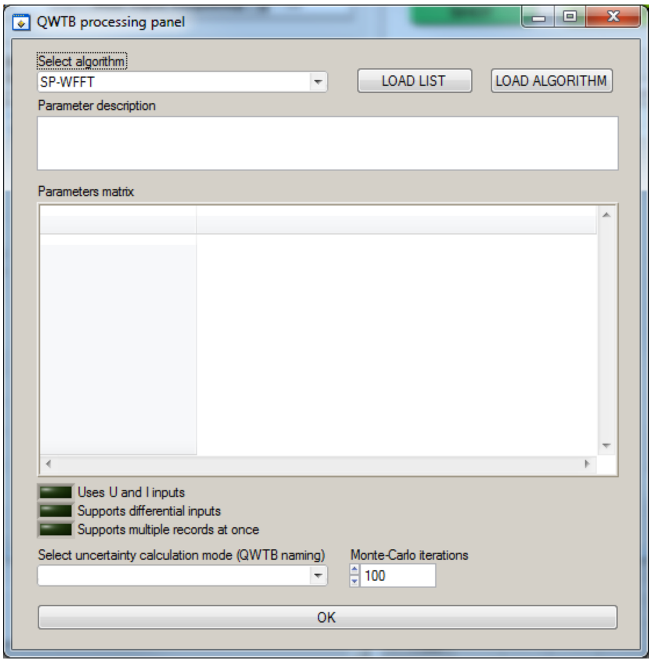


Figure A.13: Algorithm setup panel.

By pressing the button **LOAD LIST** all the available algorithms will be ready to be selected. The User can select the algorithm from a list of algorithms that appears pressing the button **Select Algorithm**. At the end by pressing **OK**, will invoke again the TPQA (TWM - Processing panel Figure A.12).

To continue with the post-processing of data the user must press the button **START**, which enables the data processing using the selected algorithms. In the **Processing Status** box a progressive number will appears, which indicates the record number under processing as shown in Figure A.14. User may need to calculate additional parameters from already digitized signals. TPQA is equipped by the batch processing tool for this case similar to TWM project. By selecting new algorithms and repeating the same procedure on the same set of digitized data new processing results will be available.

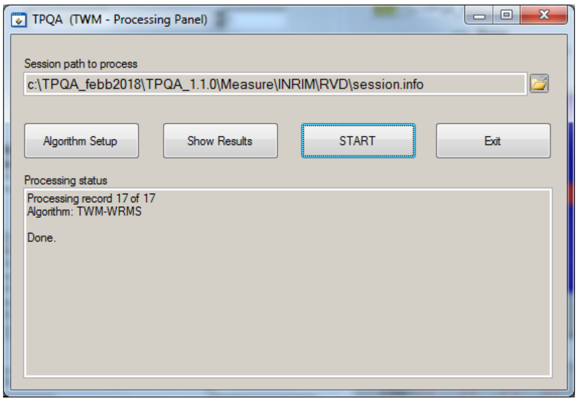


Figure A.14: TPQA (TWM processing panel) when processing record is finished.

To view the results of processing press the button **Show results** and a new panel will be invoked as shown in Figure A.15. Before to view results the user must check if the folder uploaded in the **Result file** box (see Figure A.15) is the same where have been uploaded the processing data. To view the results the user must press on **REFRESH** button and after this must control that all performed elaboration in batch processing show in the **Result matrix** box.

User by selecting the processing that a specific algorithm is responsible for (**Select Algorithm** button), can have access to all the measurement results.

The user by pressing the button **Select result** can change the visualization mode of the results.

If the user need to do other later processing with the results can use the copy and paste mode from panel.

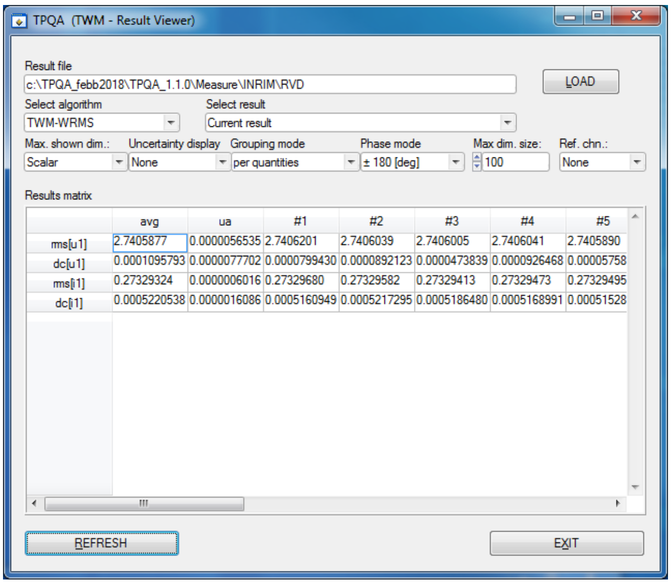


Figure A.15: TPQA (TWM-Result Viewer)

A.1.3.c.2 TPQA - CVI Data Processing (quasi real-time processing)

The second GUI for data processing elaboration can be activated by pressing "**CVI Data Processing**" command button, Figure A.16, which opens the panel shown in Figure A.17

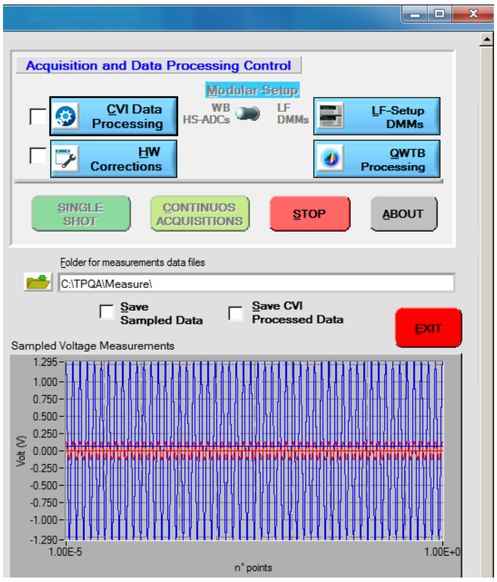


Figure A.16: Acquisition and Data Processing Control section and the measurement plot.

Figure A.18

The GUI that appears is called Data Processing (CVI).

It can be activated in continuous acquisition mode and it is possible to activate additional control panels as follows:

a) "**CVI Data Processing**" command button opens the panel shown in Figure A.17. Here there are situated the main waveform parameters computed using native algorithms furnished by NI (National Instruments).

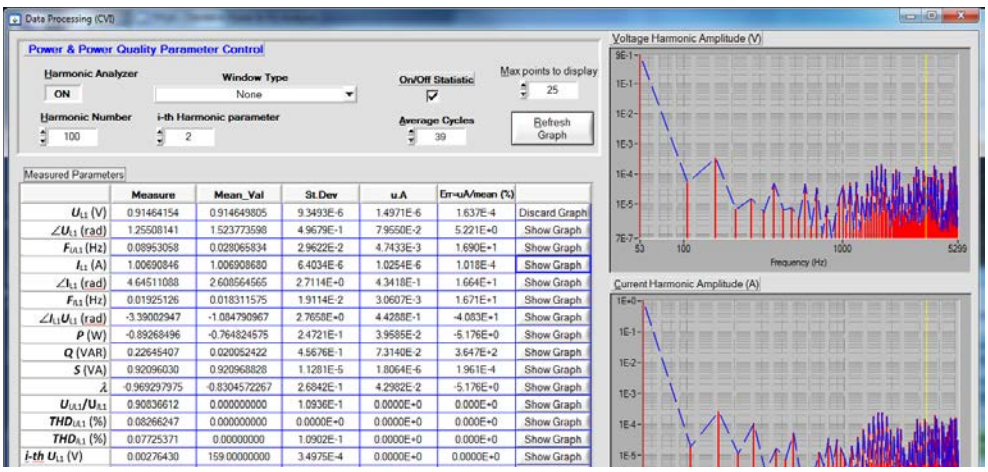


Figure A.17: Preliminary (on-line) data processing using NI-CVI FFT algorithms.

From this panel it is possible to set:

- **Harmonic analyzer ON/OFF**: this button allows according to the parameters set to "**Harmonic Number**" and "**Window Type**" to carry out an analysis on the frequency domain on the sampled waveforms. When enabled it shows on the right-side of the panel the harmonic graphs plotted against frequency. The first graph plotted refers to **Voltage harmonic amplitude (V)** and the second one to **Current harmonic amplitude (A);** both start from the characteristic frequency of 53 Hz until the selected Harmonic parameter;
- **Window type**: it permits to the user to select a specific type of window which enables different filters on the measured parameters. The program will set by default **None** of these options;
- **On/Off Statistic**: this flag allows computing statistical parameters such as **mean value, standard deviation, uncertainty A** and the **percentage error** of measured values. Everything will be displayed in the grid next to the measured values;
- **Average cycles**: this counter shows the number of samples that the system takes into account to evaluate the statistical parameters;
- **Max points to display**: it sets the maximum number of points to plot the graphs;
- **Refresh Graph**: update the harmonic graphs of Voltage and Current;
- **Show Graph:** these series of buttons next to each parameter in the grid invoke a new window displaying the corresponding measured value over time as shown in Figure A.18.
- **CLOSE**: this button allows to exit from the data processing panel.

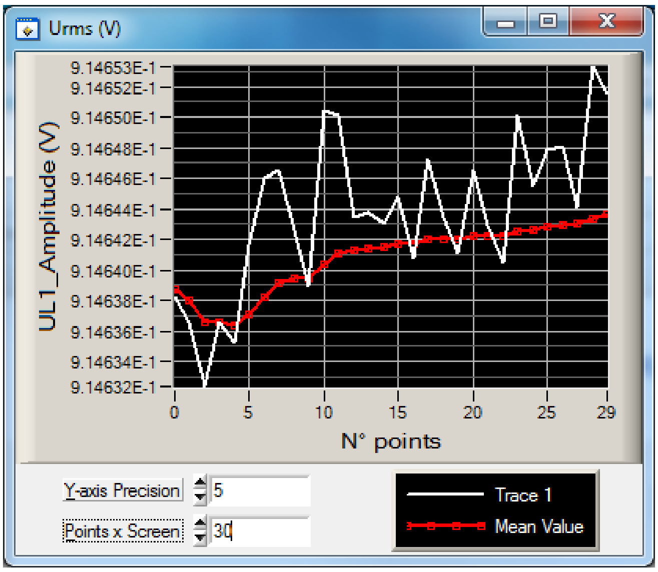


Figure A.18: **Selected Measured value graph.**

b) **Save Sampled Data:** this flag allows to save the measured parameter in the selected **Folder for measurements data files** path, as a .MATH or .INFO file;

c) **Save CVI Processed Data**: this flag allows to save all data extracted from the waveforms. It extract the whole grid containing the measured values plus all the statistical parameters as a .MATH or .INFO file.

B SOFTWARE CONFIGURATION OF THE BUILT-IN DIGITIZERS

Here is reported and example of the configuration of the TPQA open software tool to be used for power and PQ measurements. The example covers the configuration of the build-in digitizers with particular emphasis of the use of wide band macro setup based on PXI-5922 digitizers. ADCs as well as voltage and current digitizers are configured for single ended measurements. The example also report how to configure the data processing for quasi real-time processing using CVI native algorithms and QWTB processing tool box. The TPQA open software project setup has been tested as an executable file on INRIM new and existing measurement setup

B.1 MEASUREMENT SETUP FOR TESTING TPQA

TPQA handles measurement setups identified for low-frequency (LF) and wide band (WB) power and PQ measurements, according to *WP1: Design and validation of the modular power and PQ measurement setup.*

For WB measurements the system is based on single or dual NI-5922 digitizers. The digitizing boards can be configured for asynchronous or synchronous acquisitions using a common frequency reference. The macro setup based on NI-5922 digitizers has the advantage of being easily adaptable from a single phase to three phase measurements system.
With respect to the use of precision digitizers for the design of the WB system, the main features are:

- flexible vertical resolution depending on the sampling frequency;
- several synchronization and clocking strategies depending on the kind of PQ parameters under investigation;
- reconfigurable digital platforms for traditional, real time measurements and continuous acquisition for long time measurements beyond the capabilities of internal memory;
- simple synchronization of single digitizers for a polyphaser digitizer suitable for three-phase PQ measurements.

Figure B.1 shows a prototype modular system which comprises both LF and WB macro-setups developed at INRIM. Both LF and WB digitizers are handled by the same PXI chassis which has a control unit for remote control of both LF and WB digitizers based on a NI mainstream unit.

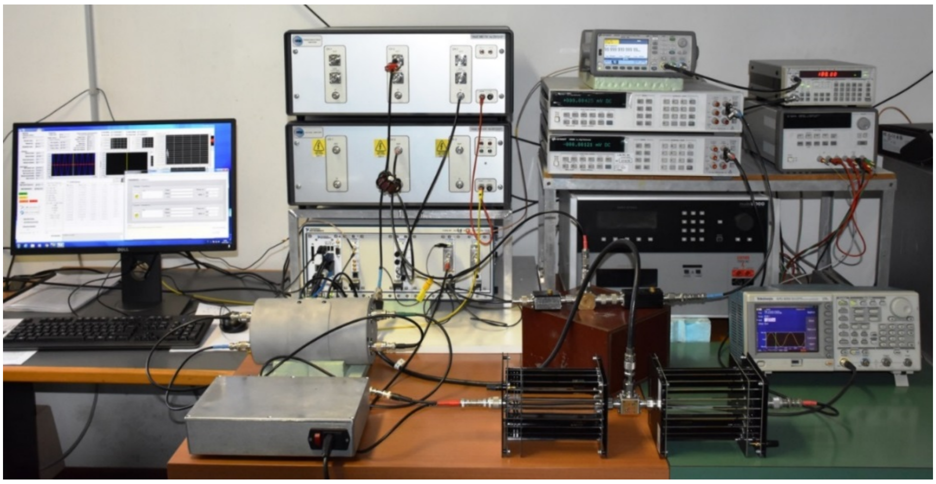


Figure B.1: **Macro setups modular system**

B.2 MEASUREMENT CONFIGURATION FOR LABWINDOWS™/CVI ENVIRONMENT (TPQA)

The following gives a typical configuration being used with TPQA software using a PXI-5922 digitizer configured for single ended power and PQ measurements.
The TPQA was developed and tested using the following hardware components:

- PXIe chassis mod.1085 equipedt with NI-PXIe-8840 embedded controller for real time measurements.
- Two high-bandwidth digitizers NI PXI-5922, to carry out differential sampled voltage measurement;
- Arbitrary waveform generator or clock generator, e.g., Agilent 331/332xxA, Agilent 335/336xxA, SRs CG635;
- Two USB ports of the NI-PXIe-8840 embedded controller are dedicated of interfacing of HP3458A digitizers using independent GPIB-USB-HS controllers
- NI PXI-8840 or equivalent mainstream unit with MS Windows Windows 7 or higher equipped with two or more USB ports.

1) <u>Configuration of WB digitizers, e.g. PXI-5922:</u>

- Select the digitizer from **WB-SETUP-Digitizer Control** menu on main panel control (see Figure A.3). For single-ended measurement select only the first digitizer, e.g. **ADC1** select the name on PXI-5922 board recognized by Ni-Scope driver.
- In the subpanel **Common Parameters** check and/or set:

  - **Channel**: set to 0,1;
  - **Time out** [s]: don't set lower than 20 s;
  - **Acquisition type**: set to Normal;
  - **Vertical range**: setup second the PXI 5922 vertical specifications;
  - **Vertical coupling**: select AC, DC or GND (default DC);
  - **Input impedance**: select 1 MΩ or 50 Ω (default 1 MΩ).
- In the subpanel **Horizontal** set:
  - **Sample Rate (Ss)**: set the sampling rate frequency of WB digitizer (default 100 kSs, maximum 15 MSs);
  - **Record Length**: set the number of points to be acquired (maximum ≈ 8 MS).
- In the subpanel **Triggering** set:
  - **Trigger Type**: set to desired mode (default Immediate, for synchronized measurements set Edge mode);
  - **Trigger source**: set to **channel 0** for **Edge** mode triggering;
  - For all further parameters see the PXI-5922 datasheet.

2) Transducers Corrections

- For correction to be used follow all steps described in A.1.3.b. Remember to specify how the voltage and current transducers are connected by selecting single or differential mode of operation.

3) Data saving

- User must select, for proper data saving and further elaboration, a directory specified in **Folder for measurement data files** placed in the main panel by pressing the related button. After selecting and or creating a new folder, e.g. press C:\TPQA\TPQA_1.1.0\Measure\INRIM\ press Done in the shown window;

- **Save Sampled Data**: used to save all acquired data, otherwise the TPQA will not save any data and no data processing will be available. In this mode it will be possible to save sampled data to be used for off-line data processing using **QWTB** tool box or other algorithms. After pressing **SINGLE SHOT** or **CONTINUOUS ACQUISITION** button the TPQA will open window where to store sequential bunches of data streaming, e.g. ADC_(n°).txt;

- **Save CVI Processed Data:** pressing this button TPQA will ask for additional file for storing processed data using CVI algorithms, based on FFT approach.

4) Data acquisition:

The user can collect data in two ways, by pressing the following buttons:

- **SINGLE SHOT**: for a single acquisition;
- **CONTINOUS ACQUISITION**: for a dynamic acquisition. In this mode TPQA always collect data and save them into specified folder by adding a progressive index in each file, e.g: ADC_n.txt. The index number appears in the numeric control **N° of Records**.

5) Data Processing:

There are two modes to show the elaboration of results on the acquired data:

1) Using QWTB tool box and its algorithms for off-line data processing as specified in the TWM guide [13]. This can be done pressing **QWTB Processing** button placed in the **Acquisition and Data Processing** control of the main panel window. This mode of operation is similar to those described in “QWTB batch processing panel” of the TWM open project tool [13]. After to have press **QWTB Processing** button, the **TPQA (TWM-processing panel)** will appear and the user must complete the following steps:

   - Press **Algorithm Setup** button and select the proper algorithm in **Select algorithm** control box and then press **OK**;
   - Press **START** button. In the processing status data log box will appear the name of algorithm and the processing records. Please wait until Done comes up.
   - Press **Show Results** button and the window **TWM result viewer** appears, otherwise user must select the **Measurement session**:
     - Check if the **Result file** reports the path containing measured data;
     - Press **Refresh** and in the **Select Algorithm** will appear the name of QWTB algorithm. The results shown in **Results matrix** can be copied and pasted in a spreadsheet, e.g. *.csv ;
     - On the same set of data the user can test all the available algorithms developed according to the TracePQM project.

2) Pressing **CVI Data Processing** an additional window for on-line data processing appears. This mode of introducing the results is useful to get a preliminary and partially traceable evaluation of the results in terms of common power and PQ parameters as described in A.1.3.c.2.

By using both the processing tool box, i.e. QWTB and CVI, it is also possible to perform a direct comparison between the common algorithms used for the estimation of main electrical power and PQ parameters on the same set of sampled data.

C INTEGRATION OF NEW TYPES OF DIGITIZERS

C.1 LABWINDOWS^{TM}/CVI ENVIRONMENT

TPQA open tool software employed in CVI environment has a modular structure which enable the possibility to fit new digitizers by developing prototype functions, which will be able to interact with the digitizer under test using vendor drivers through a translation layer. Before showing an example, it is important giving the concept of the flow chart developed for TPQA open tool software.

C.1.1 Flow chart LabWindows™/CVI environment

The TPQA is organized according to the flow chart diagram shown Figure C.1. The whole TPQA application consists of two parts:

(i) LabWindows modules (Control and Processing) that controls the instruments, initiates processing and serves as a user interface

(ii) Calculation or Processing module based on a double mechanism suitable to process digitized data using:

- quasi real-time approach for dynamic data processing based on FFT algorithms developed in Labwindows/CVI;

- post processing and formatting the data for displaying and generation of the measurement report (summary of the results formatted in compact form). The acquired data may be processed at any time. It is possible to just record batch of measurements without processing which may be helpful for time consuming calculations. The processing of the whole batch of measurements can be initiated later either via TWM or on a supercomputer. For further details the interested user can refers to TWM open software tool [1].

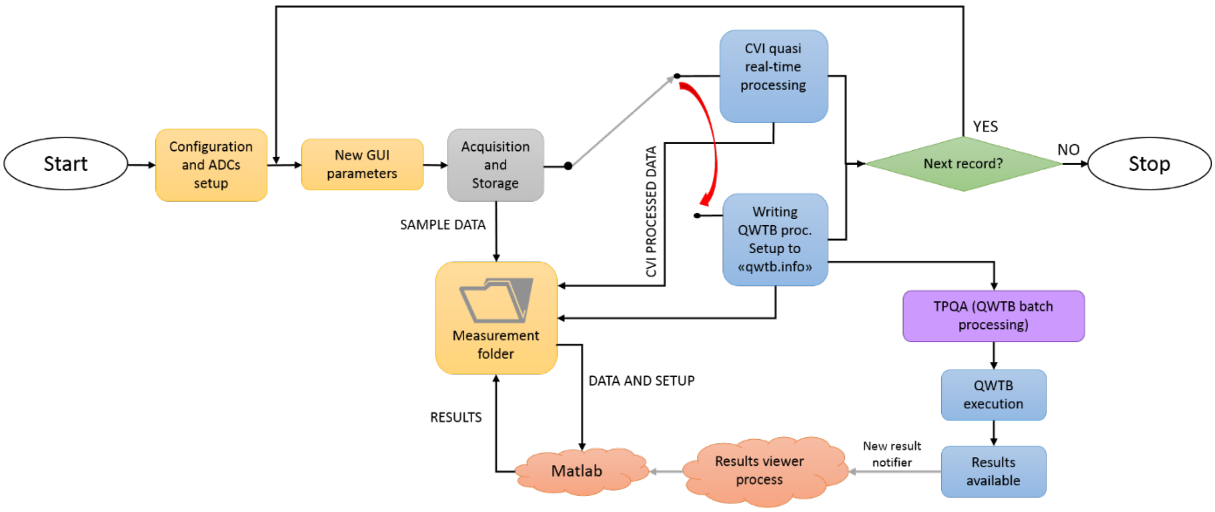


Figure C.1 **Flow chart for TPQA open software tool**

C.1.2 Integration of new digitizers in LabWindows/CVI environment

Concept of the virtual driver

The concept of the modular driver has been developed in order to avoid the direct access directly the drivers of particular instruments. It was verified that accessing directly of instrument drivers requires a different approach for each particular ADC board. It was substituted such an approach with a different mechanism which uses a *.c function translator able to interact as an interface layer between the data acquisition module and the physical instrument drivers which is called **Virtual driver**.

In TPQA open software project such an interface is composed for a specific board, e.g. PXI-NI-5922 digitizer, by two files a (*.c file and a header (*.h) file:

GenericMultiDeviceConfiguredAcquisitionTClk.c

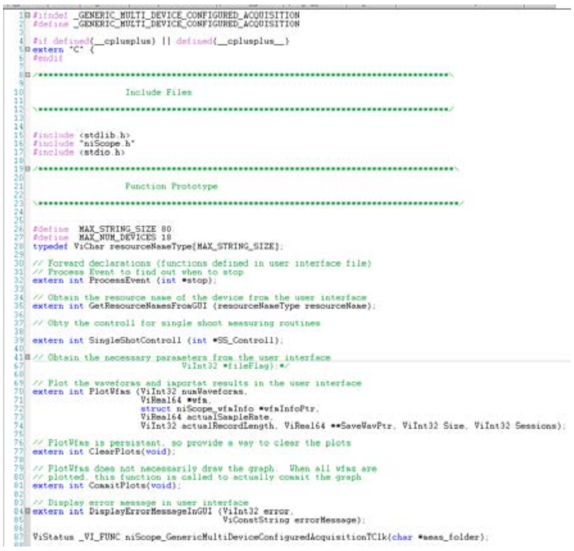
GenericMultiDeviceConfiguredAcquisitionTClk.h

Only some remarks to take into account regard to the data storing of sampled data when the digitizer runs continually. The solution adopted aims to collect the sampled data and then stored them directly on hard drive while the ADCs runs continually.

In Figure C.2 is shown the *.h file (header file) employed as translator for the virtual driver used with ni.Scope driver.

An example of the flow chart developed to describe the virtual driver employed with ni.Scope driver is shown in Figure C.3.

New virtual drivers could be inserted into the flow chart, substituting them with those highlighted in the flow chart in blue (see Figure C.3). This mean that the user will rewrite part of the code according to the vendor driver specifications.



```
#ifndef _GENERIC_MULTI_DEVICE_CONFIGURED_ACQUISITION
#define _GENERIC_MULTI_DEVICE_CONFIGURED_ACQUISITION

#if defined(__cplusplus) || defined(__cplusplus__)
extern "C" {
#endif

/*************************************************************************\

                    Include Files

\*************************************************************************/


#include <stdlib.h>
#include "niScope.h"
#include <stdio.h>

/*************************************************************************\

                    Function Prototype

\*************************************************************************/


#define  MAX_STRING_SIZE 80
#define  MAX_NUM_DEVICES 18
typedef ViChar resourceNameType[MAX_STRING_SIZE];

// Forward declarations (functions defined in user interface file)
// Process Event to find out when to stop
extern int ProcessEvent (int *stop);

// Obtain the resource name of the device from the user interface
extern int GetResourceNamesFromGUI (resourceNameType resourceName);

// Obty the controll for single shoot measuring routines

extern int SingleShotControll (int *SS_Controll);

// Obtain the necessary parameters from the user interface
                         ViInt32 *fileFlag);*/

// Plot the waveforms and importat results in the user interface
extern int PlotWfms (ViInt32 numWaveforms,
                     ViReal64 *wfm,
                     struct niScope_wfmInfo *wfmInfoPtr,
                     ViReal64 actualSampleRate,
                     ViInt32 actualRecordLength, ViReal64 **SaveWavPtr, ViInt32 Size, ViInt32 Sessions);

// PlotWfms is persistent, so provide a way to clear the plots
extern int ClearPlots(void);

// PlotWfms does not necessarily draw the graph.  When all wfms are
// plotted, this function is called to actually commit the graph
extern int CommitPlots(void);

// Display error message in user interface
extern int DisplayErrorMessageInGUI (ViInt32 error,
                                     ViConstString errorMessage);

ViStatus _VI_FUNC niScope_GenericMultiDeviceConfiguredAcquisitionTClk(char *meas_folder);
```

```
/*************************************************************************\
\*************************************************************************/

#define handleNIScopeErr(fCall)  {                                            \
        ViChar sessionNumber[4000];                                           \
        ViChar errorSource[4000];                                             \
        ViStatus code = (fCall);                                              \
        if ( (code < 0) || ((error == 0) && (code > 0)) ) {                   \
           error = code;                                                      \
           strncpy(errorSource, #fCall, MAX_FUNCTION_NAME_SIZE);              \
           errorSource[MAX_FUNCTION_NAME_SIZE - 1] = '\0';                    \
           strcpy(errorSource, strtok( errorSource, " (") );                  \
           sprintf(sessionNumber, " for session with (zero-based) index %d", i); \
           strcat(errorSource, sessionNumber);                                \
           niScope_errorHandler(sessions[i], error, errorSource, errorMessage); \
        }                                                                     \
        if (code < 0) goto Error;                                             \
     }

#define handleNITClkErr(fCall)  {                                             \
        ViStatus code = (fCall);                                              \
        if ( (code < 0) || ((error == 0) && (code > 0)) ) {                   \
           error = code;                                                      \
           niTClk_GetExtendedErrorInfo(errorMessage, 4000);                   \
        }                                                                     \
        if (code < 0) goto Error;                                             \
     }

#define handleSimpleExampleErr(fCall)  {                                      \
        ViStatus code = (fCall);                                              \
        if ( (code < 0) || ((error == 0) && (code > 0)) ) {                   \
           error = code;                                                      \
        }                                                                     \
        if (code < 0) goto Error;                                             \
     }

#define handleExampleErrWithText(text)  {                                     \
        error = -1;                                                           \
        strcpy(errorMessage, text);                                           \
        goto Error;                                                           \
     }


/*************************************************************************\

                    End Include File

\*************************************************************************/

#if defined(__cplusplus) || defined (__cplusplus__)
}
#endif

#endif /*_GENERIC_MULTI_DEVICE_CONFIGURE_ACQUISITION */
```

Figure C.2 Header structure of the translator used with ni.scope driver.

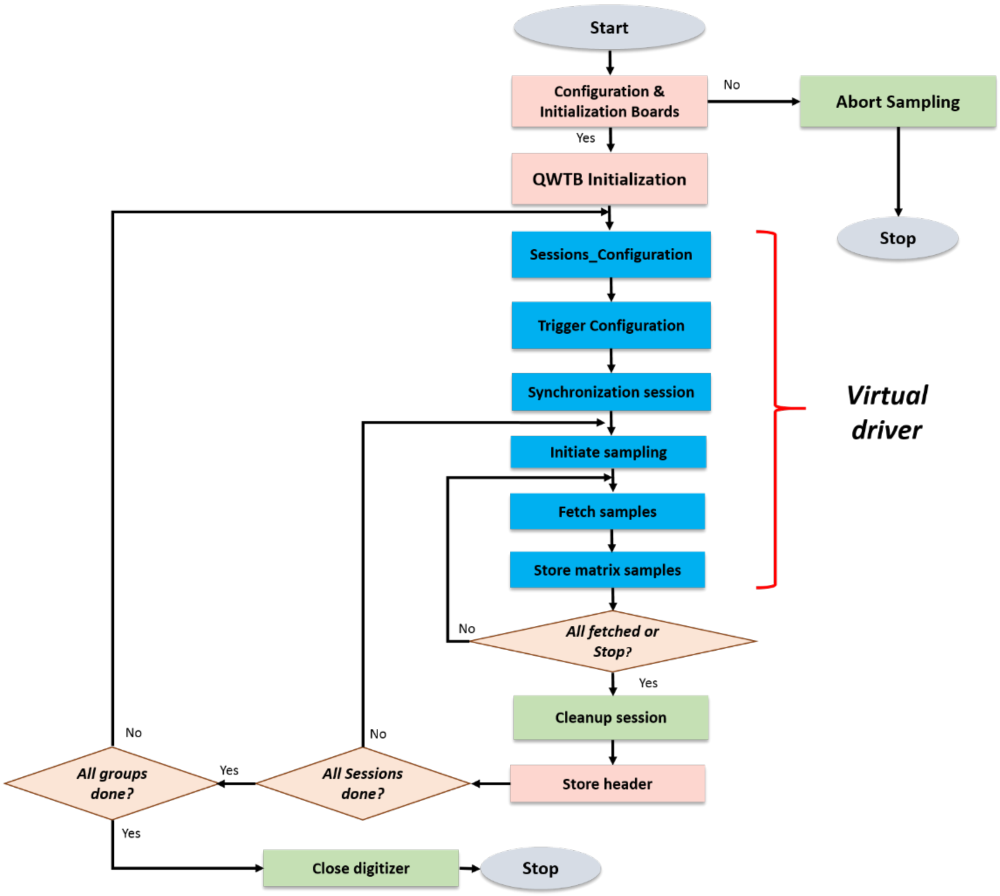


Figure C.3: Example of a structure for generic Virtual digitizer

The TPQA program allows to the user to change the parameters in real time. For this the developed TPQA code has some difference with the TWM structure [15].

D RESOURCES

[1] *TPQA tool*, url: https://github.com/btrinchera/TPQA

[2] MathWorks, *Matlab*, url: https://www.mathworks.com/products/matlab.html.

[3] National Instruments, *LabVIEW Run-Time Engine 2013 (32-bit)*, url: http://www.ni.com/download/labview-run-time-engine-2013/4059/en/.

[4] National Instruments, *LabVIEW and VISA drivers compatibility*, url: http://www.ni.com/product-documentation/53413/en/.

[5] National Instruments. *LabVIEW and niScope drivers compatibility*, url: http://www.ni.com/product-documentation/53540/en/.

[6] *A2.4.4 TWM algorithms description*, Available at "./TWM/doc" or url: https://github.com/smaslan/TWM/blob/master/doc/A244%20Algorithms%20description.pdf.

[7] *QWTB – Q-Wave toolbox*, url: https://qwtb.github.io/qwtb/.

[8] *A2.3.2 TWM file formats and concept*, available at "./TWM/doc", url: https://github.com/smaslan/TWM/blob/master/doc/A231%20Data%20exchange%20format%20and%20file%20formats.docx.

[9] *A2.3.1 Corrections Files Reference Manual*, available at "./TWM/doc", url: https://github.com/smaslan/TWM/blob/master/doc/A231%20Correction%20Files%20Reference%20Manual.docx.

[10] A2.1.4: Concept of Interfacing LabWindows/CVI to Matlab, url: https://github.com/btrinchera/TPQA/blob/master/doc/A214-%20LabWidowsCVI_to_Matlab_Interface.docx

[11] *A2.3.1 Data exchange format and file formats*, available at "./TWM/doc", url: https://github.com/smaslan/TWM/blob/master/doc/A231%20Data%20exchange%20format%20and%20file%20formats.docx.

[12] *A2.3.2 Algorithm Exchange Format*, available at "./TWM/doc", url: https://github.com/smaslan/TWM/blob/master/doc/A232%20Algorithm%20Exchange%20Format.docx.

[13] *A3.3.1Installation and Guide*, available at "./TWM/doc", url: https://github.com/smaslan/TWM/blob/master/doc/A331%20Installation%20and%20Guide.docx.

[14] R. Lapuh, *Sampling with 3458A,* Left Right, Ljubljana, 2018. ISBN : 9789619447604.

[15] *A2.4.5 TWM structure*, available at "./TWM/doc", url: https://github.com/smaslan/TWM/blob/master/doc/A245%20TWM%20structure.docx

[16] *A2.4.5 TPQA structure*, available at "./TPQA/doc", url: https://github.com/btrinchera/TPQA/blob/master/doc/A245_TPQA%20Structure_extended.docx