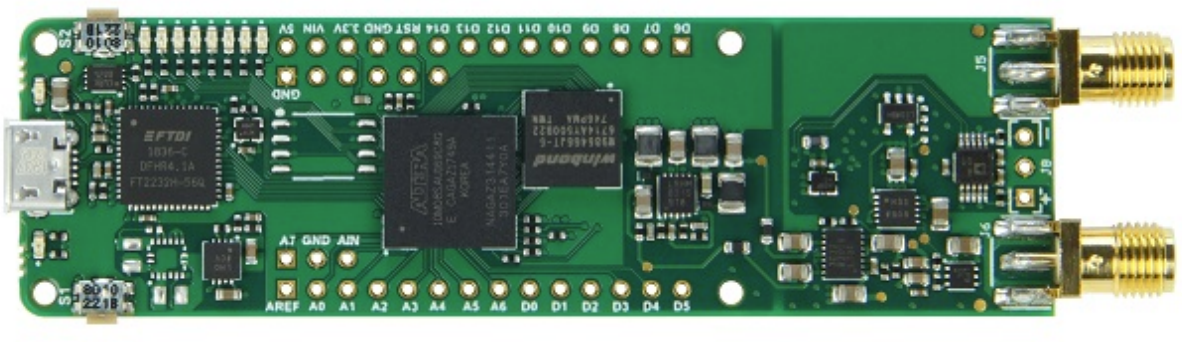
1 Documentation

- TEI0015 - Installation guide for Jupyter
- TEI0015 - Demo ADC data acquisition and Fourier transformation
- TEI0015 - Communication Interface and Commands
- TEI0015 - Programming guide: Interface or communication description

2 Download

- TEI0015 Jupyter Demo

3 TEI0015 - Installation guide for Jupyter



Example picture

3.1 Overview

The modules TEI0015, TEI0016 and TEI0023 offer software demonstrations of their basic functionality and communication interface. This manual provides a step by step guide for installing the required software to run the demo "Jupyter Demo FFT".

For the ease of accessibility and programmability the language **Python** (since version **3.8**) has been chosen. **Jupyter** provides an excellent and open source entry for beginners and professionals. Jupyter files are called Notebooks and have the ending **.ipynb**, this demos source code is contained within a Notebook file.

The following description of steps applies in its details to computers running windows 10, for other operation systems, the steps are in general similar.

3.2 Download

The demo is available through a link in the parent page.

The download is a folder, containing all the necessary files and documents. The **download folder** is compressed into a **zip archive** and needs to be extracted, to be accessible for Jupyter.

3.2.1 Making the demos accessible for Jupyter

Jupyter's file system access is limited to the user folder, so a convenient way is to copy the **extracted** demo folder into your users folder, for example:

C:\Users\Username**download-folder**

3.2.2 Content of the zip

Content of the zip archive - **TEI00XY-0x-08-C8A_Jupyter-Demo-FFT-vX.Y_quartusXY.Z_JupX.Y_JupLX.Y_YearMonthDay-Time .zip** :

- Folder "Jupyter Demo":
  The Notebook / Demo - TEI00-15-16-23_Jupter-Demo-FFT-vX.Y .ipynb
  A python code module - TEI00xyCodeModule .py
- Folder "Setup-Notebooks":
  The Notebook - Python_install_module_dependencies .ipynb
  The Notebook - Performs_modifications_to_JupyterLab .ipynb
  The Notebook - List_required_modules .ipynb
- Board configuration file TEI00xy... .pof
- Manual

3.3 Installation of "Anaconda Individual Edition" respectively Jupyter and JupyterLab

The simplest way to execute the demos in Jupyter is accomplished through the installation of "Anaconda Individual Edition" which is free and Open-source. This will install more than is needed but nearly all requirements in one step.

The Anaconda website provides detailed instructions on how to install the application, just follow this link.

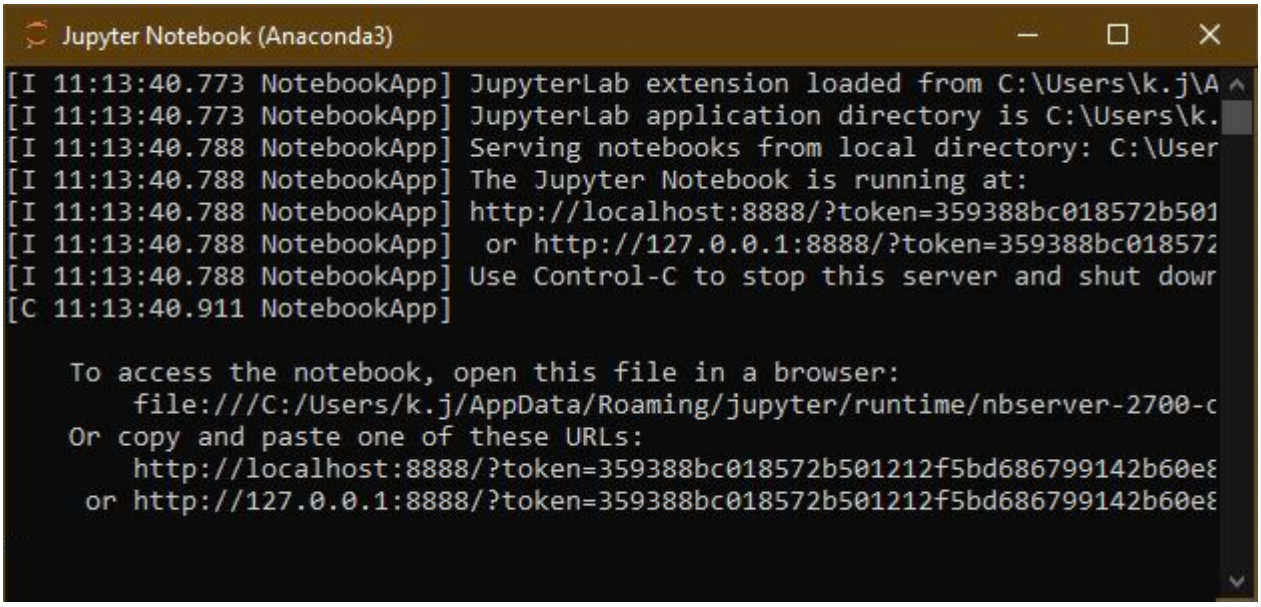
It is generally advised to use the default settings of the Anaconda installer. If the installer offers you the installation of optional applications, just skip those, they are not needed to run the demos or edit their code.

3.4 Starting Jupyter

To open Jupyter, press the windows key and type Jupyter, this presents "Jupyter Notebook (AnacondaX)" to you, from which one can start Jupyter.

Jupyter is based on the "Client-server structure", the server is executed in the background, and the client is a webpage inside your default browser. This webpage allows to navigate through the file system and to open and execute the Notebooks.

So opening Jupyter begins by starting the server, which is presented as a console displaying status messages of the server, shortly after, the webpage will be opened.
The console must be open all the time, you want Jupyter to run. You can minimize it.



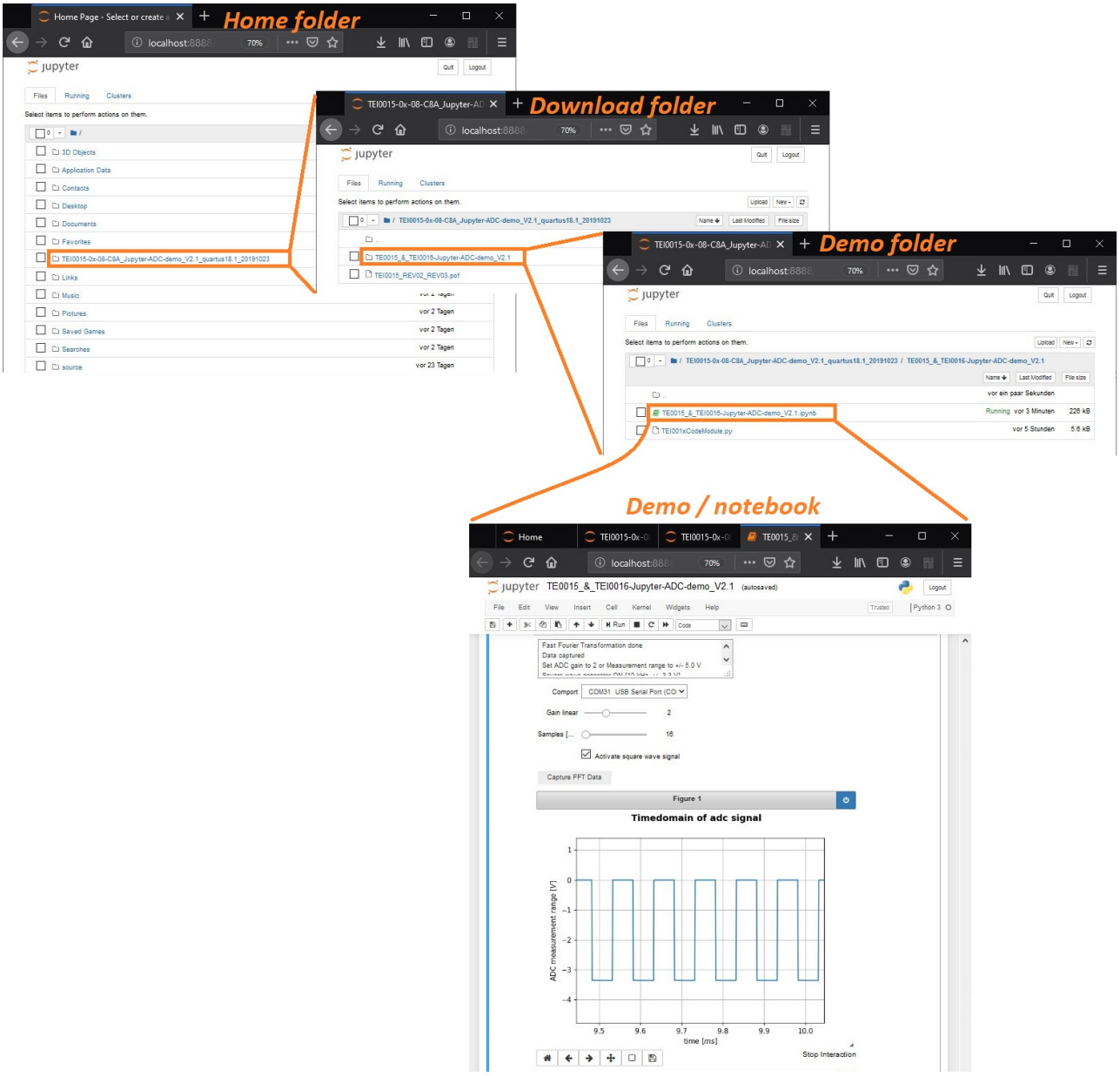
3.5 Using the Jupyter tab to navigate to a Notebook

The following is a general explanation on how to navigate via the Jupyter browser page. Generally, the Jupyter tab inside your browser is in itself a file browser. Jupyter has access only to your **user or home folder** which it displays after the program start. To start a notebook, one has to navigate to the folder containing the notebook file.

Left clicking on a folder opens it. Going back or a level up is accomplished by clicking the **back button** of your browser.

In the picture below these steps are exemplarily shown:

- Jupyter tab - Home folder
  → Download folder
- Download folder
  → Demo folder
- Demo folder
  → Demo Notebook

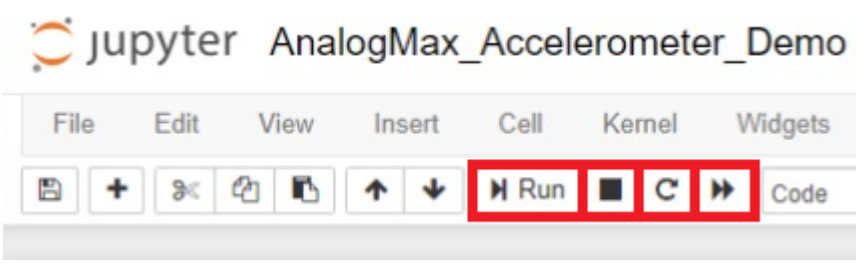


3.6 Executing a Notebook

A notebooks browser tab is the combination of a simple IDE together with the capability to directly run scripts.

A notebook consists of cells, in which its content, the applications source code resides. The code can easily be altered and directly executed from within the cell. The output of a cell for data, tables, graphs and GUI elements are below a cell. To a notebook can Cells be added or removed.

Before executing a Notebook, the PC's focus must be upon the cell, left clicking into the cells source code focuses this cell.



Now the demo can be run through pressing the **Run** button,
stopped by the button with the black rectangular,
reloaded through the circular arrow and lastly
reloaded and rerun by the two arrowed button.

Important! When the code of a Notebook was changed, one **has to reload** the Notebook.

The notebook can be closed either by closing the tab or via File → "Close and Halt".

3.7 Installation of Python software modules

Use the Jupyter browser tab to navigate along the path:
→ Jupyter tab - Home folder
→ Extracted demo archive : **TEI00XY-0x-08-C8A_Jupyter-Demo-FFT-vX.Y_quartusXY.Z_JupX.Y_JupLX.Y_YearMonthDay-Time .zip**
→ Folder: Setup-notebooks
Content:
Python_install_module_dependencies.ipynb
Performs_modifications_to_JupyterLab.ipynb
List_required_modules.ipynb

To install the missing Python code modules for this demo (serial/ pyserial/ ipympl), the Notebook **Python_install_module_dependencies.ipynb** has to be executed. This installs the required Python software modules.

After installation of the modules, Jupyter needs to be restarted.

JupyterLab requires also the execution of the Notebook **Performs_modifications_to_JupyterLab.ipynb** to function correctly (This modifies the output for interaction with matplotLib). Afterwards, JupyterLab can be started.

3.8 Executing the Demo Notebook

The parent page provides a link to a detailed description on how to use this Demo and install the module driver.

Connect a module via USB, if necessary install the driver. Every demo communicates with the module by a serial COM-port. This port has been assigned a number during the driver installation. This Demo notebook contains a graphical user interface. The available COM-ports will be scanned when the Notebook is executed. They are presented and selectable through the GUI. The Notebook containing the Demo **TEI00XY_Jupter-Demo-FFT_vX.Y .ipynb** is stored along the path:

→ Jupyter tab - Home folder
→ Extracted demo archive : **TEI00XY-0x-08-C8A_Jupyter-Demo-FFT-vX.Y_quartusXY.Z_JupX.Y_JupLX.Y_YearMonthDay-Time .zip**
→ Folder: Jupyter-Demo
Content:
TEI00XY_Jupter-Demo-FFT_vX.Y .ipynb
TEI00xyCodeModule.py

As an option, a default COM-port can be set at the beginning of the notebook and the style of the FFT plot can be altered from logarithmic to linear. At the bottom of the Notebook, the size of the graphs can be altered.

3.9 Annotations

- The demos are only linked to Jupyter through the import and use of the following modules:
  IPython - Interacting from within the Notebook with its output / plots
  ipywidgets - Interactive Widgets/GUI elements for the Jupyter Notebook
- Show line numbers inside a notebook: View → Toggle Line Numbers
- Autocomplete is available when pressing the Tab key
- Variables of a notebook cell are not encapsulated from other cells inside the same notebook

4 TEI0015 - Demo ADC data acquisition and Fourier transformation

4.1 Overview

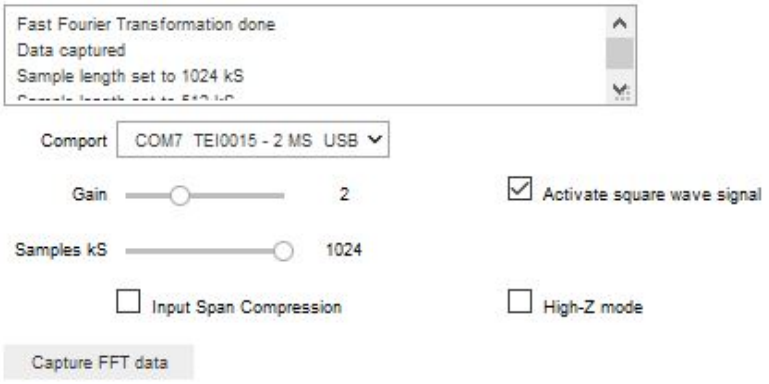
This demo provides an example on how to use the communication interface provided in the modules firmware to setup the pre-amplification gain and trigger an ADC measurement. This measurement is converted to show it's value over time, and it's Fast Fourier Transformation / frequency spectrum.

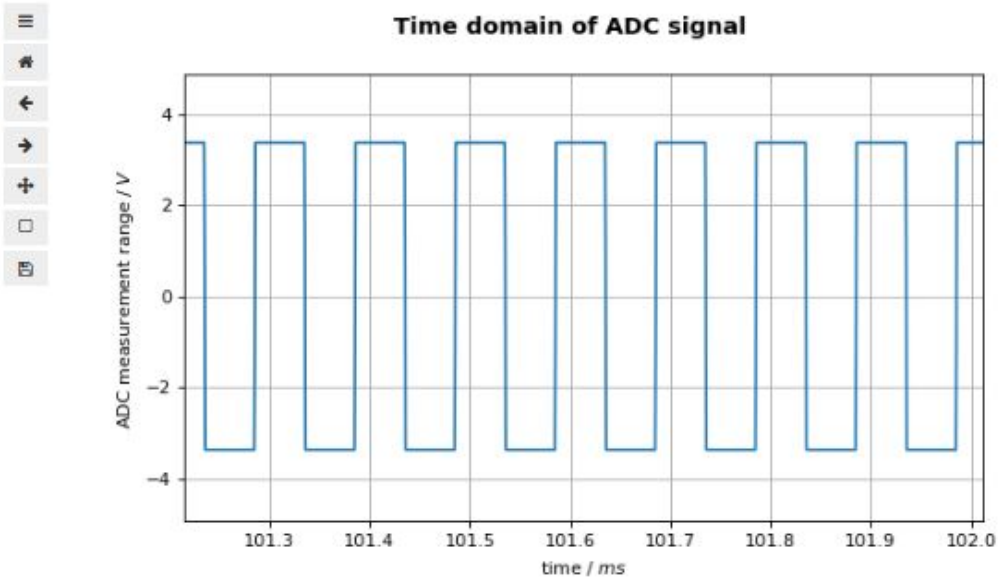
The **download** contains the **demo** in a separate folder, a folder with **Setup Notebooks**, the **FPGA Design file** and the **related wiki pages** in pdf format.

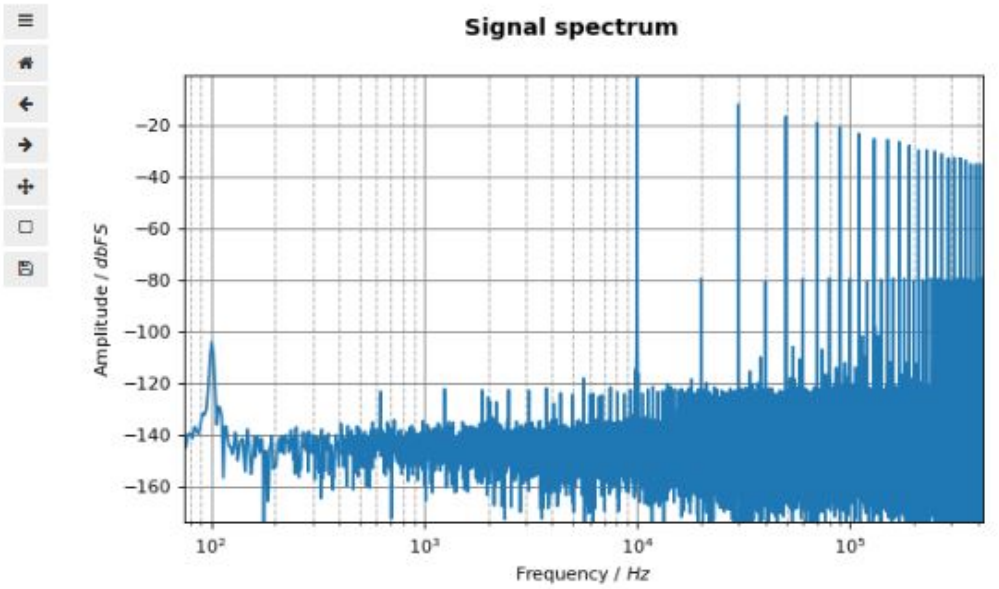
The demo consist of two files, one is the Notebook which contains the GUI and the other is a code module, containing the analytical part of the demo. These files must be in the same folder when running the demo. The modules ADC is controlled via the **Intel Max 10** FPGA. In shipment condition, the FPGA is programmed with the necessary **FPGA Design** to run this demo.

The wiki page "**Installation guide for Jupyter**", available via the superior page, describes **how to run Jupyter, install this Notebooks dependencies, open a Notebook** and **execute it.**

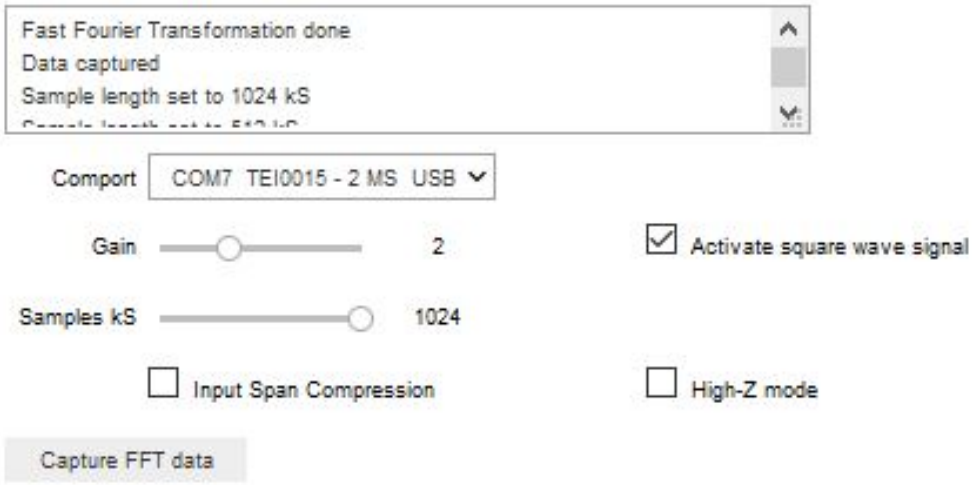
Picture of the Notebook







4.2 GUI elements



Drop-down List "**Comport**" :
- COM-port list for selecting a COM-port.
Listed are the port, the modules name and its USB ID
During notebook initialization, ports are scanned

Slider "**Gain**" :
Select the pre-amplification gain of the module
Different values in dependency to the module

Slider "**Samples kS**" :
Slider to set the samples to evaluate
Range is from 16 kS to 1024 kS

Check box "**Activate square wave signal**" :
Activates the modules square wave generator
10 kHz, +3.3V, 50% duty cycle, signal available with and without 180° phase shift

Check box "**Input span compression**" :
Mode of the modules ADC in which it's reference voltage is reduced to 80% to give head room in case the supply voltage is near to the reference voltage
Not available on all modules, greyed out / deactivated for these modules

Check box "**High-Z mode**" :
Mode of the modules ADC in which it reduces its current consumption. Not applicable to signal greater then 100 kHz
Not available on all modules, greyed out / deactivated for these modules

Button "**Capture FFT data**" :
Re-applies the appointed GUI values and features to the module and triggers a signal measurement followed by the data conversion and plotting

4.3 Using the demo

This demo is designed with the focus of capturing signals. Therefore the module works like a storage oscilloscope.

For every trigger event, the module saves 1 million samples of ADC measurement in its SDRAM. In the Notebook, the user just selects which amount of the data are to be used for the evaluation.

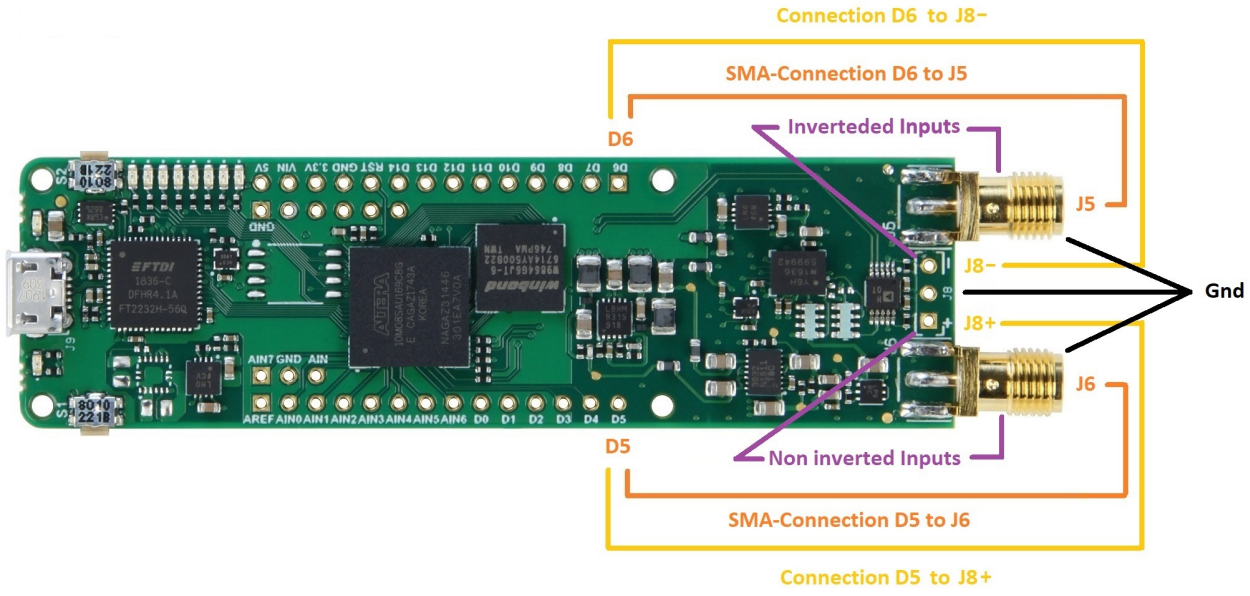
4.3.1 Module and host pc setup

The Notebook displays a graphical user interface in its running state, for setting up the comport, sample parameters, gain and module specific features. The following list describes the setup steps necessary prior to executing the Notebook:

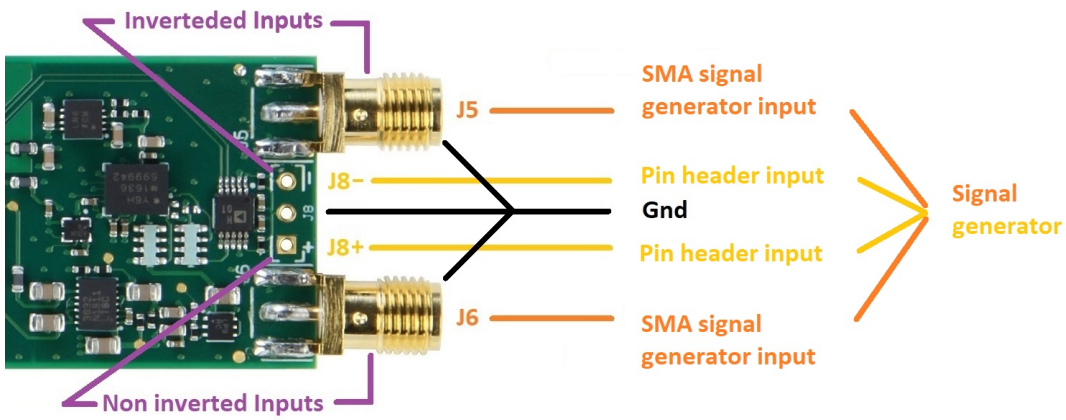
- Connect the module to the host pc
- Apply a signal to the modules inputs:
  Either via SMA- or via Pin Header- connection,
  it is sufficient to use only one of the differential inputs,
  but for the best results, apply the signal via both differential inputs
  (Connection diagrams are in the next chapter)
- Connect the module to the host pc
- Open Jupyter and navigate to the Notebook.
- Place the mouse cursor into the Notebooks cell via left click
- Run the Note by pressing the Run button
  (The demo scans for existing comports in its initialisation phase. So the module needs to be connected to the computer prior to running the demo)
- Enjoy !

4.3.2 Connecting the input signal

Connecting the square wave generator to the modules inputs



Appling a signal to the modules inputs



5 TEI0015 - Communication Interface and Commands

The modules TEI0015, TEI0016 and TEI0023 implement a handler for executing commands. The serial interface speed
must be set to 115200 bits, commands consists of a single character in UTF-8 encoding.
Each command must be transmitted individually.

All commands are identical for all tree modules, except commands for setting the gain.

The modules **TEI0015** and **TEI0016** recognizes the following gain commands since module revision 02:

"**1**" Sets the pre-amplification of the ADC's input to 1

"**2**" Sets the pre-amplification of the ADC's input to 2

"**4**" Sets the pre-amplification of the ADC's input to 4

"**8**" Sets the pre-amplification of the ADC's input to 8

The following gain commands are recognized by all **TEI0023** modules in every module revision:

"**0**" Deactivates the pre-amplifier

"**1**" Sets the pre-amplification of the ADC's input to 0.25

"**2**" Sets the pre-amplification of the ADC's input to 0.5

"**3**" Sets the pre-amplification of the ADC's input to 1

"**4**" Sets the pre-amplification of the ADC's input to 2

"**5**" Sets the pre-amplification of the ADC's input to 4

"**6**" Sets the pre-amplification of the ADC's input to 8

"**8**" Sets the pre-amplification of the ADC's input to 16

The modules **TEI0015** and **TEI0023** have an ADC with additional features since revision 02 for **TEI0015** and revision 03 for **TEI0023**:

"**S**" Activates the Input Span Compression

"**s**" Deactivates the Input Span Compression

"**H**" Activates the High-Z Mode

"**h**" Deactivates the High-Z Mode

These commands are recognized by **every module**:

"**t**" The ADC measures 1 mega samples and saves the values into its SD-RAM

"**x**" Instead of ADC values, the value "12345" is stored 1M times into its SD-RAM, values are transmitted via ".", "+" and "*"

"**y**" Instead of ADC value, hexadecimal values, in ascending order, are generated and stored into the SD-RAM, the values are transmitted in via "." , "+" and "*"

"**z**" The value "12345" is generated and direct transmitted 256 times

"r" The ADC measure once and transmits this value

"**.**" A single value of stored ADC or generated measurement is transmitted

"**+**" 128 values of stored ADC or generated measurements are transmitted

"*****" 16 kbit values of stored ADC or generated measurements are transmitted

"**?**" The module returns its ID:
TEI0015 with ADC AD4003 / 2 MSps returns "1"
TEI0016-0x-08-C8**A** with ADC ADAQ7988 / 0.5 MSps returns "2"
TEI0016-0x-08-C8**B** with ADC ADAQ7980 / 1 MSps returns "3"
TEI0023A with ADC AD4003 / 2 MSps returns "4"

"**F**" The module activates a square wave signal,
frequency = 10 kHz and amplitude is +3,3 V / ground
the signal is accessible on the pads
- D5 in normal mode and
- D6 in time inverted mode

"**f**" Deactivation of the square wave signal

6 TEI0015 - Programming guide: Interface or communication description

6.1 Communicating with module

To communicate with the module, a serial COM-port with a speed set to **115200** bits needs to be opened. Commands consists of a single **character** in **UTF-8** encoding. It is good practice to communication with the module following these steps:

- Open a serial COM-port
- Clear the PCs serial comport input buffer of the opened comport

- Send the desired commands, each one in a single write operation to the comport
- Close the serial comport as soon as possible

These steps apply also for read operations.

6.2 Using the ADC for high speed consecutive measurements

The module provides a method to gather highly accurate consecutive ADC measurements in a single event. In this mode of operation, one mega sample of ADC values are performed and stored inside the modules SD-RAM.

The following step should be taken in this mode:

- Open a serial COM-port
- For TEI0015 and TEI0016: Send the command "1", "2", "4" or "8" for the ADC pre-amplification of 1, 2, 4, 8
  For TEI0023: Send the command "1", "2", "3", "4", "5", "6" or "7" for the ADC pre-amplification of 0.25, 0.5, 1, 2, 4, 8 and 16
- Send the command "t" to trigger the consecutive measurement.
  (The module always measures 1 MSample of data into its SD-RAM)
- Clear the PCs serial comport input buffer of the opened comport
- Send the command "+" or "*" to the module, it then transmits 128 or 16384 Samples of ADC values
- Read the amount of ADC values in one chunk of 128 or 16384 samples from the PCs serial input buffer
  (Otherwise there is a high possibility of a misalignment of nibbles)
- Repeat the reading of chunks to a maximum of 1 mega sample
- Close the comport

After a trigger event, the one mega sample of data is stored until your retrigger. So processing the data can be done for each chunk individually or the whole one mega sample.

Convert the RAW ADC data into standard integer values.

6.3 Brief ADC informations

6.3.1 Module TEI0015 / AD4003BCPZ-RL7

Resolution: 18-bit in 5 nibbles
Maximum sampling rate: 2 MSPS

Order of Values:

	Hex	Dec		Hex	Dec
Mid scale	0x00000	0			
Positive 1 LSB	0x00001	1	to full scale -1 LSB	0x1ffff	131071
Negative full scale	0x20000	131072	to -1 LSB	0x3FFFF	262143

The layout of the ADC circuit is further described in the Analog Devices circuit note CN-0385.

6.3.2 Module TEI0016 / ADAQ7988 or ADAQ7980

Resolution: 16-bit in 4 nibbles
Maximum sampling rate: 0.5 MSps / 1 MSps

Order of Values:

	Hex	Dec		Hex	Dec
Negative full scale is	0x0000	0	to -1 LSB	0x7fff	32767
Mid scale is	0x8000	32768			
Positive 1 LSB	0x8001	32769	to full scale	0xffff	65536

The layout of the ADC circuit is further described in the Analog Devices circuit note CN-0393.

6.3.3 Module TEI0023 / ADAQ4003BBCZ

Resolution: 18-bit in 5 nibbles
Maximum sampling rate: 2 MSPS

Order of Values:

	Hex	Dec		Hex	Dec
Mid scale	0x00000	0			
Positive 1 LSB	0x00001	1	to full scale -1 LSB	0x1ffff	131071
Negative full scale	0x20000	131072	to -1 LSB	0x3FFFF	262143