FFT Module Reference

This document is the reference for the FFT sub module of the Python SLab system.

Index

Introduction

The “SLab System” is a software solution that provides a low cost access to circuit measurements by the use of low cost development boards.
The complete system has a natural tendency to feature creeping. In order to make the dimension of the project easy to manage, it has been divided in several modules.
The main module, associated to the file slab.py, contains all the code that directly accesses the Hardware Board and the most immediate low complexity commands.
The functionality of the main module is extended by the use of additional modules. This document describes the Fast Fourier Transform FFT module, associated to the file slab_fft.py that includes a set of functions to work with the signals in the frequency domain.

If you want to use the FFT module you will also need to use the main slab module as it is the one that includes the ***connect*** command that starts the communication with the Hardware Board. That means that you will need to use two imports.

```
>>> import slab
>>> import slab_fft as fft
```

If you need to use other additional modules you can add more lines for other sub modules. Observe that we have used an alias name for the module namespace, so, a command with the name “command” will be accessed as “fft.command”.

FFT Related Commands

The SLab uses numpy to perform the FFT over measured signals. This section includes all commands that depend on the FFT.

ftrasform(signal, time, ts)

Arguments	signal	Vector with time dependent signal	V1
	time	Sampling time	
	ts	Maximum sample frequency (Defaults to board reported)	
Returns		List of complex gain numpy array	

Transforms from time to frequency domain. Uses the FFT of the signal and performs several corrections:

1) Only positive frequencies are provided
2) Factor 2/N applied except for DC that use 1/N

In order to generate the frequency vector, sample time of **signal** is needed. It can be obtained from the **time** vector or from the sample time **ts**. If neither **time** nor **ts** is provided, the command will use the current sample time

Returns a tuple with:

1) Complex amplitude vector
2) Frequency vector

distortion(v1, v2, freq, show)

Arguments	v1	Low peak value of tone (Volt)	V1
	v2	High peak value of tone (Volt)	
	freq	Frequency of tone (Hz)	
	show	Show curves and text if true (Defaults to True)	
Returns		Tuple with: THD, THD+N,2nd H, 3rd H	

Generates sine wave tone at DAC1 at the indicated frequency and peaks and reads a circuit output at ADC1. Calculates four values related to distortion.
Noise floor limits measurements. A limit in THD and THD+N of 0.2% and a limit of -60 dBc on 2nd and 3rd harmonics are expected when using a Nucleo board.
Connect DAC1 to ADC1 to obtain the floor limits for the hardware board.

If the ***show*** parameter is true (default case), the obtained signal in the time and frequency domain is shown together with some text that gives the four reported distortion values.

Returns a four element tuple with the following distortion values:

1) THD (%)
2) THD+ N (%)
3) 2nd Harmonic (dBc)
4) 3rd Harmonic (dBc)

References

STM32 Nucleo Page

http://www.st.com/en/evaluation-tools/stm32-mcu-nucleo.html

MBED Developer Page

https://developer.mbed.org/

Python page:

https://www.python.org/

Anaconda page:

https://www.continuum.io/downloads

TinyCad

https://sourceforge.net/projects/tinycad/

Circuit images on this document have been drawn using the free software TinyCad