**PITCH FINDER API**

1. **Void Main ()**

**Location:** “pitchFinder.c”

**Description:** this is the code that will launch when executing the program. It listens for audio inputs and finds out its Pitch in real time.

The pitch finder is designed to detect, identify and display a pitch between 10 Hz and 4kHz. It works with a sampling rate of 8kHz.

**Parameters:**

* + Framesize: the number of samples that will represent the sound segment in the frequency domain.
    - Framesize defined as **128**.
  + Separation Th I, Separation Th II: these are both the splitting frequency intervals; they determine where do the high, mediums and low frequencies end and begin.
    - Separation Th I defined as **20.**
    - Separation Th II defined as 4**0.**

**Dependencies:**

- <dsp\h\dsp.h>

- <board\h\sask.h>

- <peripherals\adc\h\ADCChannelDrv.h>

- <peripherals\pwm\h\OCPWMDrv.h>

- <board\inc\ex\_sask\_generic.h>

- <board\inc\ex\_sask\_led.h>

- <peripherals\timers\inc\ex\_timer.h>

1. **VOID ADCChannelRead (pADCChannelHandle,frctAudioIn,FRAME\_SIZE);**

**Location:** “Source\_Tree\user\libs\peripherals\adc\src”

**Description:** given an initialized handler, the sample size and the input source; this method will quantify and store the data in the selected buffer.

**Dependencies:**

#include "..\h\ADCChannelDrv.h"

1. **VOID fourierTransform(FRAME\_SIZE,compX,frctAudioIn)**

**Location:** “..\inc”

**Description:** given the frame size and the input variable; the method computes the FFT and stores it in the variable **compX.**

**Dependencies:**

#include "..\inc\transform.h"

#include <dsp.h>

1. **filterNegativeFreq(FRAME\_SIZE,compXfiltered,compX)**

**Location:** “..\inc”

**Description:** given the frame size and the input variable; the method filters the negative part of a signal in the frequency domain and stores it in **compXfiltered.**

**Dependencies:**

#include "..\inc\transform.h"

#include <dsp.h>