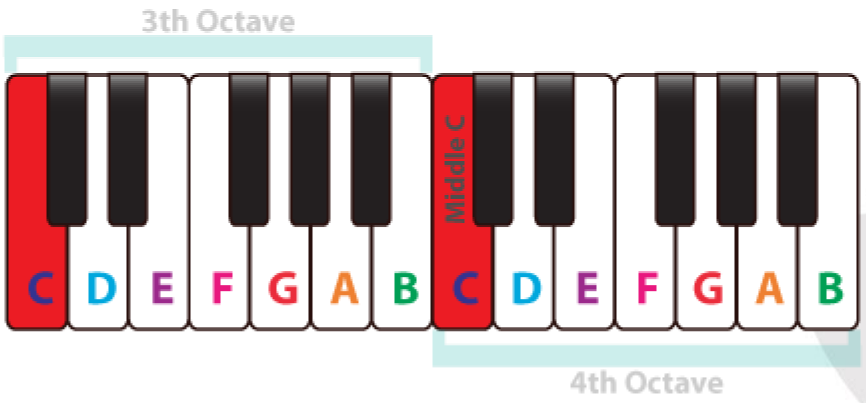
Faculty of Information Engineering and Technology
Department of Communication Engineering

Prof. Dr. Ahmed El-Mahdy

# Signals and Systems Theory (COMM401)

## Lab Project (PIANO) Audio Processing



German University in Cairo
الجامعة الألمانية بالقاهرة

# Project Description

- This project is classified into 2 milestones:

Milestone 1: You are the PIANIST, you should mix and match frequencies to generate your own song.

Milestone 2: Noise Cancellation and Frequency Domain (will be posted later after the midterms).

# Project Guidelines

- Groups of **TWO** students MAX (From the same Lab group, NO cross labs allowed)

- Copied reports or code means **zero** for both versions !

- At the end of each milestone you should submit:

Code: Your Python script/s
Report: A report containing a brief description of what you have implemented followed by the output figure/s obtained

•Submissions will be e-mailed to your TA through
•Subject: Tutorial Number and in the Body write your group members name:
Eng. Menna Saleh: signalslab.mennasaleh@gmail.com
Eng. Nouran Zaghlool: signalslab.nouranzaghlool@gmail.com
Dr. Sarah Azzam: signalslab.sarah@gmail.com
Eng. Maha El-Feshawy: signalslab.maha@gmail.com
Eng. Maggie Shammaa: signalslab.maggie@gmail.com

## Background

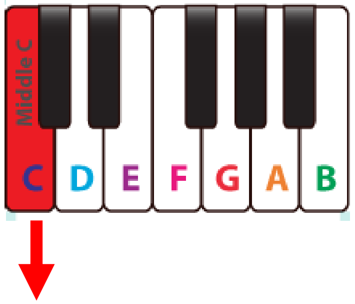
- Single tone generation:

Each pressed piano key/note number $i$, corresponds to a single tone with frequency $f_i$ generated for a certain period of time $T_i$starting from time $t_i$ over a defined time range $t$.

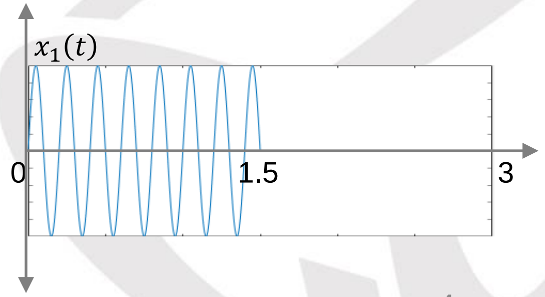
$$x_i(t) = \sin(2\pi f_i t)\ \ [u(t - t_i) - u(t - t_i - T_i)]$$

Where the unit step functions defines the playing interval.

Example: Assume that we started pressing the middle C key ($f_1 = 261.63\ Hz$), at time $t_1 = 0$ for a duration of $T_1 = 1.5$ seconds, the resultant signal is:



$$x_1(t) = \sin(2 * 261.63\pi t)\ [u(t) - u(t - 0.5)]$$



# Milestone 1: The Pianist

- Signal/Song Generation:

Generate your own signal using $N$ pairs of notes chosen from the 3rd and 4th piano octaves assuming that you are playing will both hands at the same time (3rd octave with the left hand/ 4th octave with right hand),

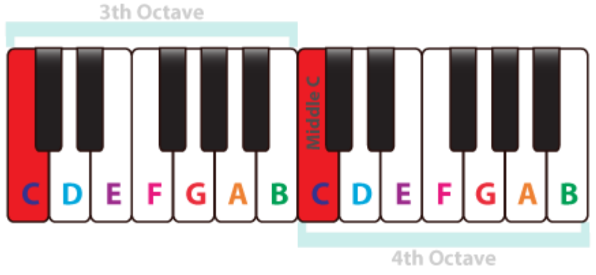
Each piano key/note frequency is given below. Your final song could be generated as follows,

$$x(t) = \sum_{i=1}^{N} [\sin(2\pi F_i t) + \sin(2\pi f_i t)]\,[u(t - t_i) - u(t - t_i - T_i)]$$

Chosen from the 3rd octave

Chosen from the 4th octave

| Note | Frequency | Note | Frequency |
|---|---|---|---|
| C3 | 130.81 | C4 | 261.63 |
| D3 | 146.83 | D4 | 293.66 |
| E3 | 164.81 | E4 | 329.63 |
| F3 | 174.61 | F4 | 349.23 |
| G3 | 196 | G4 | 392 |
| A3 | 220 | A4 | 440 |
| B3 | 246.93 | B4 | 493.88 |



# Milestone 1: The Pianist

- Procedure:

1) You will need to import the following libraries.

```
import numpy as np
import matplotlib.pyplot as plt
import sounddevice as sd
```

2) Set the total song playing time to $3$ seconds starting from $0$ for $12 \times 1024$ samples.

$$t = \text{np.linspace}(0\,,3\,,12 * 1024)$$

3) Customize your own song by setting the number of pairs of notes $N$. For each pair number $i$, set the left hand frequency $\mathrm{F}_i$, right hand frequency $f_i$, the pressing starting time $t_i$, and how long you will press both keys $T_i$.

N.B you can freely customize these values to play the song you want.

If you want to play with one hand only, you can set the frequency of the other hand to 0.

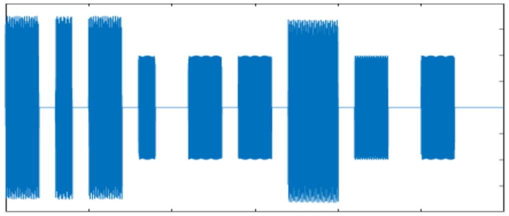
# Milestone 1: The Pianist

- Procedure:

4) Define your signal/song using the previous parameters:

$$x(t) = \sum_{i=1}^{N} [\sin(2\pi \mathrm{F}_i t) + \sin(2\pi f_i t)]\,[u(t - t_i) - u(t - t_i - T_i)]$$

5) Plot your signal/song in the time domain your output will look similar to the following: $\mathrm{plt.plot(t, x)}$



NB. The time separation between the notes will vary from one group to another according to your song. We also can't notice the variations in the sinusoidal tones due to the high frequencies.

6) Finally: Play your song using:

$\mathrm{sd.play}(\mathrm{x}, 3 * 1024)$

GOOD LUCK