DIGITAL SIGNAL PROCESSING

SEMESTER COURSE PROJECT:
VOICE RECORDING AND PLAYBACK

TASK: RECORD A SHORT AUDIO CLIP, APPLY BASIC PROCESSING, AND THEN PLAY BACK THE MODIFIED SIGNAL.

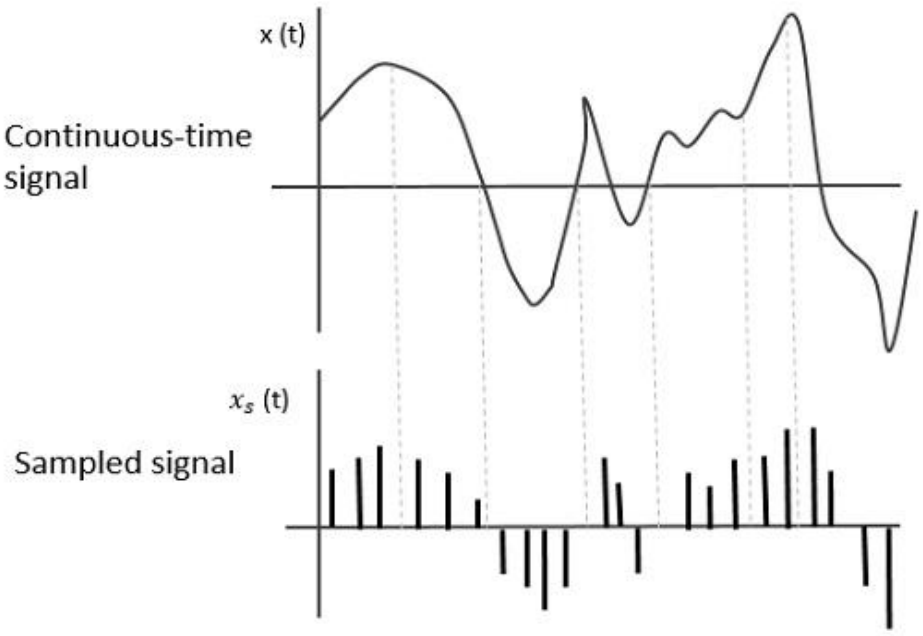
Group members…….ID NO.
Fiseha Asteraye……UGR/ 0337/13
Taklu Birranu……UGR/6538/13
Ephrem Woldemichal…UGR/3784/13

Submitted to:Belayneh Melka
Date:31/01/2023

***Recording a Short Audio Clip*:**

Use the ***audiorecorder*** object to record a short audio clip for 10 seconds.
In this step:
- We create an ***audiorecorder*** object and record a short audio clip for 10 seconds.
- The recorded voice data is acquired using the ***getaudiodata function***, and the sampling frequency (fs) is obtained from the recorder object.

***Sampling (Analog-to-Digital Conversion)* :**



When we record audio, we're converting a continuous *analog* signal into a *digital* format, a process known as sampling. When we record in MATLAB, this conversion is taken care of automatically as the audio is stored as digital data in the ***voiceData*** variable.

Frequency analysis and ploting:

Frequency analysis:

The frequency analysis is performed using the ***Fast Fourier Transform (FFT)***. The FFT provides a way to analyze the frequency content of a signal, allowing us to see which frequencies are present and their respective amplitudes. This is important for understanding the spectral characteristics of the recorded audio signal.

For instance in the Matlab code attached,

nfft defines the number of data points used for the FFT computation.

- **Y** stores the result of the FFT, representing the frequency domain signal.
- **f** is used to create the frequency range to align with the FFT result.

Plotting the recorded signal:Plotting the Signal

We visualize the time-domain and frequency domain characteristics of the audio signal through plots

t-represents the time vector for the unsampled signal. We create a subplot for visualizing the unsampled audio signal.The same approach applies to the sampled and filtered signals as well.
By following these steps, we're able to perform frequency analysis and create visual representations of the audio signal in both the time and frequency domains

Processing and Conditioning :

I *Downsampling:*

In the provided code, downsampling is applied through the MATLAB function **downsample.** Downsampling reduces the sampling rate of a signal, effectively reducing the data and compressing the signal in the time domain.
We employed the downsampling factor of 4.
In the code attached ,**downsampleFactor** indicates the factor by which the signal is down-sampled.
-**voiceData** captures the downsampled audio signal.
-**fs_downsampled** represents the updated sampling frequency after downsampling.

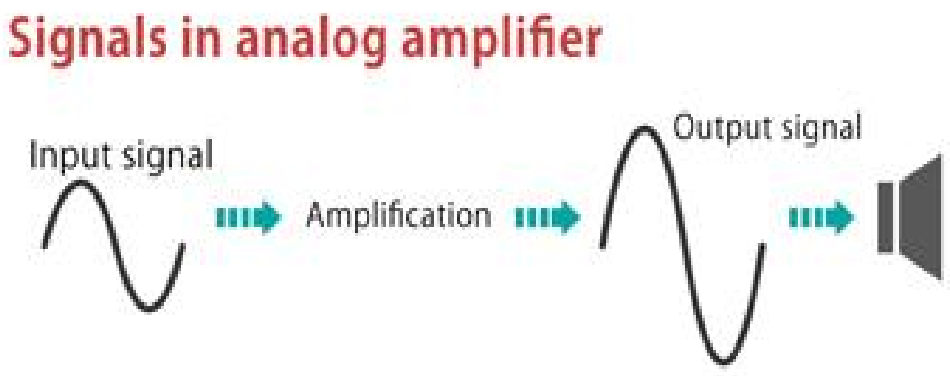
II Noise Filtering:

Apply a simple low-pass filter to suppress high-frequency noise in the recorded audio.

Here, we design a Butterworth low-pass filter by specifying the cutoff frequency and order. Then, we apply this filter to the amplified audio signal using the filter function.

we utilized the **butter** function to design a low-pass filter. Here's a breakdown of what's happening:
- cutoffFreq specifies the desired cutoff frequency for the filter. This is set to 3000 Hz in the code.
- The butter function constructs a low-pass Butterworth digital filter by taking in the filter order (6 in our case), the normalized cutoff frequency (expressed with respect to the Nyquist frequency=fs/2), and the filter type ('low' for a low-pass filter).

III Amplify the Audio Signal :



Amplify the recorded audio signal to increase its volume level.
Here, we simply amplify the recorded audio signal by multiplying it with a constant factor (10 in this example) to increase its amplitude.
amplifiedVoice = 2 * filteredVoice;
Hence, the amplitude of the signal is multiplied by 2(doubled).

IV Reconstruction(Digital-to-Analog Signal):

The reconstruction process involves converting the filtered digital signal back into an analog format, allowing for playback.

V Play Back the Modified Audio Clip:

Play back the processed audio using MATLAB's sound playback capabilities.
The function **sound( )** which takes the **amplifiedVoice** Object and the original sampling frequency **fs** Plays back the modified audio at the original sampling frequency.

Summarizing Diagram: