Voice Based Speaker Identification

Rahul Sharma (09D26008)
Prashant Srivastava (09D01021)
Saurabh Gandhi (09026012)

Objective:

The idea was to identify a speaker based on the sound of his/her speech

Theory:

A few key terms would be used, such as a vowel, and the Fast Fourier Transform of the speech sound would be obtained. Then the frequency will be mapped onto a scale which is most apt for human voice recognition. This frequency scale is called a mel or a melody scale. The relation for converting FFT to mel scale is:

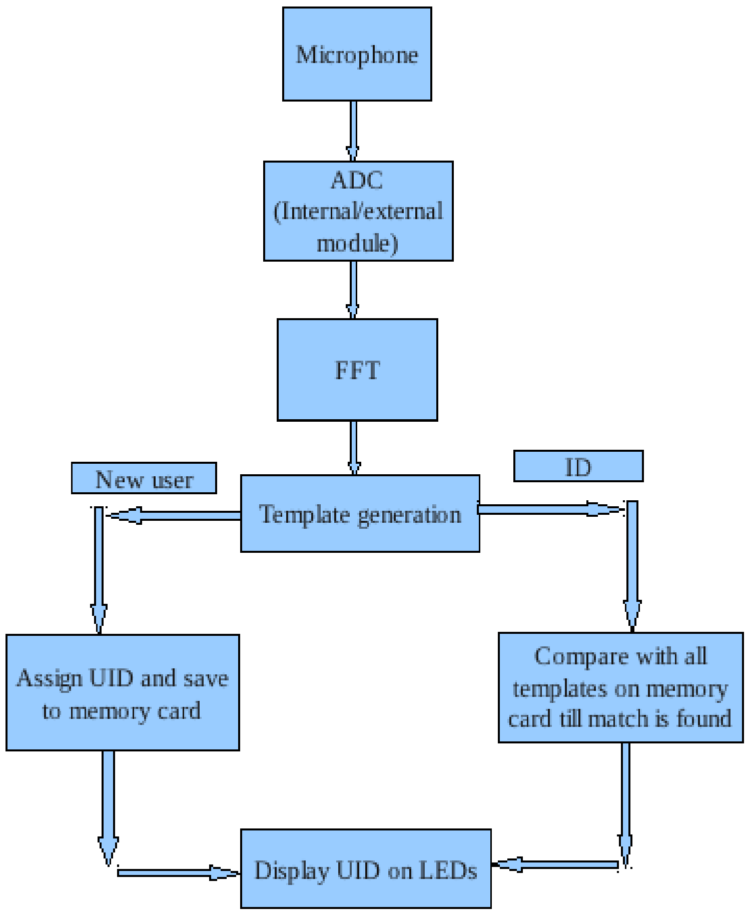
$$m=2595\log(1+f/700)$$

Once we get the mel frequency components, we calculate mel frequency cepstral coefficients or MFCCs. A **cepstrum** is the result of taking the Fast Fourier Transform (FFT) of the logarithm of the spectrum of a signal.In sound processing, the **mel-frequency cepstrum** (**MFC**) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.The difference between the cepstrum and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum.

The MFCC algorithm returns a feature extracted from the power spectrum of the mel frequency. This feature can be compared directly or if the size is large, vector clustering can be performed on it using algorithms such as the k-means algorithm and then compared. For comparision, we compute the euclidean distance between the template vectors and the voice closest to the stored template is returned. Thresholding may be performed for giving specific positive/negative result.

Ideal Implementation:

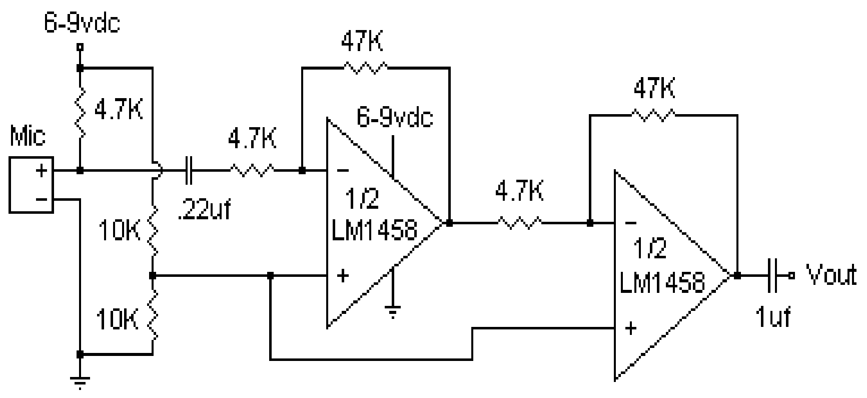
Block diagram:


Microphone
ADC (Internal/external module)
FFT
New user
Template generation
ID
Assign UID and save to memory card
Compare with all templates on memory card till match is found
Display UID on LEDs

Electronics :

We used a microphone to capture the sound and a two stage amplifier and filter to enhace the signal.

Electret Condenser Microphone
Bias Circuit and Preamp



Programming:

We used a MATLAB open source toolbox called **Voicebox** for implementing the theory mentioned above. It takes a .wav file as input. It then calculates the FFT, maps it onto the mel scale and then finds the MFCCs, using Hamming windows for sampling. Optimum sampling frequency of 16 kHz was used. The output vector was reduced in size by K-Means clustering. Finally the euclidean distance between the sample and previously stored templates was evaluated and a descision on whose voice is most closely resembled was reached.

This was to be implemented on the arduino.

Difficulties in Achieveing the goal:

Converting MATLAB code to C: C is a very basic language. Main problem was that MATLAB uses many kind of data structures and its function operate on those structures. So to write an equivalent function was difficult even for the simplest of MATLAB functions. Despite these difficulties we partly converted the code into C till we hit another roadblock!

Memory Limitations: It had extreme memory requirements and sometimes array sizes of around 50000 elements were required! So we decided to use external memory elements like EEPROM. We tried but were unable to get it to work. We realised after converting the code partially that even the code is going to be too big for arduino. We implemented an FFT algorithm on arduino. However, due to memory constraints we had to limit to a low resolution.

Parts Implemented:

The signal was captured using ADC on the arduino board. Then a 16 point radix-4 FFT algorithm was used to obtain the fourier coefficients of the captured signal. Sampling was done according to the Nyquist criteria at 4 kHz (every 250 microseconds). We got real and imaginary parts of 8 frequency components which were sampled in real time. We then calculated the normalised magnitude of all the frequency components and displayed it in the form of a histogram on serial monitor.

Storage of tempelates on sd-card was also implemented but was not required since tempelates could not be generated due to above mentioned difficulties.

The MATLAB code, too, was partially converted to C, as mentioned above, but could not be used further.

Result:

We have managed to identify the relative magnitudes of audio signals and display it graphically. This can be used directly as a guitar tuner, or in general, for identifying major frequency components in any signal. However, for applications requiring high resolution sampling and FFT (such as voice identification), this device cannot be used primarily due to size constraints.

Work Division:

Hardware:

- Microphone pre-amplifier: All of us

Software:

- Actual coding: All of us
- FFT Algorithm: Prashant
- MATLAB to C: Rahul
- SD-CARD: Saurabh

References:

- http://www.retroleum.co.uk/electronics-articles/basic-mmc-card-access/
- http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html
- http://www.mit.edu/~emin/source_code/fft/index.html
- Digital Signal Processing, Proakis and Manakolav