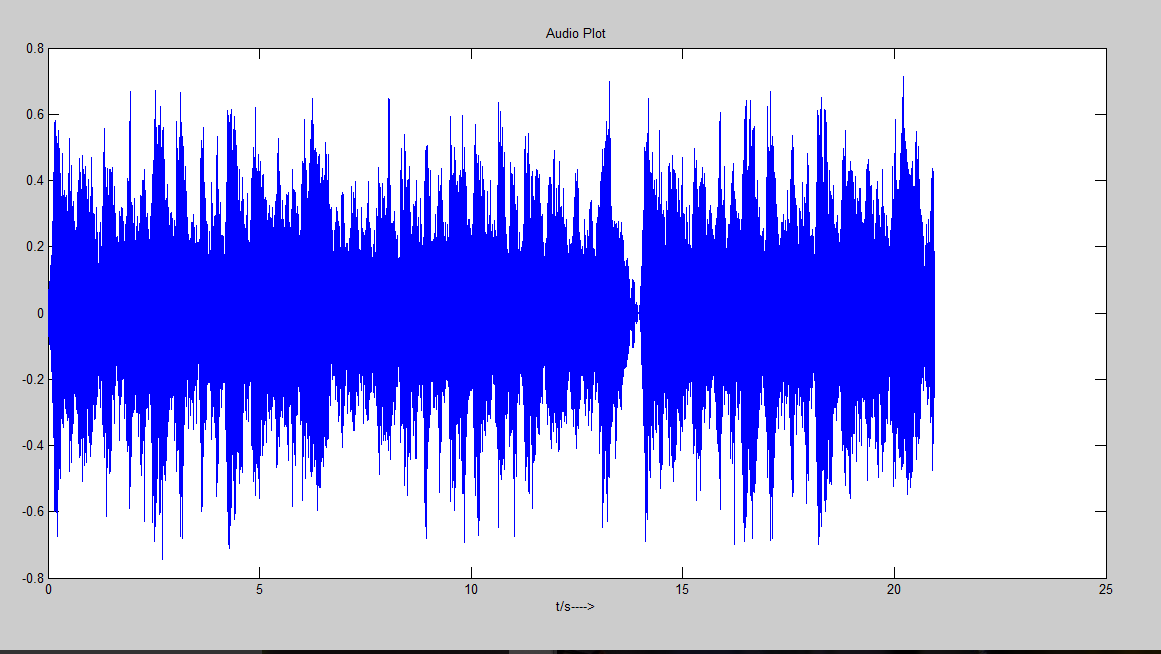
**Abdullah Farooq**

**Digital Signal Processing**

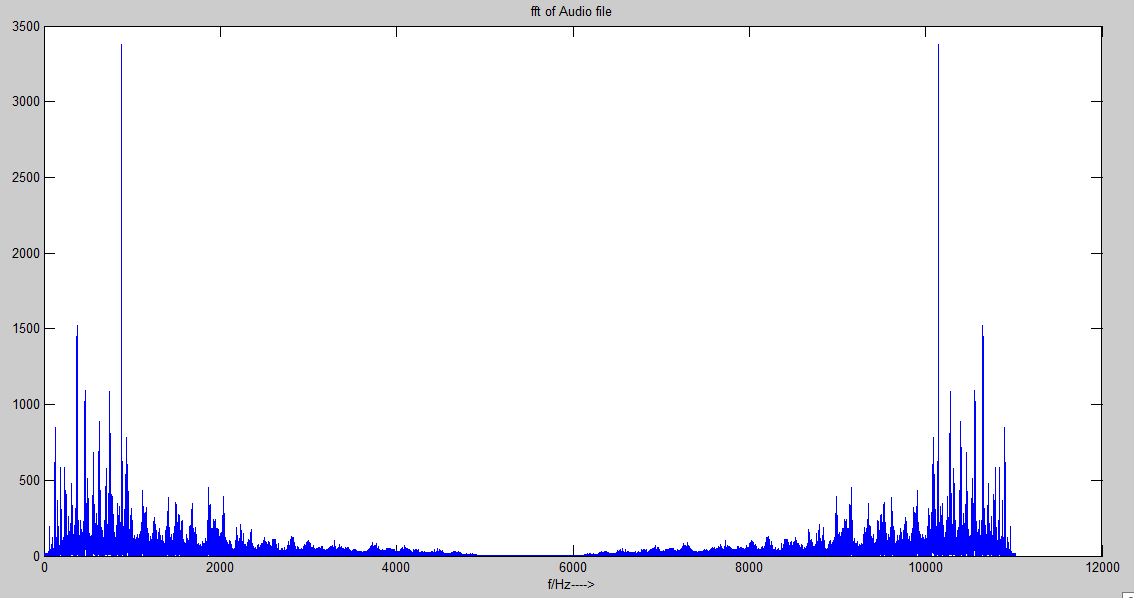
**Project Report**

**University of Houston**

First, the MATLAB code reads the audio file “ECE6342\_morse\_code\_sound.wav” in a variable y. Then I plotted this audio signal in the time domain using command plot (t,y) with t=[1/fs:1/fs:L/fs]; That is the starting value of t is 1/fs which is almost zero and values of t are taken at 1/fs intervals so that we have a large number of values and the final value of t is L/fs with L being the length of signal y. L/fs comes out to be almost 21s (the duration of the audio signal). Its plot is shown below:

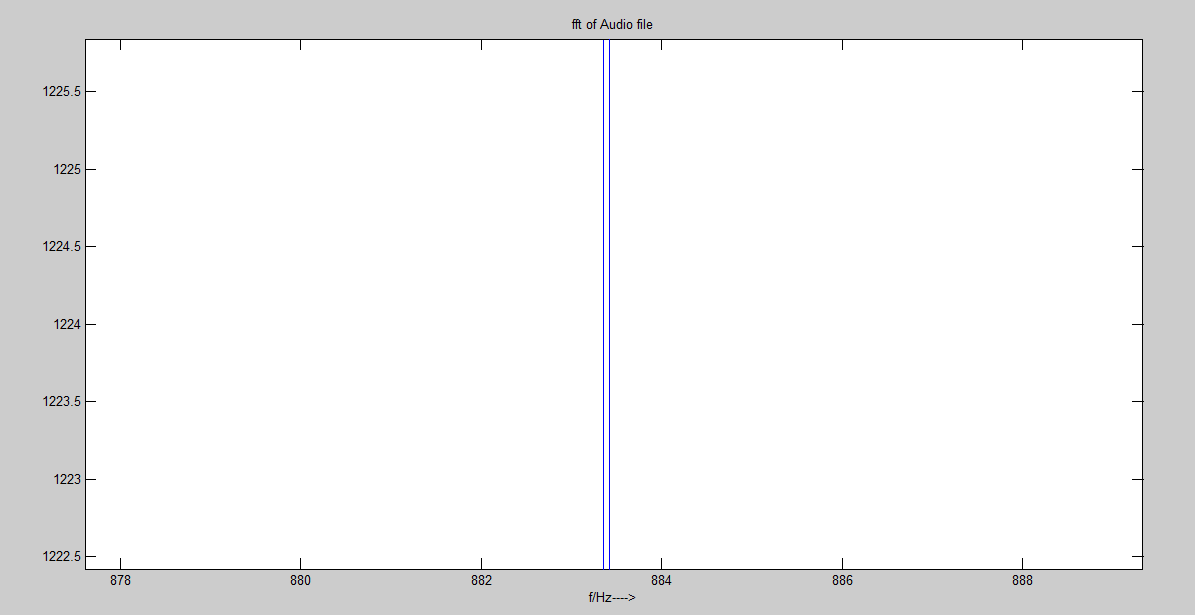


Its fft plot is:

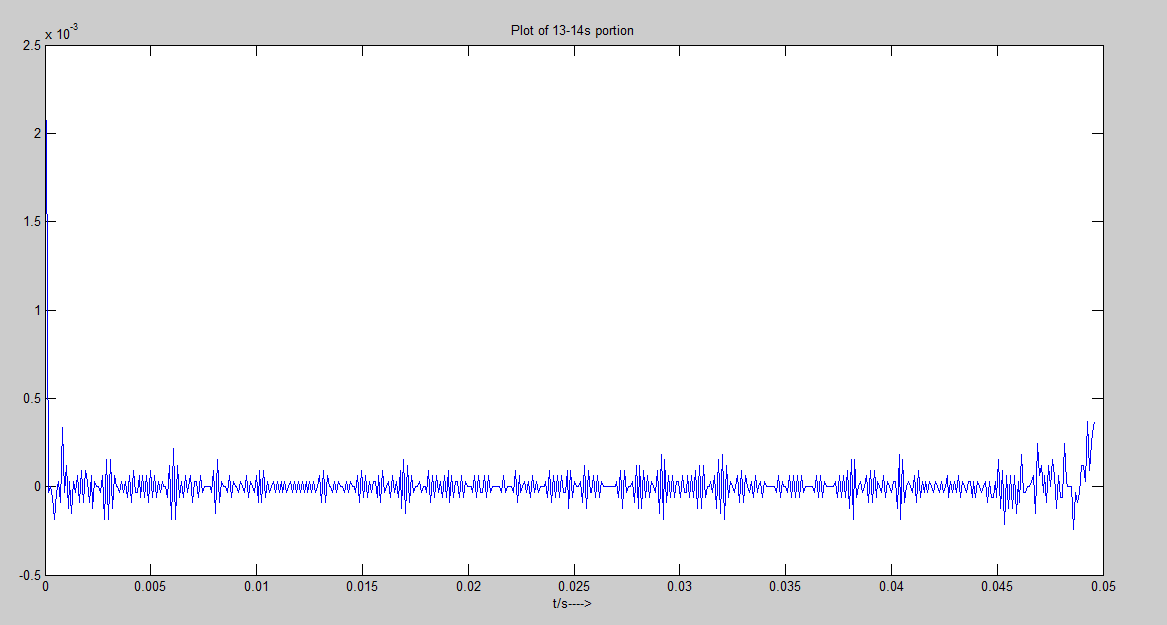


From this and by listening to the audio I noticed that the audio is silent between 13-14s and only morse code is there during this time so I zoomed in at the exact values and obtained the values of 13.923s and 13.9725s during which only morse code is present and separated this signal from the rest of the audio and named it x.

I found the morse code frequency by observing the peak value in the above graph which corresponds to the morse code frequency and when I zoomed in I obtained a value of 883Hz:



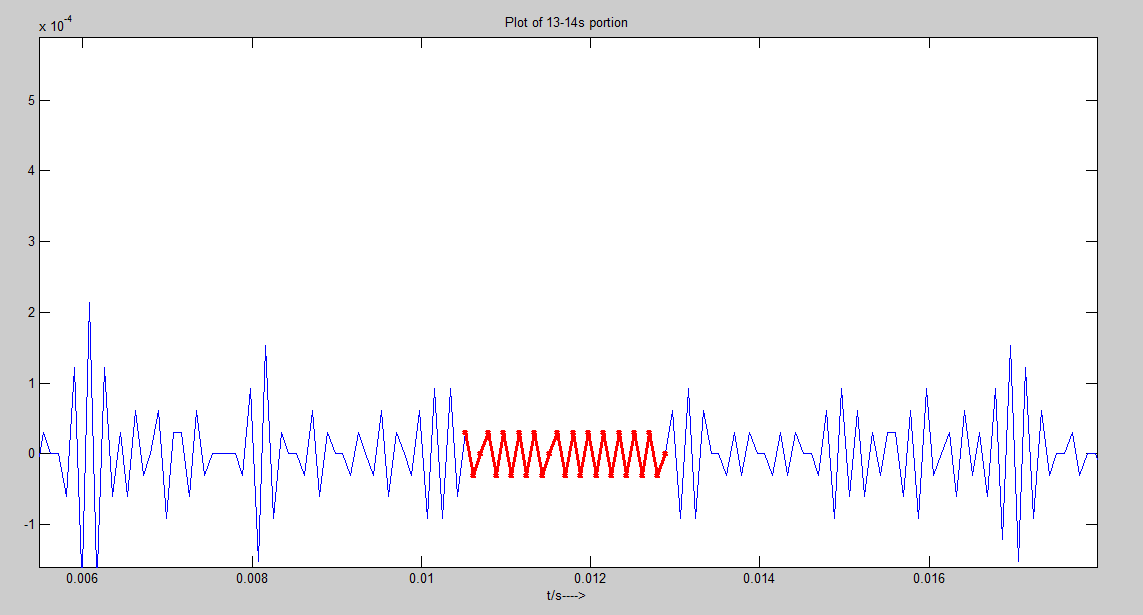
This graph is the same as the first fft graph except that it is zoomed in at the exact value. The morse code frequency can also be found out from the zero crossings between 13-14s region specifically from 13.923s to 13.9725s:



If we count the number of zero crossings (# of time signal changes sign) in this signal and reject the values that are too close to zero, the number of zero crossings obtained are approximately 97. Then the number of cycles are 97/2=48.5

Now in 13.9725-13.923=0.0495s 48.5 cycles are completed then in one second 979 cycles would be completed which is the frequency that is close to 883 value we got before.

Some of the values which have been rejected as they are too close to zero (as they represent noise) in this approach are shown in red in the graph below which is the zoomed in version of the graph above:

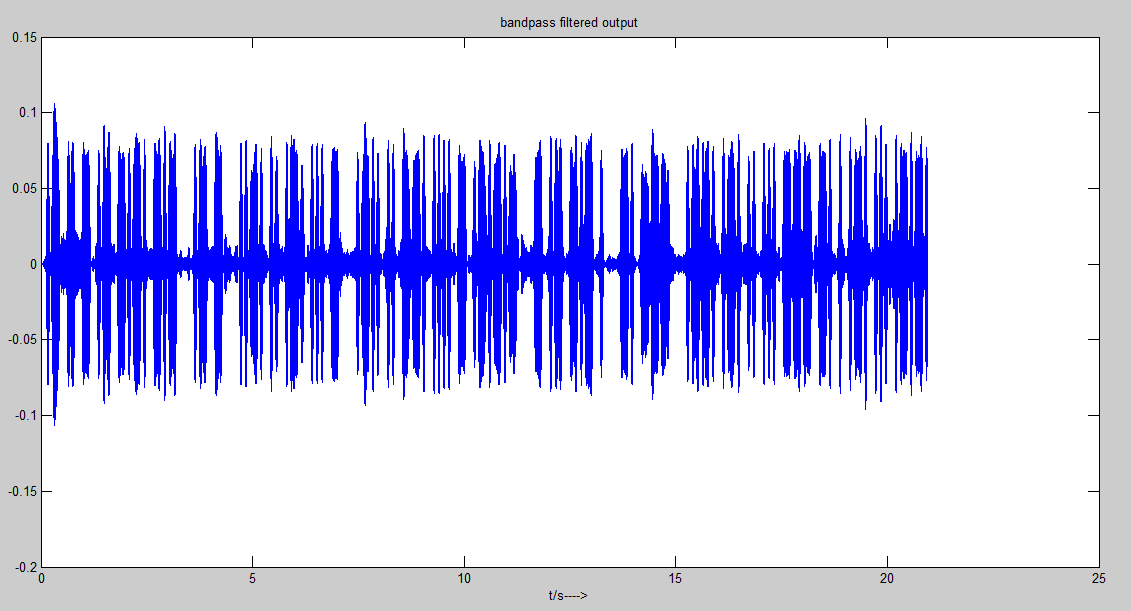


I chose the value 883Hz as it is more accurate as compared to zero crossings approach.

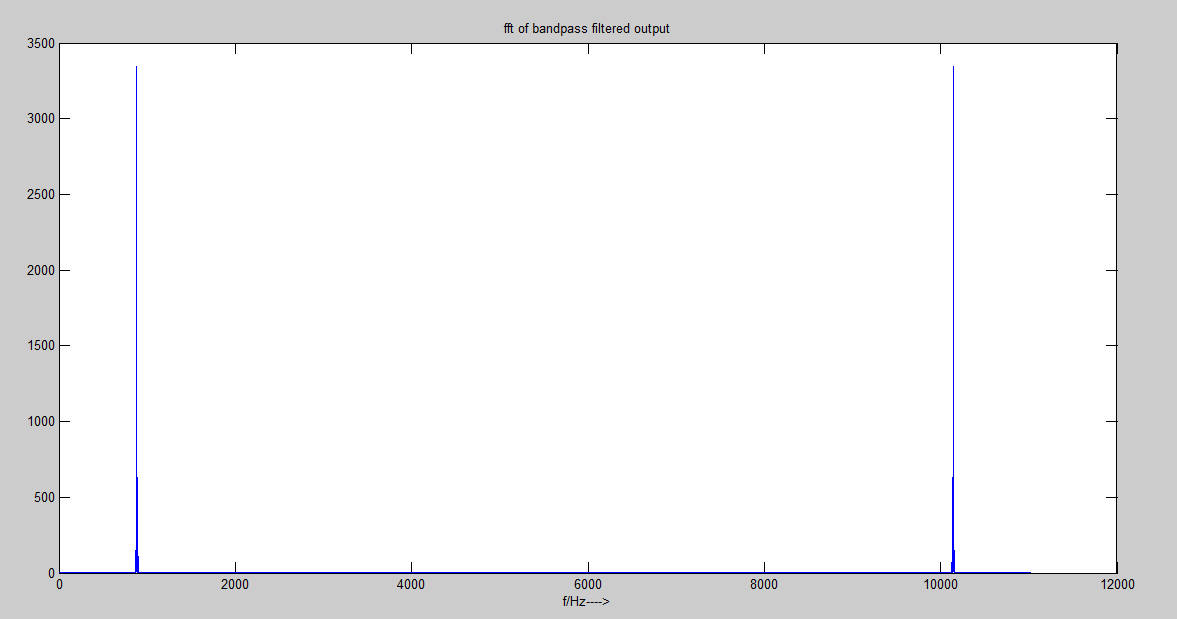
Next I made a bandpass filter with order n=3000 and chose the lower and upper frequencies as

883-15=868Hz and 883+15=898Hz and used fir1 command which is MATLAB’s built in command for any type of filter you specify. Then I passed the original audio signal y through this bandpass filter that was made by fir1 command by using a command called filter to obtain a signal called f2 which is the morse code. In the fir1 command I have used values of 2\*868=1736 and 2\*898=1796 that is double the frequency around which I wanted to build a bandpass filter but that is only due to the structure of fir1 command which takes normalized frequencies and it takes frequencies between 0 and 1 and I have divided these by fs to make them normalized frequencies but eventually our frequencies are 868Hz and 898Hz. Next I have used the same frequencies and the same function to make a bandstop filter to obtain our morse code free audio signal named f3 in my MATLAB program.

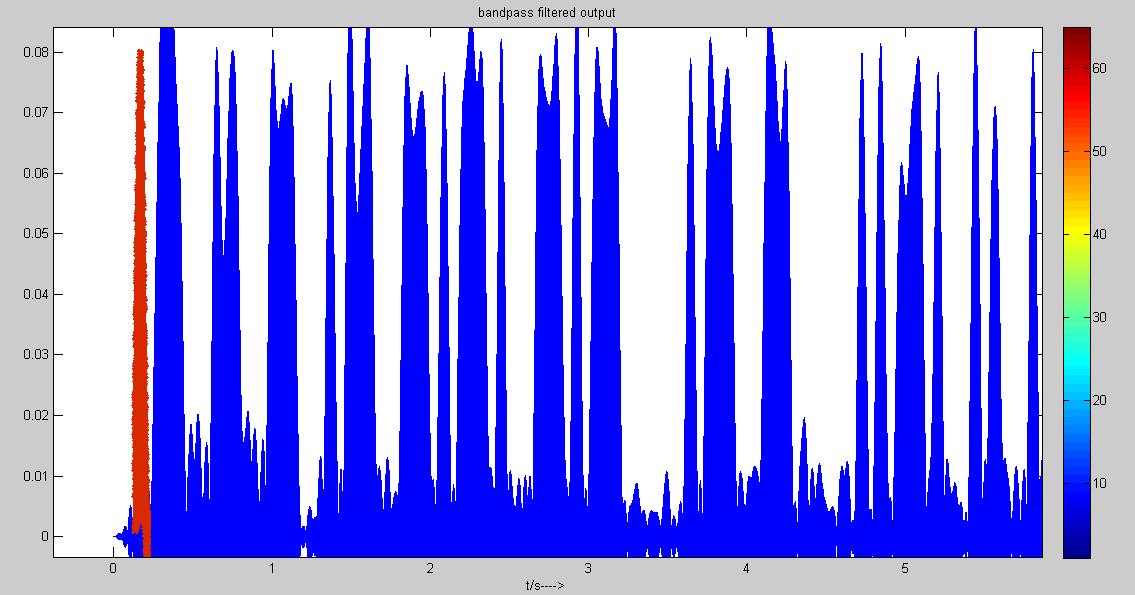
The following is a graph of the bandpass filtered output which is the morse code:



fft of bandpass filtered output graph:



Next I decoded this bandpass filtered output by looking at the graph below which is the zoomed version of the above graph (bandpass filtered output). The red peak represents one time unit and is therefore a dot. The peak right next to it is a dash which is three time units and we can clearly see that it’s width is greater than the ‘dot’ peak. And right next to it are 3 spaces which represent space between letters in a word and its width is also 3 time units. I found out the length of one time unit by analysing the last few peaks which were mentioned in the Project as dot dash dot dash dot. From there I found out approximately what will be the length of dot and what will be the length of dash.



One word is approximately completed by 3.5s which is why we see 7 spaces at that time which show that one word has been completed. Following these steps till the end I obtained the following sequence:

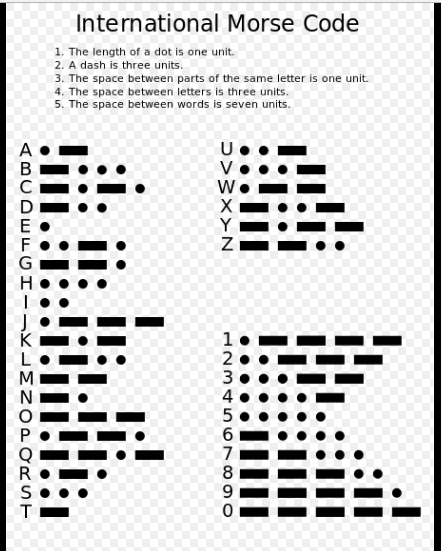
**.-///-///-///.-///-.-.///-.-///////.-///-///////..-.///..///.-.///...///-///////.-..///..///--.///....///-///.-.-.-///////-///.-///-.-///.///////-.///---///////.--.///.-.///..///...///---///-.///.///.-.///...///.-.-.**

Where ‘.’=dot

‘-’=dash

‘/’=space

I obtained the decoded message by looking at the following table from Wikipedia:



First is dot dash which is equal to ‘A’

Then there are 3 spaces which means now there will be a second letter which is ‘dash’ which is equal to ‘T’

So by following this I obtained the decoded message:

A T T A C K A T F I R S T L I G H T. T A K E N O P R I S O N E R S <AR>