Alexandria University

Faculty of Engineering

Electronics and Communication Department

Mini Project II

Communication System

By

Ahmed Sameh Taha 21010083
Karen Mostafa El-Bardan 21010966
Amira Mohamed Mokhtar 21010308
Rewan Gamal AbdEl-Kader 21010540

Supervised by:

Dr. Mohamed Moselhy

Eng. Ahmed Mostafa

Eng. Esraa Ragab

The project aims to implement a very simple communication system to send a sound file over a channel, then adding noise to it and finally, the sound is received by the receiver,

The components of the system:

- Transmitter
- Channel
- Noise
- Receiver

1) Transmitter:

In this stage, the signal is entered and prepared to be transmitted over the channel

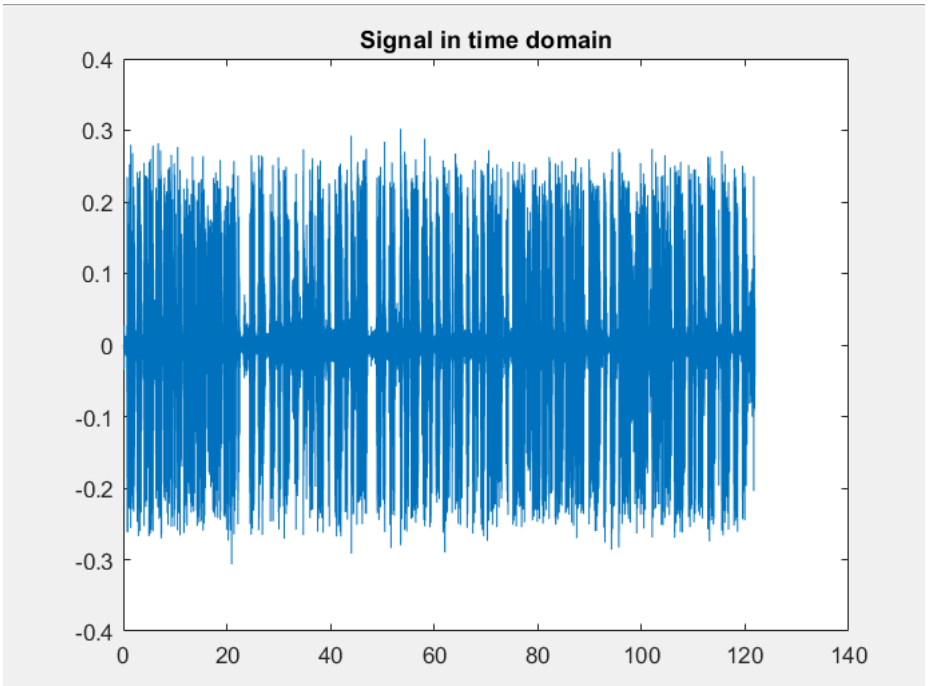
Procedures:

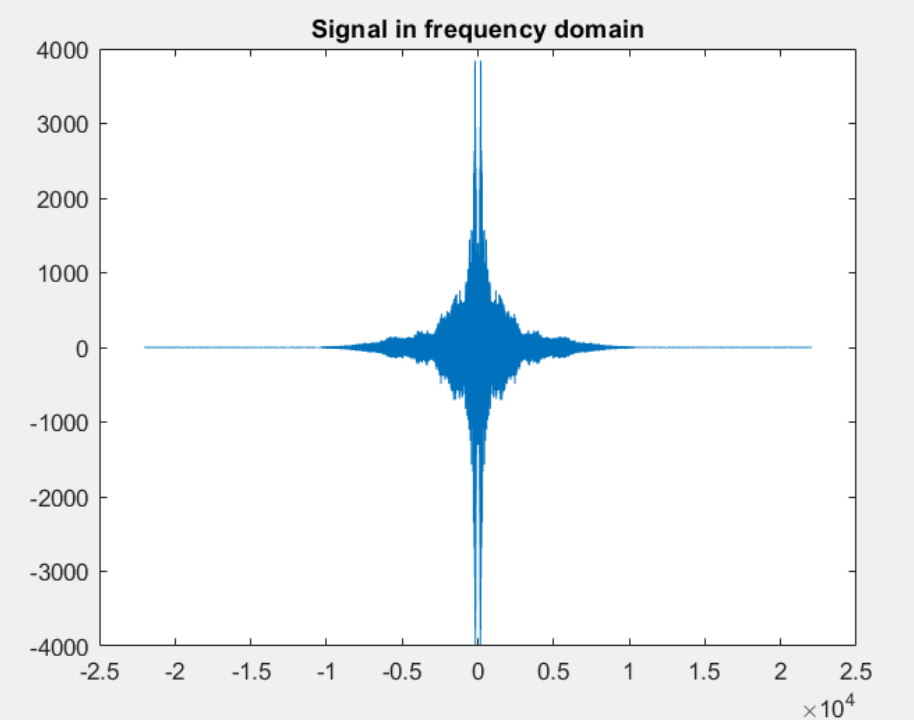
- The sound is imported and played
- The sound file was plotted in time domain and the frequency domain

Code:

```
[y,fs] = audioread('Sound.mp3');
y = y(:,1) ;
sound(y,fs);

t = linspace(0, length(y)/fs, length(y));

figure;
plot(t, y);
title('Signal in time domain');

f=linspace(-fs/2,fs/2,length(y));
y_freq=real(fftshift(fft(y)));
figure;
plot(f,y_freq);
title('Signal in frequency domain');
```

Output:


Signal in time domain
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
0
20
40
60
80
100
120
140


Signal in frequency domain
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
×10^4

2) Channel:

Sound:

The channel has the following impulse response. We will need to pass sound message over the channel

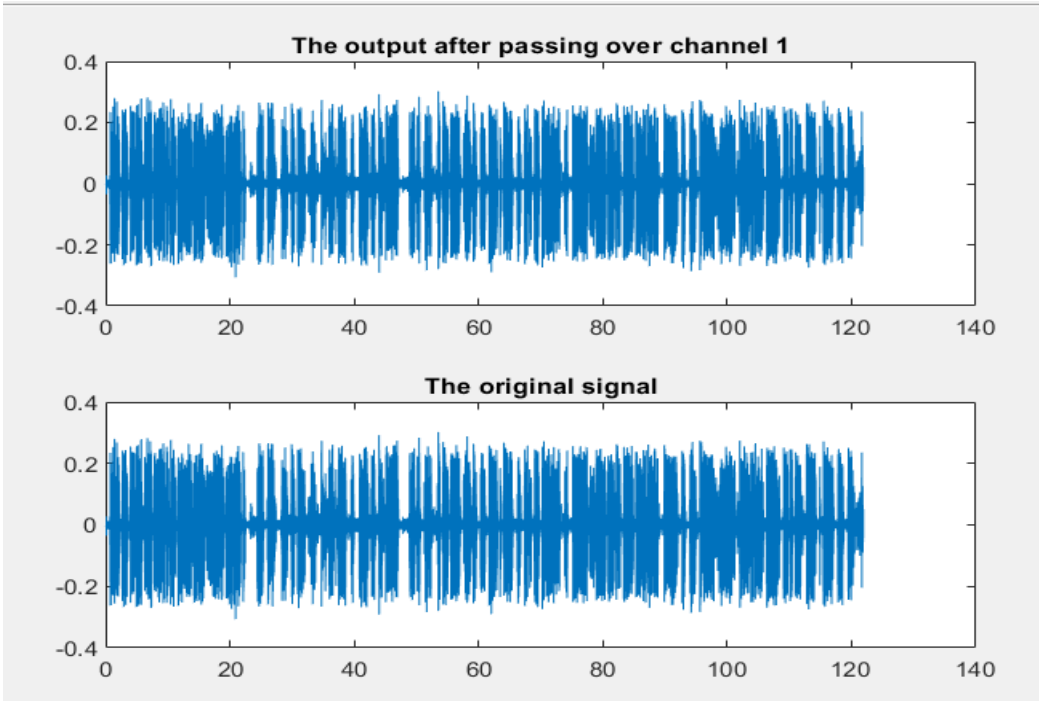
We 4 options for the channel impulse response:

1. Delta function:

Code:

```
19 -    h1=[1 zeros(1,length(y)-1)];
20 -    y_1=conv(h1,y);
21 -    y_1 = y_1(t<=length(y));
22 -    figure;
23 -    subplot(2,1,1);
24 -    plot(t,y_1);
25 -    title('The output after passing over channel 1');
26 -    subplot(2,1,2);
27 -    plot(t,y);
28 -    title('The original signal');
```

Output:

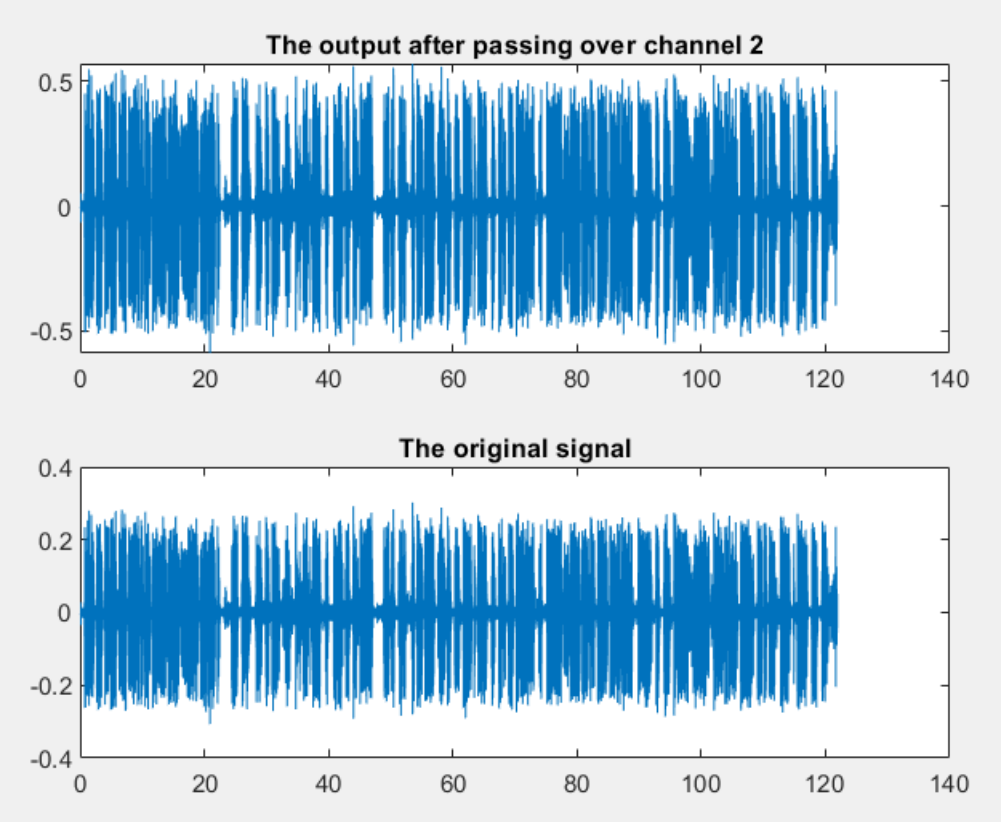


2. exp(-2pi*5000t):

Code:

```
30 -    h2=exp(-2*pi*5000*t);
31 -    y_2=conv(h2,y);
32 -    y_2 = y_2(t<=length(y));
33 -    figure;
34 -    subplot(2,1,1);
35 -    plot(t,y_2);
36 -    title('The output after passing over channel 2');
37 -    subplot(2,1,2);
38 -    plot(t,y);
39 -    title('The original signal');
```

Output:

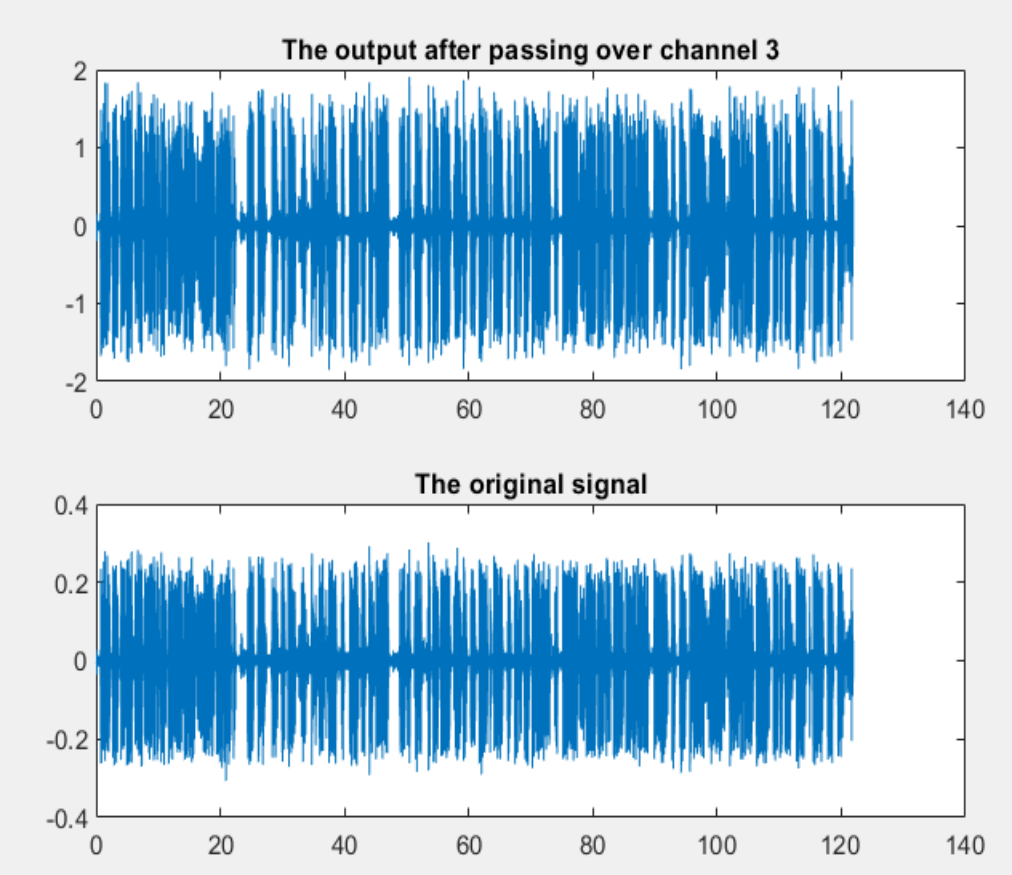


3. exp(-2pi*1000t):

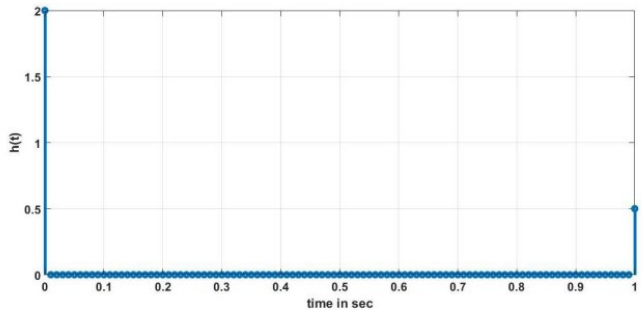
Code:

```
41 -    h3=exp(-2*pi*1000*t);
42 -    y_3=conv(h3,y);
43 -    y_3 = y_3(t<=length(y));
44 -    figure;
45 -    subplot(2,1,1);
46 -    plot(t,y_3);
47 -    title('The output after passing over channel 3');
48 -    subplot(2,1,2);
49 -    plot(t,y);
50 -    title('The original signal');
```

Output:



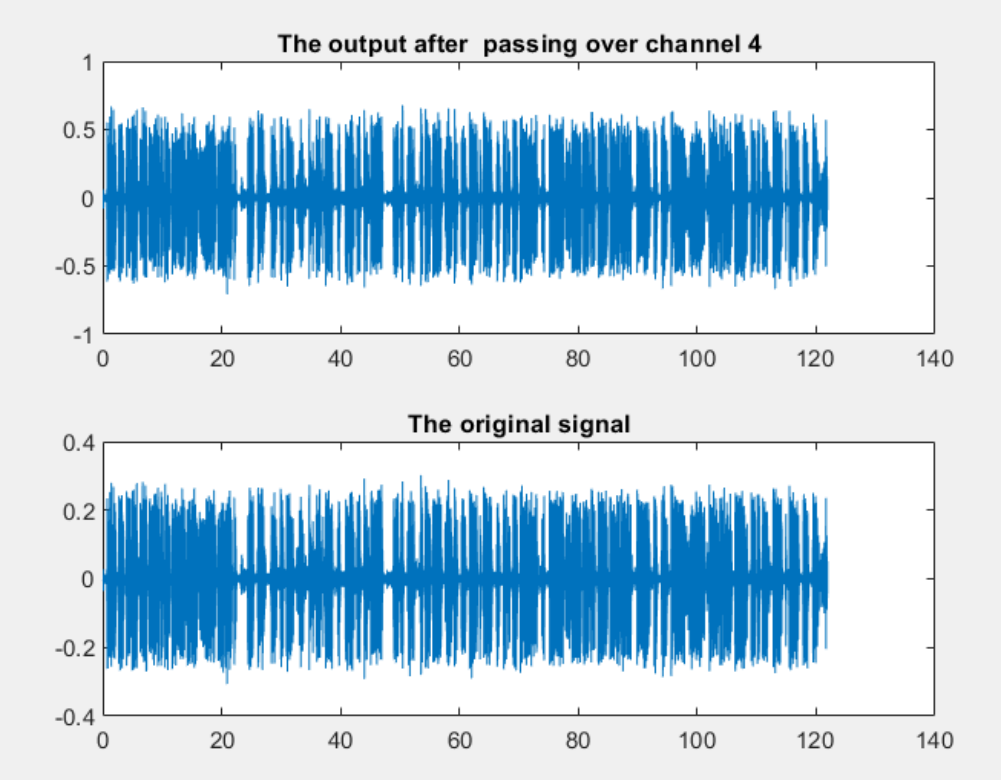
4. The channel has the following impulse response:



Code:

```
52 -    h4=[2 zeros(1,length(0:0.1:1)-2) 0.5];
53 -    y_4=conv(h4,y);
54 -    y_4= y_4(t<=length(y));
55 -    figure;
56 -    subplot(2,1,1);
57 -    plot(t,y_4);
58 -    title('The output after  passing over channel 4');
59 -    subplot(2,1,2);
60 -    plot(t,y);
61 -    title('The original signal');
```

Output:



3) Noise:

The program should have the ability to add noise (simply random signal) to the output of the channel

The random signal generation is done as following

Z(t) =sigma*randn(1,length(x))

Where x is a vector represents the output of the channel.

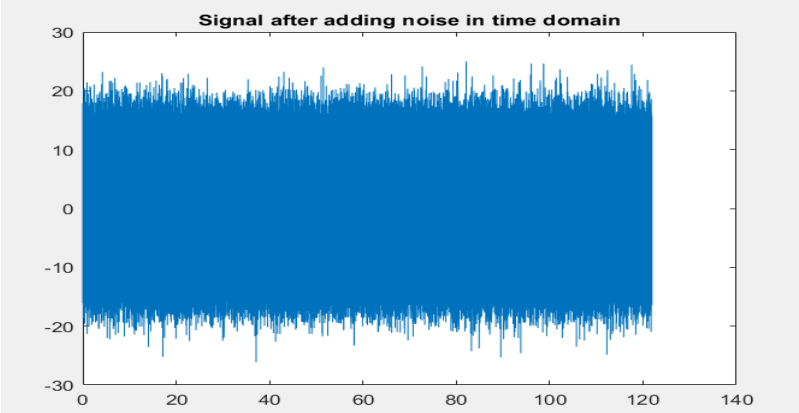
Code:

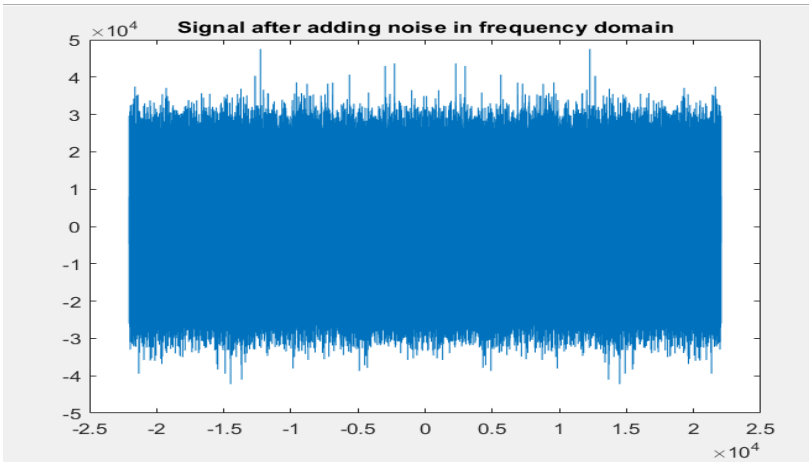
```
sigma = input('Enter the sigma value of the noise: ');
Channel = input('Enter the channel number that the noise will be added to it: ');
switch Channel
    case 1
        Z = (sigma*randn(1,length(y_1)))';
        y_noise = y_1 + Z ;
    case 2
        Z = (sigma*randn(1,length(y_2)))';
        y_noise = y_2 + Z ;
    case 3
        Z = (sigma*randn(1,length(y_3)))';
        y_noise = y_3 + Z ;
    case 4
        Z = (sigma*randn(1,length(y_4)))';
        y_noise = y_4 + Z ;
end
```

```
    sound(y_noise,fs);

    figure;
    plot(t, y_noise);
    title('Signal after adding noise in time domain');

    f=linspace(-fs/2,fs/2,length(y_noise));
    y_freq=real(fftshift(fft(y_noise)));
    figure;
    plot(f,y_freq);
    title('Signal after adding noise in frequency domain');
```

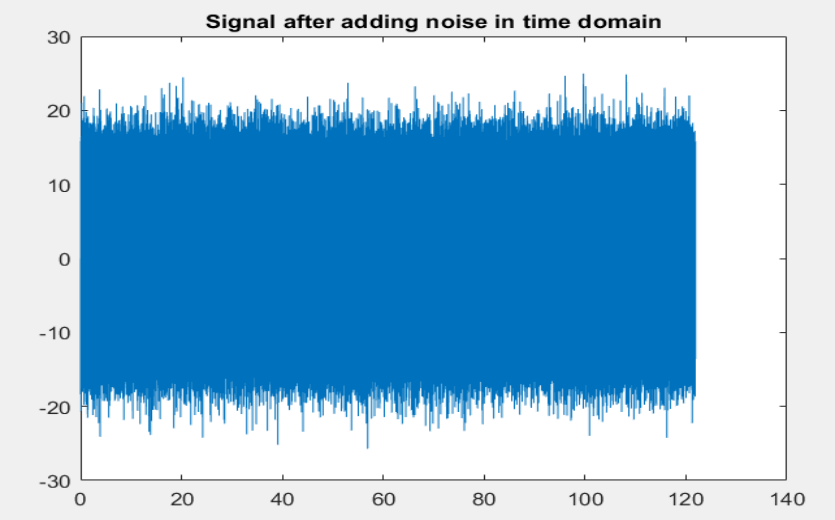
1. Delta function:

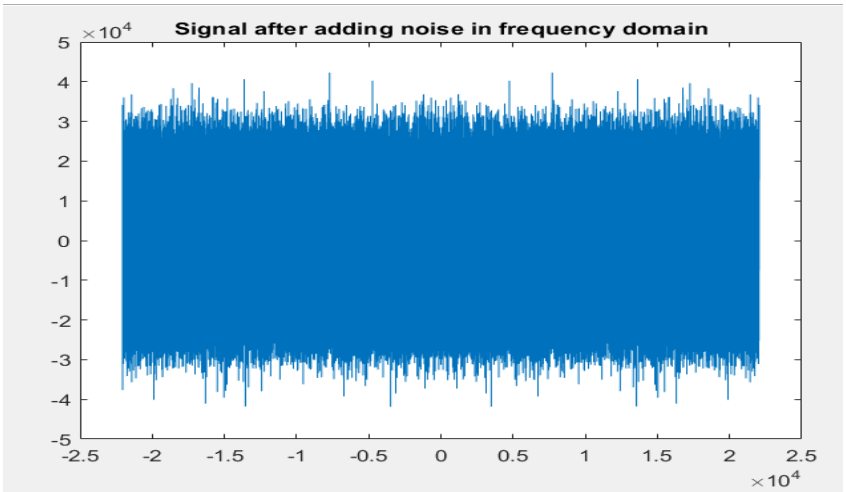
Output:



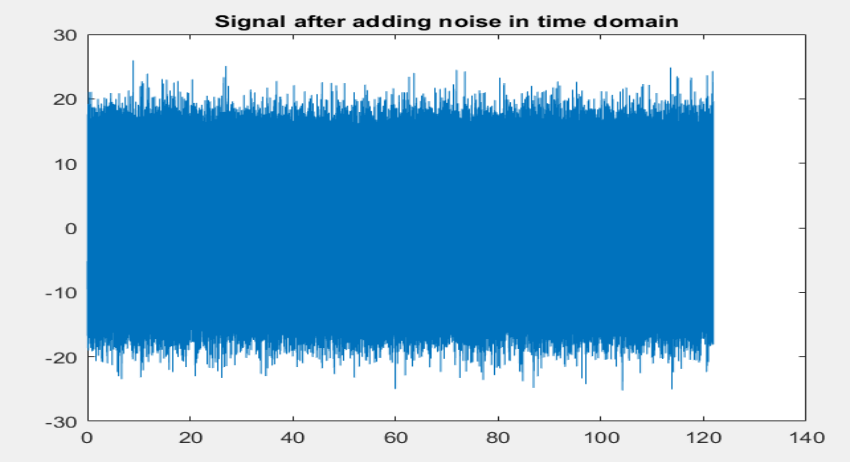


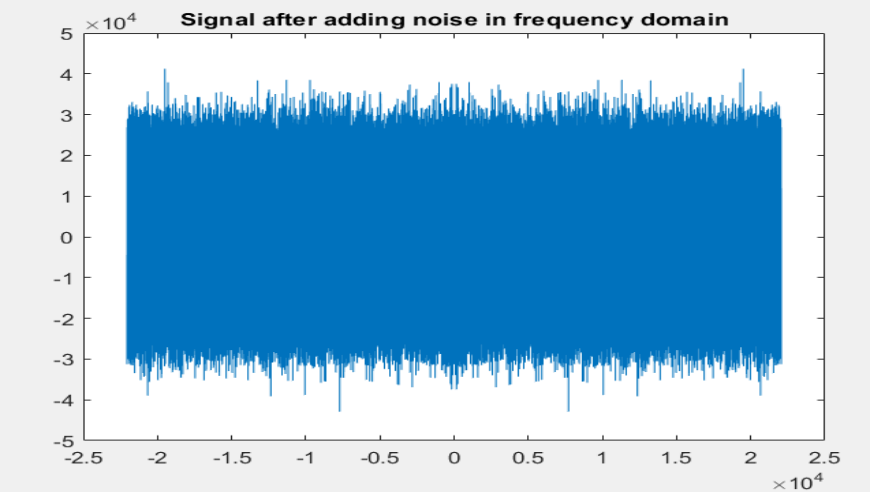
2. exp(-2pi*5000t):



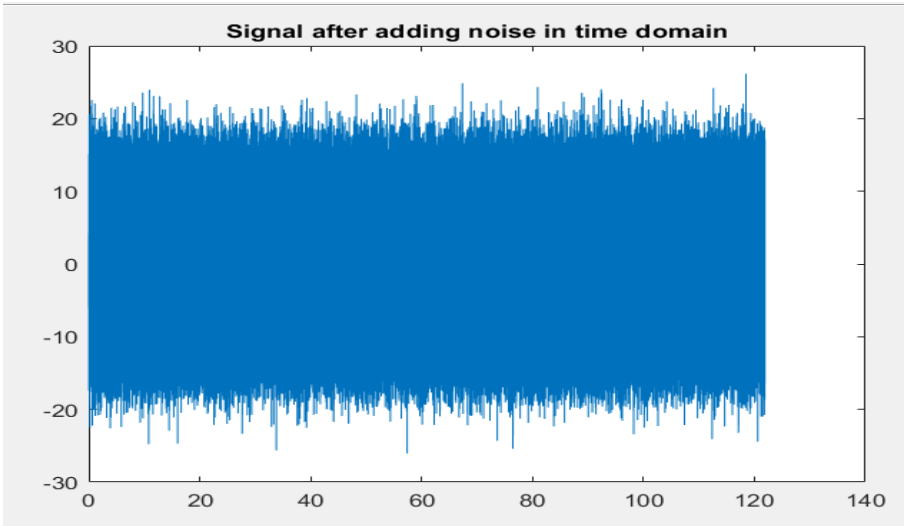


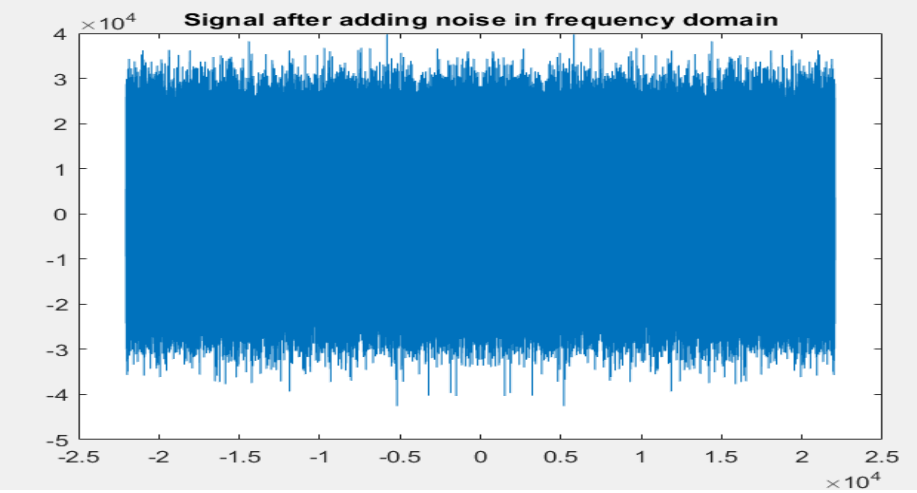
3. exp(-2pi*1000t):





4. The channel has the following impulse response:





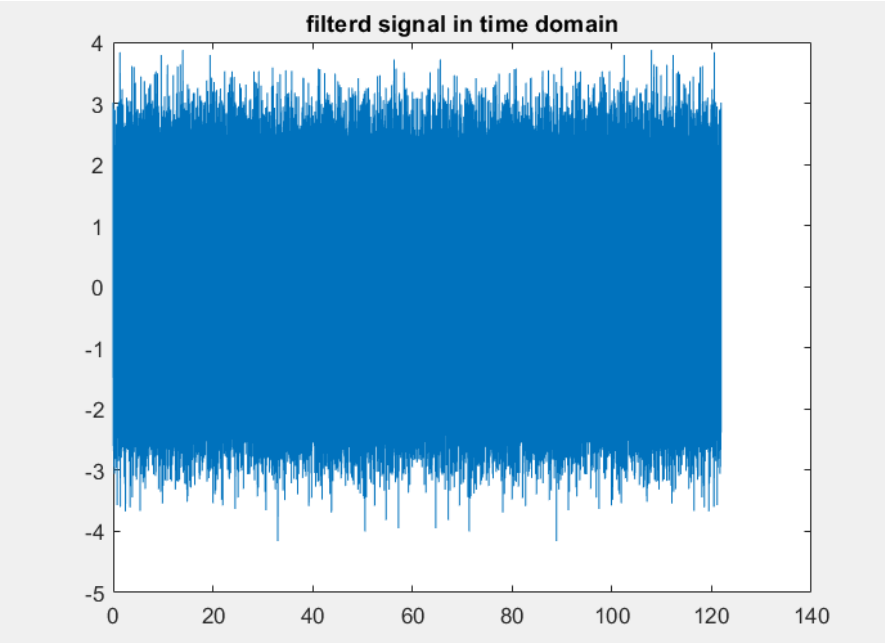
4) Receiver

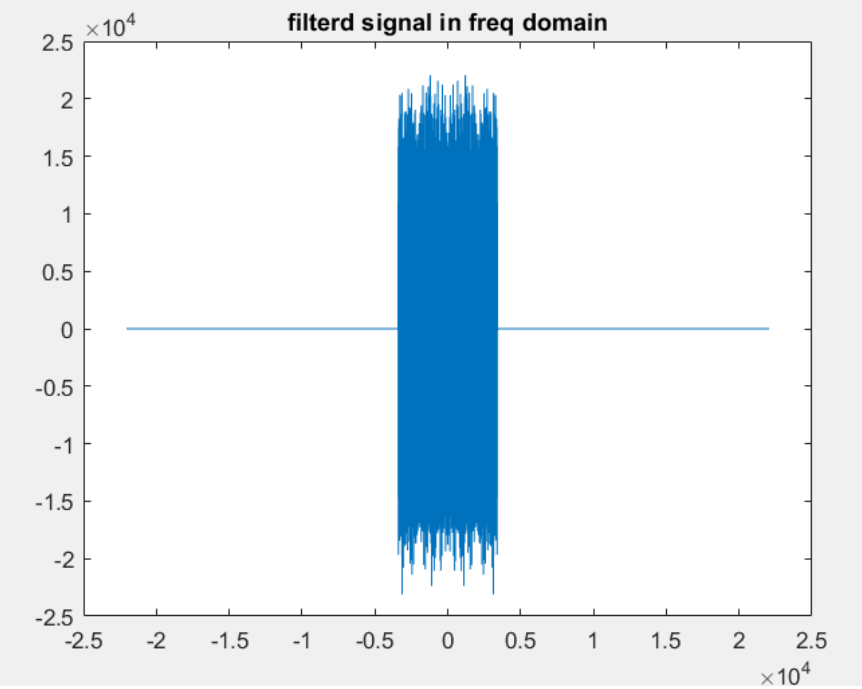
An ideal low pass filter which has a cut off of 3400 KHz will be constructed and then, pass the noisy sound over the ideal filter.

Code:

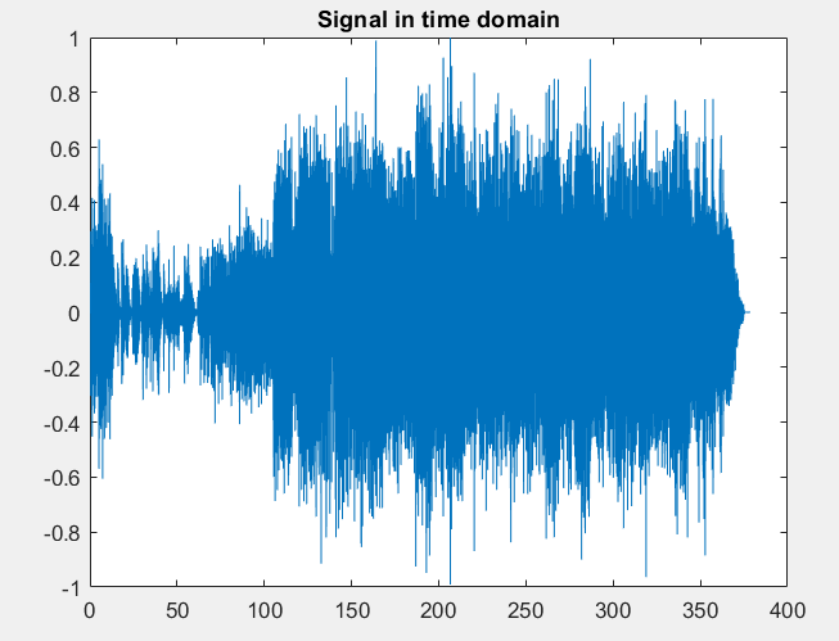
```
93 -    cutoff_frequency = 3400;
94 -    y_freq([1:round(((fs/2)-cutoff_frequency))*(length(y_noise)/fs) round(((length(y_noise)-((fs./2)-cutoff_frequency)*(length(y_noise)/fs)+1))):length(y_noise)])= 0;
95
96 -    filtered_signal_time = real(ifft(ifftshift(y_freq)));
97 -    figure ;
98 -    plot(t , filtered_signal_time);
99 -    title ('filterd signal in time domain');
100
101 -   f=linspace(-fs/2,fs/2,length(y_noise));
102 -   figure ;
103 -   plot (f,y_freq);
104 -   title ('filterd signal in freq domain');
105
106 -   sound(filtered_signal_time , fs);
```

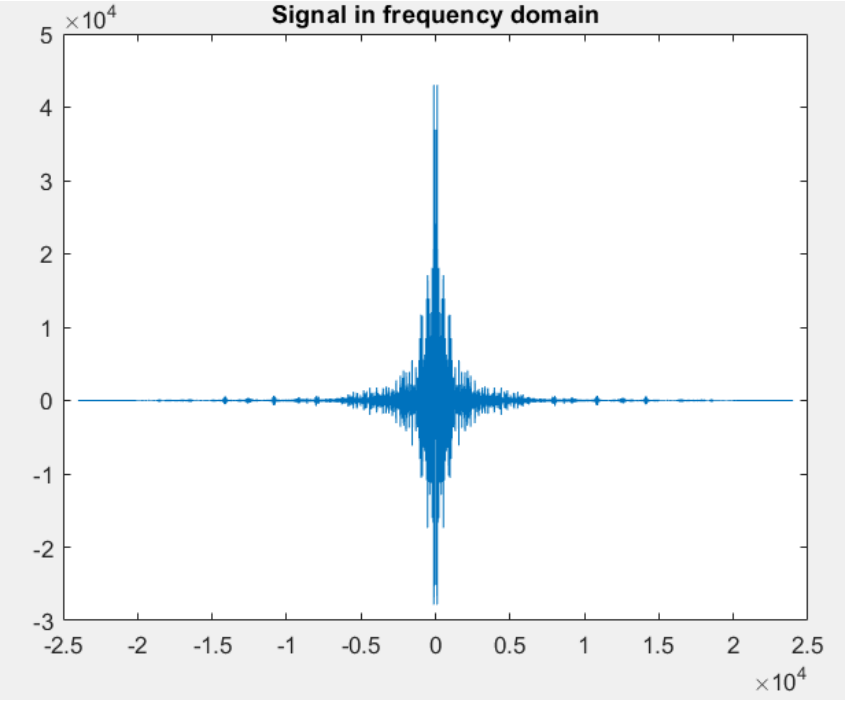
Output:


filterd signal in time domain
4
3
2
1
0
-1
-2
-3
-4
-5
0
20
40
60
80
100
120
140


filterd signal in freq domain
×10^4
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
×10^4

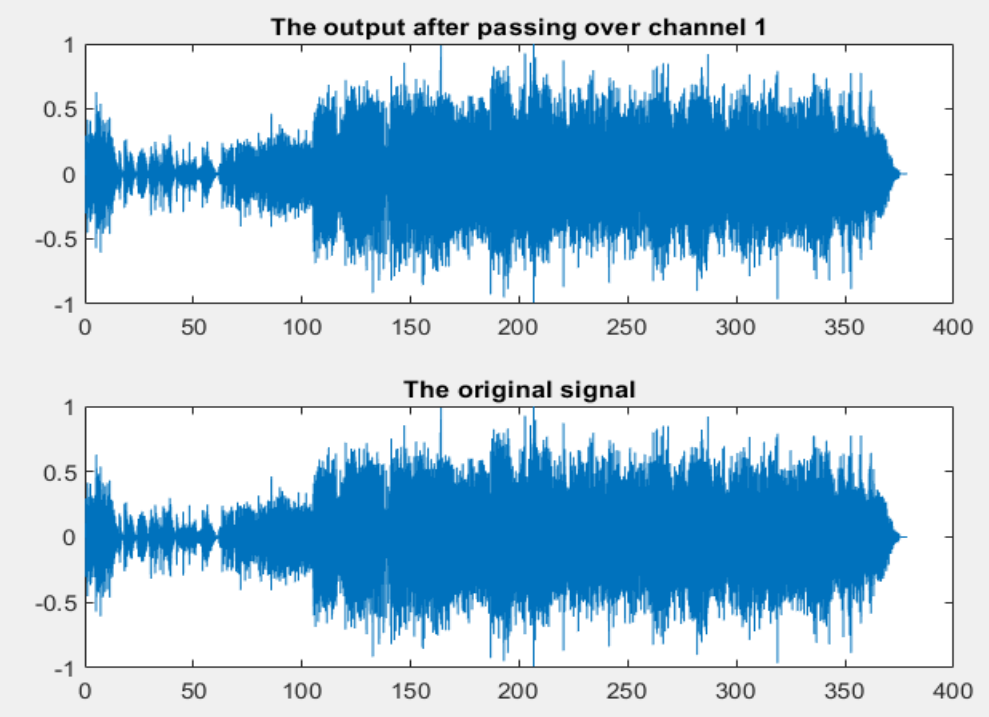
This project is tried for a Music file:



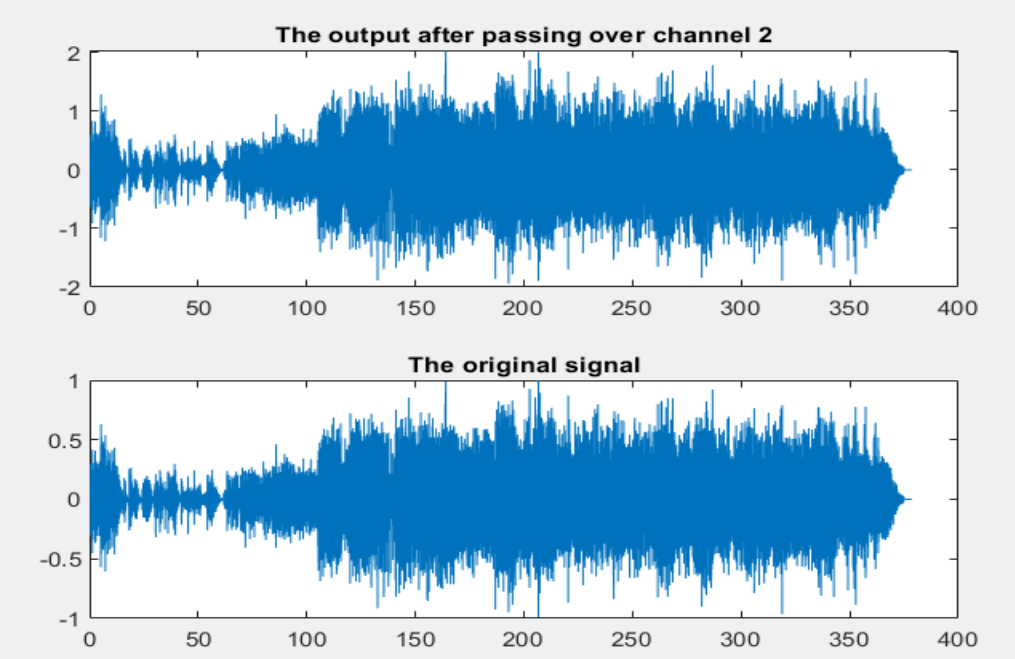


Channel:

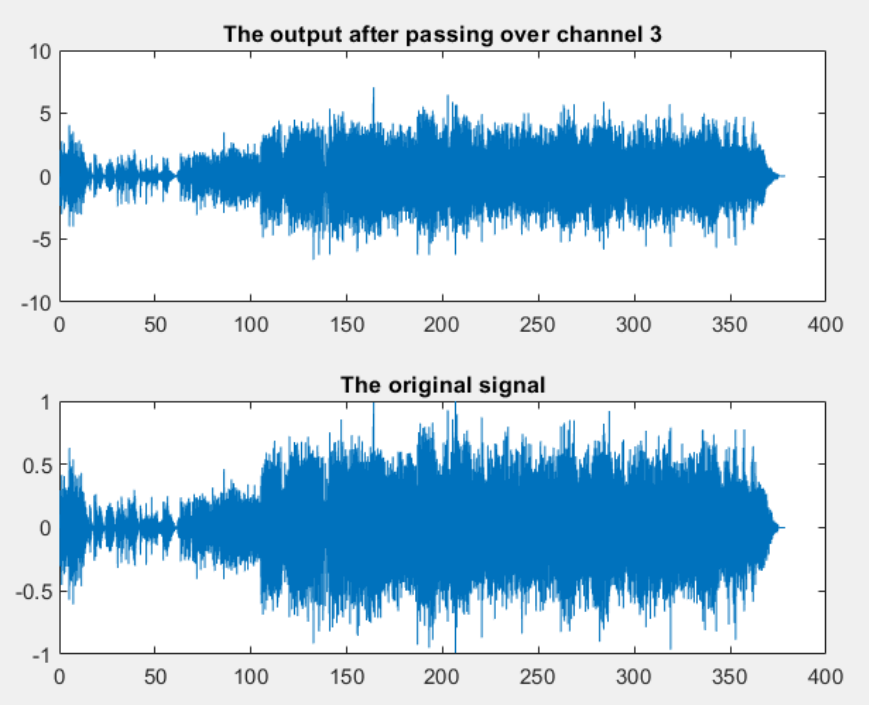
1. Delta function:



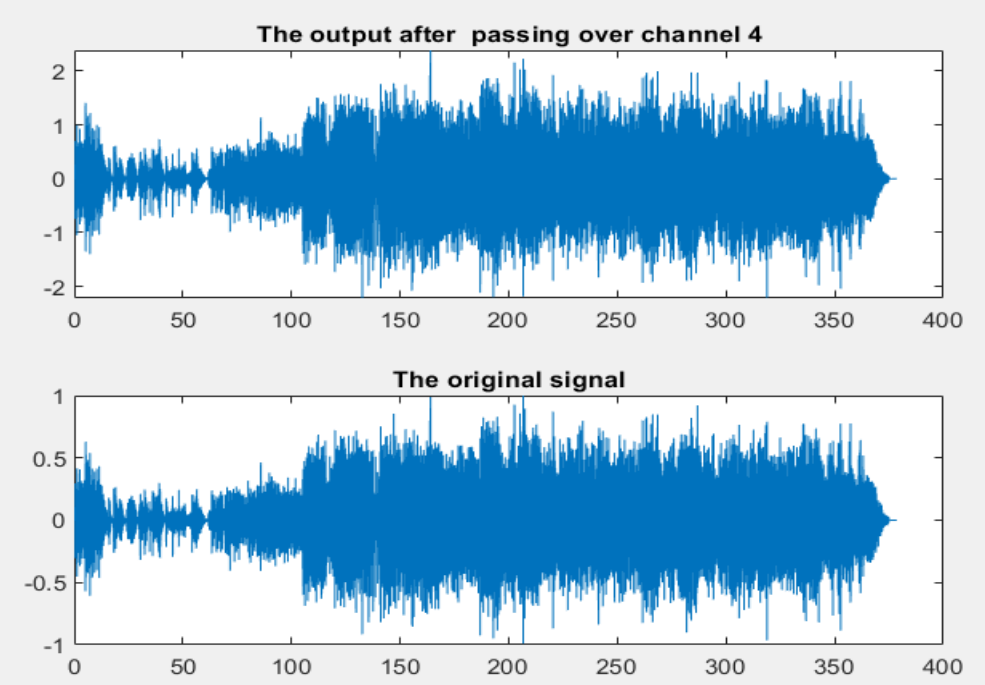
2. exp(-2pi*5000t):



3. exp(-2pi*1000t):

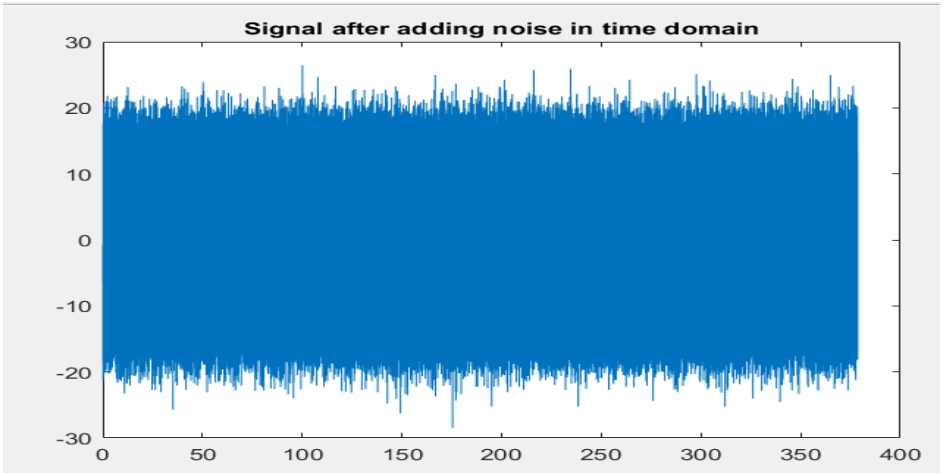


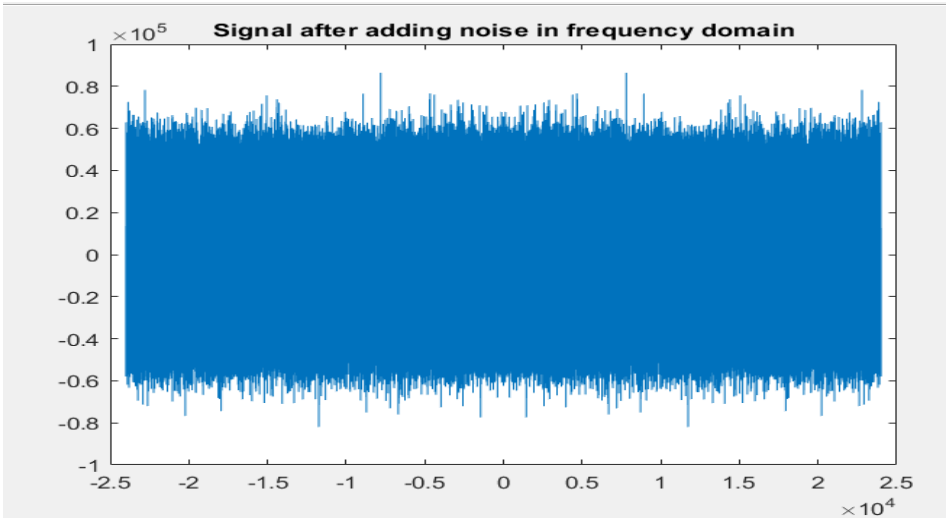
4. The channel has the following impulse response



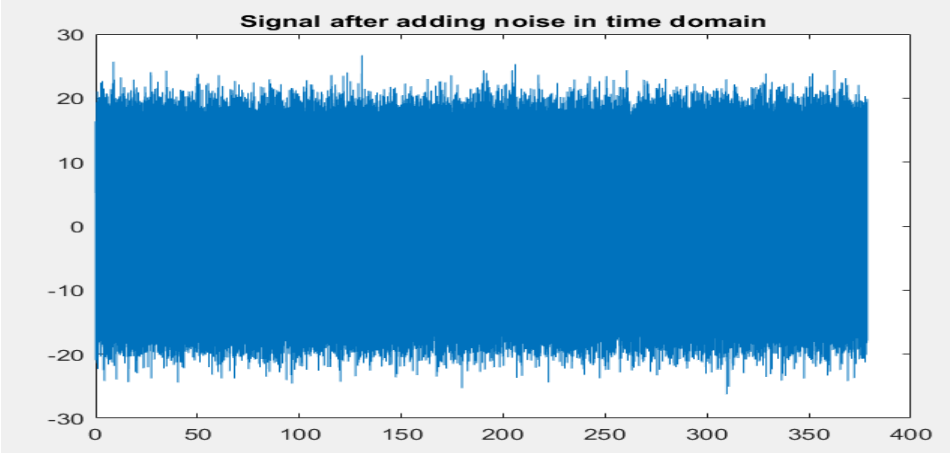
Noise:

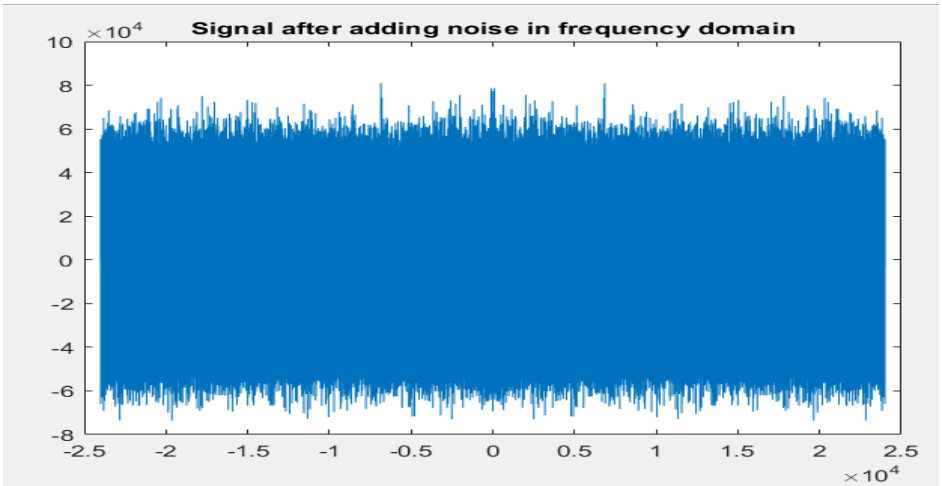
1. Delta function:



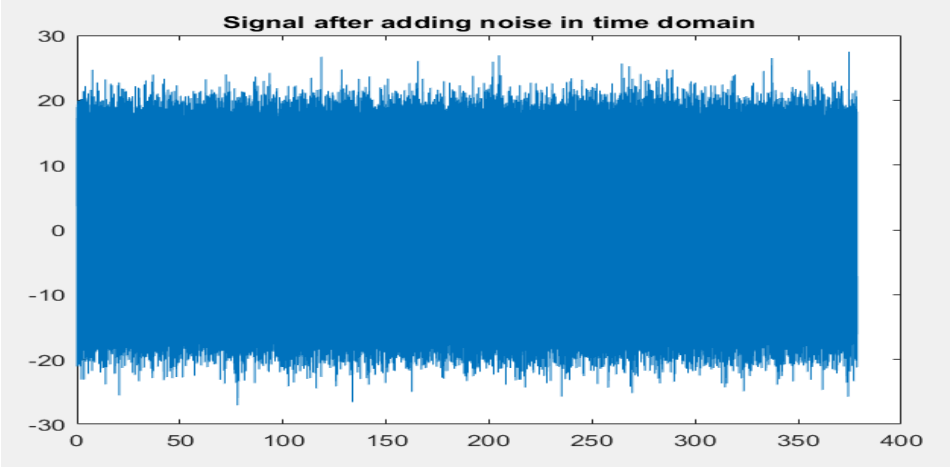


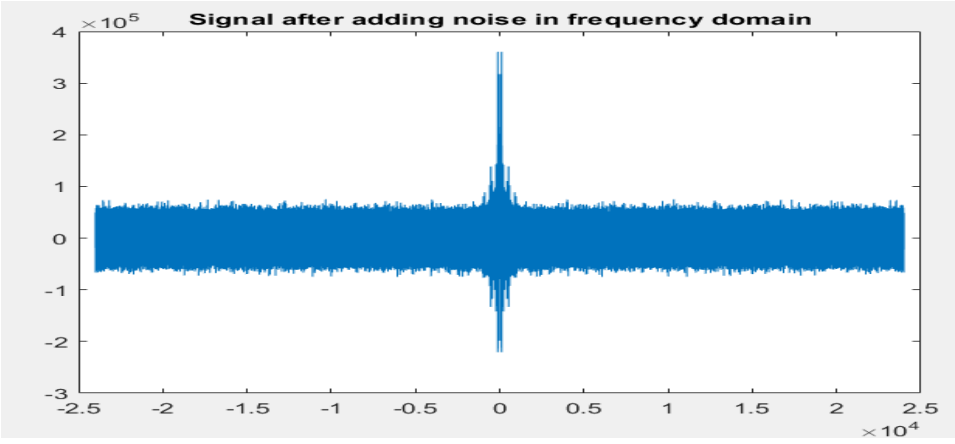
2. exp(-2pi*5000t):



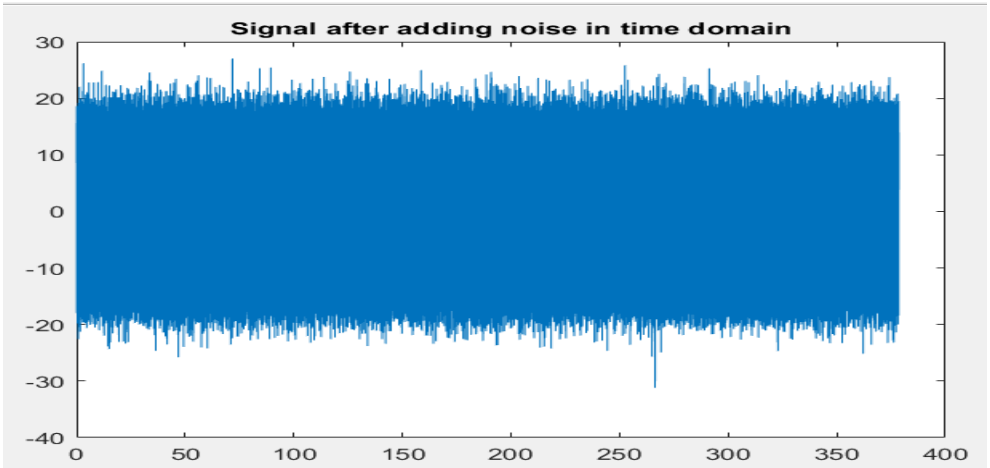


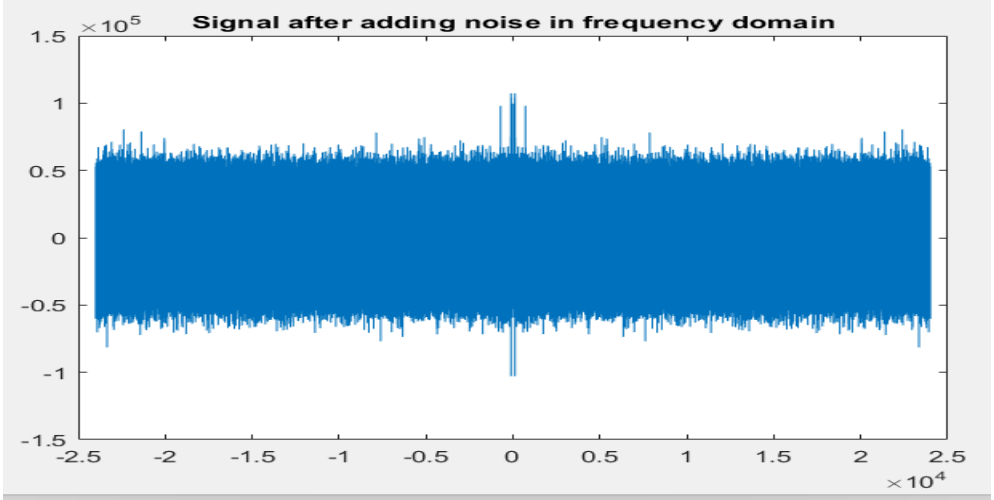
3. exp(-2pi*1000t):



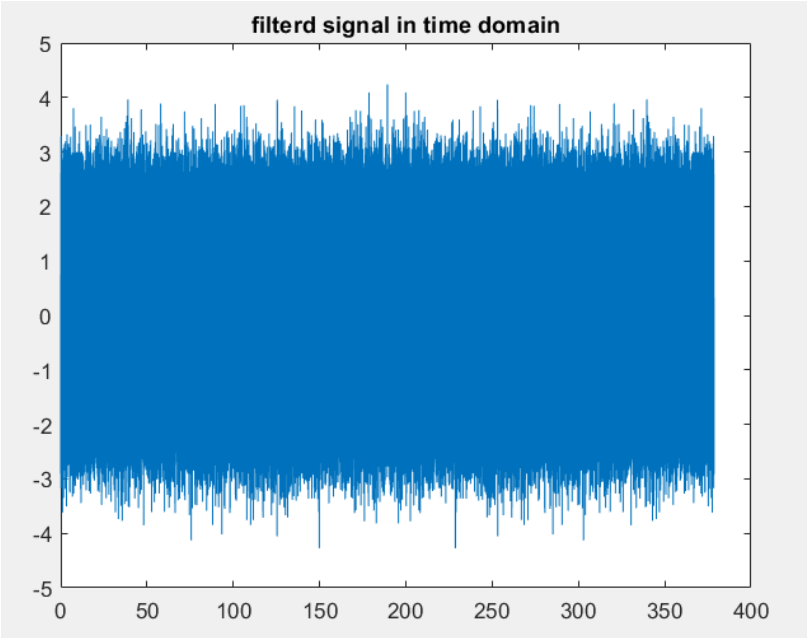


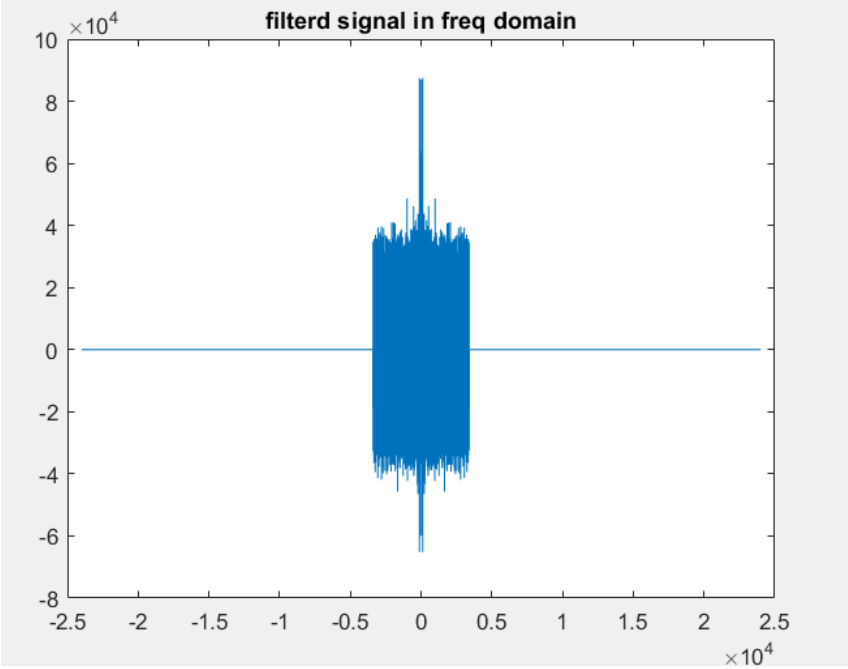
4. The channel has the following impulse response





Receiver:





The effect of the project when it is applied to both files (sound and music):

Music signal has a broader frequency range and more intricate frequency content. So, it will be affected by the low pass filter. Sound signal wasn't affected as the frequency is low.

Code

```
[y,fs] = audioread('Sound.mp3');
y = y(:,1) ;
sound(y,fs);

t = linspace(0, length(y)/fs, length(y));

figure;
plot(t, y);
title('Signal in time domain');

f=linspace(-fs/2,fs/2,length(y));
y_freq=real(fftshift(fft(y)));
figure;
plot(f,y_freq);
title('Signal in frequency domain');


h1=[1 zeros(1,length(y)-1)];
h2=exp(-2*pi*5000*t);
h3=exp(-2*pi*1000*t);
h4=[2 zeros(1,length(0:0.1:1)-2) 0.5];

y_1=conv(h1,y);
y_1 = y_1(t<=length(y));
figure;
subplot(2,1,1);
plot(t,y_1);
title('The output after passing over channel 1');
subplot(2,1,2);
plot(t,y);
title('The original signal');

y_2=conv(h2,y);
y_2 = y_2(t<=length(y));
figure;
subplot(2,1,1);
plot(t,y_2);
title('The output after passing over channel 2');
subplot(2,1,2);
plot(t,y);
title('The original signal');

y_3=conv(h3,y);
y_3 = y_3(t<=length(y));
figure;
subplot(2,1,1);
plot(t,y_3);
title('The output after passing over channel 3');
subplot(2,1,2);
plot(t,y);
title('The original signal');

y_4=conv(h4,y);
y_4= y_4(t<=length(y));
figure;
subplot(2,1,1);
plot(t,y_4);
title('The output after  passing over channel 4');
subplot(2,1,2);
plot(t,y);
title('The original signal');

sigma = input('Enter the sigma value of the noise: ');
Channel = input('Enter the channel number that the noise will be added to it:
');
switch Channel
    case 1
        Z = (sigma*randn(1,length(y_1)))';
        y_noise = y_1 + Z ;
    case 2
        Z = (sigma*randn(1,length(y_2)))';
        y_noise = y_2 + Z ;
    case 3
        Z = (sigma*randn(1,length(y_3)))';
        y_noise = y_3 + Z ;
    case 4
        Z = (sigma*randn(1,length(y_4)))';
        y_noise = y_4 + Z ;
end

sound(y_noise,fs);

figure;
plot(t, y_noise);
title('Signal after adding noise in time domain');

f=linspace(-fs/2,fs/2,length(y_noise));
y_freq=real(fftshift(fft(y_noise)));
figure;
plot(f,y_freq);
title('Signal after adding noise in frequency domain');


cutoff_frequency = 3400;
y_freq([1:round(((fs/2)-cutoff_frequency))*(length(y_noise)/fs)
round(((length(y_noise)-((fs./2)-cutoff_frequency)*
(length(y_noise)/fs)+1))):length(y_noise)])= 0;

filtered_signal_time = real(ifft(ifftshift(y_freq)));
figure ;
plot(t , filtered_signal_time);
title ('filterd signal in time domain');

f=linspace(-fs/2,fs/2,length(y_noise));
figure ;
plot (f,y_freq);
title ('filterd signal in freq domain');

sound(filtered_signal_time , fs);
```