PES ASSIGNMENT 6

Comb filter design and Audio filtering

Submitted by

Sai Joshitha Annareddy

21EE62R10

MTech

1. Record audio signal and AC noise

Matlab code

```
clc
clear all
Fs = 16000 ;
nBits = 16 ;
nChannels = 1 ;
ID = -1; % default audio input device
recObj = audiorecorder(Fs,nBits,nChannels,ID);
disp('Start recording.')
recordblocking(recObj,2);
disp('stop Recording.');
voice = getaudiodata(recObj);
audiowrite('Filename.wav', voice, 16000);
```

2. Read audio signal and noise signal
3. Add two signals
4. Observe the time domain and frequency domain plots of signal noise and signal with noise
5. Design comb filter using convex optimization method at noise frequencies.
   Design comb filter by inbuilt command iircomb
6. Get output by convolution of input with filter
7. Convert into fixed point using fi command
8. Write fixed-point filter coefficients & input signal

Matlab code

```
clc
clear all
close all
[s,Fs]=audioread("Sampleaudio.wav");

[v,Fs]=audioread("ACnoise.wav");

x=s+v;
%notch filter design
Fs=16000;
Ts=1/Fs;

M=256;%order
N=M/2+1;%no of filter coefficients

%notch width
alfa=16*2*pi*Ts;

 fo=72*Ts;%notch freq1
 wo=2*pi*fo;

 f1=320*Ts
 w1=2*pi*f1;

 f2 =480*Ts;%notchfreq2
 w2 = 2*pi*f2;

  f3 =800*Ts;%notchfreq2
 w3 = 2*pi*f3;


%FInd the P matrix
P=zeros(N);
q=zeros(N,1);

%part1
dw=pi/300;%freq domain sampling
w=[0:dw:(wo-alfa/2)]';
nM=[0:N-1]';
U=cos(nM*w');
for n1=1:N
    for n2=1:N
```

```
    P(n1,n2)=P(n1,n2)+trapz(U(n1,:).*U(n2,:))*dw;
    end
end
q=q-2*trapz(U,2)*dw;


%part2
dw=pi/500;
W=10000;%Weight of Notch part
epsi=0.0001;
w=[(wo-alfa/2):dw:(wo+alfa/2)]';
nM=[0:N-1]';
U=cos(nM*w');
for n1=1:N
    for n2=1:N
    P(n1,n2)=P(n1,n2)+W*trapz(U(n1,:).*U(n2,:))*dw;
    end
 end
q=q-2*W*epsi*trapz(U,2)*dw;

%part3
dw=pi/300;
w=[(wo+alfa/2):dw:(w1-alfa/2)]';
nM=[0:N-1]';
U = cos(nM*w');
for n1=1:N
    for n2=1:N
    P(n1,n2)=P(n1,n2)+trapz(U(n1,:).*U(n2,:))*dw;
    end
   end
q=q-2*trapz(U,2)*dw;

%part 4

dw=pi/500;
w=[(w1-alfa/2):dw:(w1+alfa/2)]';
nM=[0:N-1]';
U = cos(nM*w');
for n1=1:N
    for n2=1:N
    P(n1,n2)=P(n1,n2)+W*trapz(U(n1,:).*U(n2,:))*dw;
    end
   end
```

```
q=q-2*W*epsi*trapz(U,2)*dw;

%part 5
dw=pi/300;
w=[(w1+alfa/2):dw:w2-alfa/2]';
nM=[0:N-1]';
U = cos(nM*w');
for n1=1:N
    for n2=1:N
    P(n1,n2)=P(n1,n2)+trapz(U(n1,:).*U(n2,:))*dw;
    end
   end
q=q-2*trapz(U,2)*dw;

%part 6

dw=pi/500;
w=[(w2-alfa/2):dw:(w2+alfa/2)]';
nM=[0:N-1]';
U = cos(nM*w');
for n1=1:N
    for n2=1:N
    P(n1,n2)=P(n1,n2)+W*trapz(U(n1,:).*U(n2,:))*dw;
    end
   end
q=q-2*W*epsi*trapz(U,2)*dw;
%part 7
dw=pi/300;
w=[(w2+alfa/2):dw:(w3-alfa/2)]';
nM=[0:N-1]';
U = cos(nM*w');
for n1=1:N
    for n2=1:N
    P(n1,n2)=P(n1,n2)+trapz(U(n1,:).*U(n2,:))*dw;
    end
   end
q=q-2*trapz(U,2)*dw;
%part8
dw=pi/500;
w=[(w3-alfa/2):dw:(w3+alfa/2)]';
nM=[0:N-1]';
U = cos(nM*w');
for n1=1:N
```

```
    for n2=1:N
    P(n1,n2)=P(n1,n2)+W*trapz(U(n1,:).*U(n2,:))*dw;
    end
   end
q=q-2*W*epsi*trapz(U,2)*dw;
%part 9
dw=pi/300;
w=[(w3+alfa/2):dw:pi]';
nM=[0:N-1]';
U = cos(nM*w');
for n1=1:N
    for n2=1:N
    P(n1,n2)=P(n1,n2)+trapz(U(n1,:).*U(n2,:))*dw;
    end
   end
q=q-2*trapz(U,2)*dw;

%solve for minimization
a=-P\(q/2);
for k=1:M/2-1
    h(M/2-k)=a(k+1)/2;
    h(M/2+k)=a(k+1)/2;
end
h(M/2)=a(1);



%combfilterdesign
fs = 16000; fo = 100;  q = 35; bw = (fo/(fs/2))/q;
z = iircomb(fs/fo,bw,'notch'); % Note type flag 'notch'


y=conv(x,h)
y1=conv(x,z)

B = 16;
h_frac = floor(log2(2^B-1/max(abs(h))));
sig_frac =floor(log2(2^B-1/max(abs(x))));

h_fixed=fi(h, 1, B , h_frac);
sig_fixed = fi(x, 1, B, sig_frac);

y = conv(h,x);
```

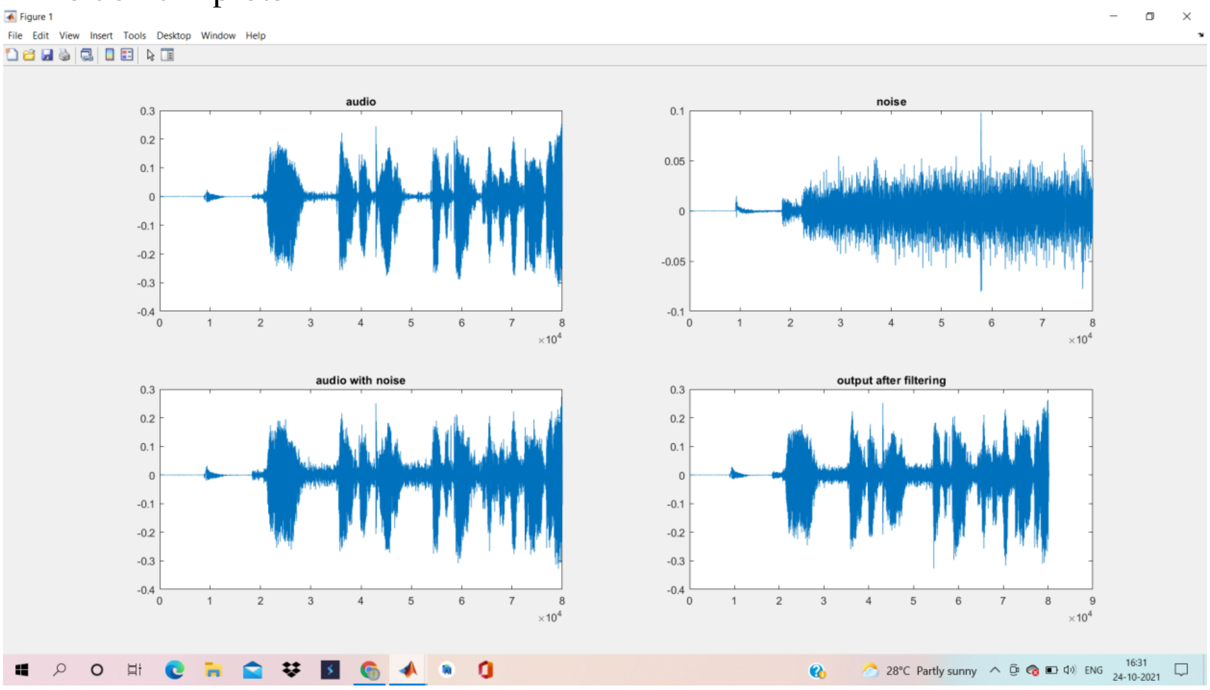
```
y_fixed = conv(h_fixed, sig_fixed);



%plots in time domain
figure(1);
subplot(221)
plot(s)
title("audio")
subplot(222)
plot(v)
title("noise")
subplot(223)
plot(x)
title("audio with noise")
subplot(224)
plot(y)
title("output after filtering")

%plots of power spectral density
figure(2)
subplot(231)
pwelch(s)
title("audio")
subplot(232)
pwelch(v)
title("noise")
subplot(233)
pwelch(x)
title("audio with noise")
subplot(234)
pwelch(h)
title("combfilter")
subplot(235)
pwelch(y)
title("output after filtering using fir designed combfilter")
subplot(236)
pwelch(y1)
title("output after filtering using iircombfilter")


%plots in frequency domain
figure(3);
```

```
F=linspace(0,Fs/2,1000);
subplot(231)
sf=freqz(s,1,F,Fs);
plot(F,abs(sf))
title("audio")
subplot(232)
vf=freqz(v,1,F,Fs);
plot(F,abs(vf))
title("noise")
subplot(233)
xf=freqz(x,1,F,Fs);
plot(F,abs(xf))
title("audio with noise")
subplot(234)
hf=freqz(h,1,F,Fs);
plot(F,abs(hf))
title("combfilter")
subplot(235)
yf=freqz(y,1,F,Fs);
plot(F,abs(yf))
title("output after filtering using firdesigned combfilter")
subplot(236)
y1f=freqz(y1,1,F,Fs);
plot(F,abs(y1f))
title("output after filtering using iircomb filter")

%after conversion to fixed point

figure(4)
subplot(311)
plot(sig_fixed)
title("Input Signal - 16 bit of fixed point")
subplot(312)
plot(y_fixed)
title("output - 16 bit convolution of fixed point")
subplot(313)
plot(h_fixed)
title("firfilter of 16 bit fixed point");

%% Write fixed-point filter coefficients & input signal in hex format
file1=fopen('Filter Co-efficients from MATLAB.txt', 'w');
for i=1:1:length(h_fixed)
    h=h_fixed(i);
```

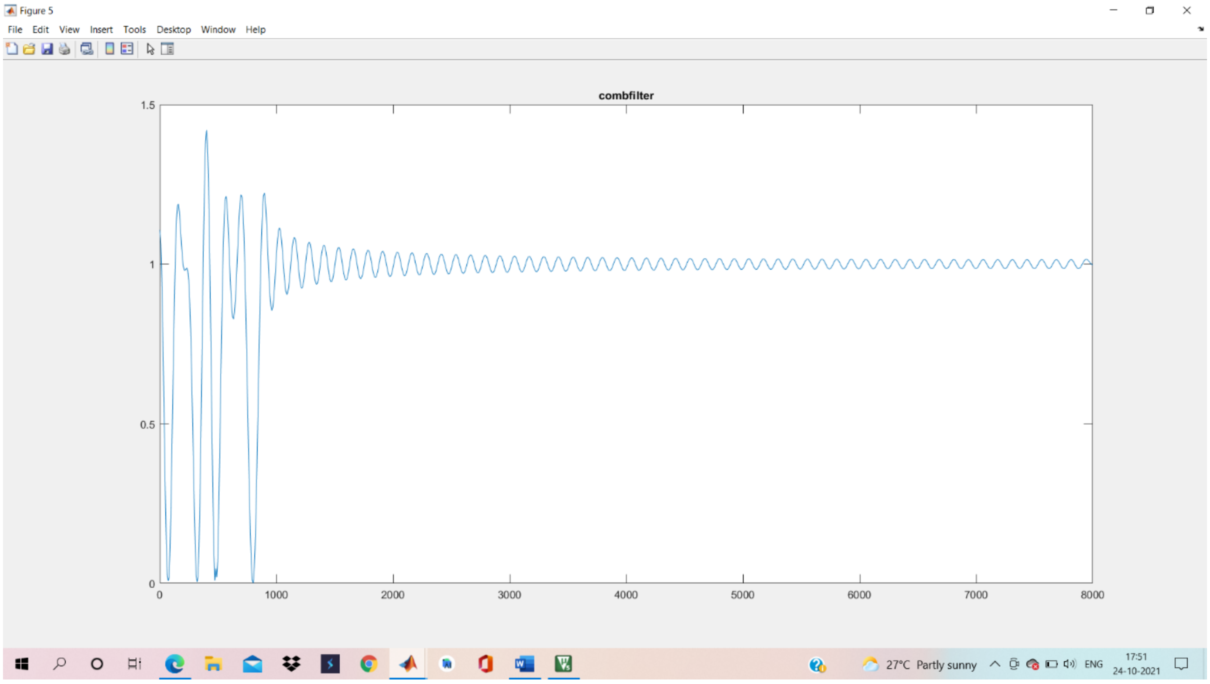
```
    if i<length(h_fixed)
        fprintf(file1, '0x%s, ', hex(h));
    else
        fprintf(file1, '0x%s', hex(h));
    end
end
fclose(file1);
file2=fopen('Input Signal Data from MATLAB.txt', 'w');
for i=1:length(sig_fixed)
    si=sig_fixed(i);
    if i<length(sig_fixed)
        fprintf(file2, '0x%s, ', hex(si));
    else
        fprintf(file2, '0x%s', hex(si));
    end
end
fclose(file2);
```

Matlab plots:

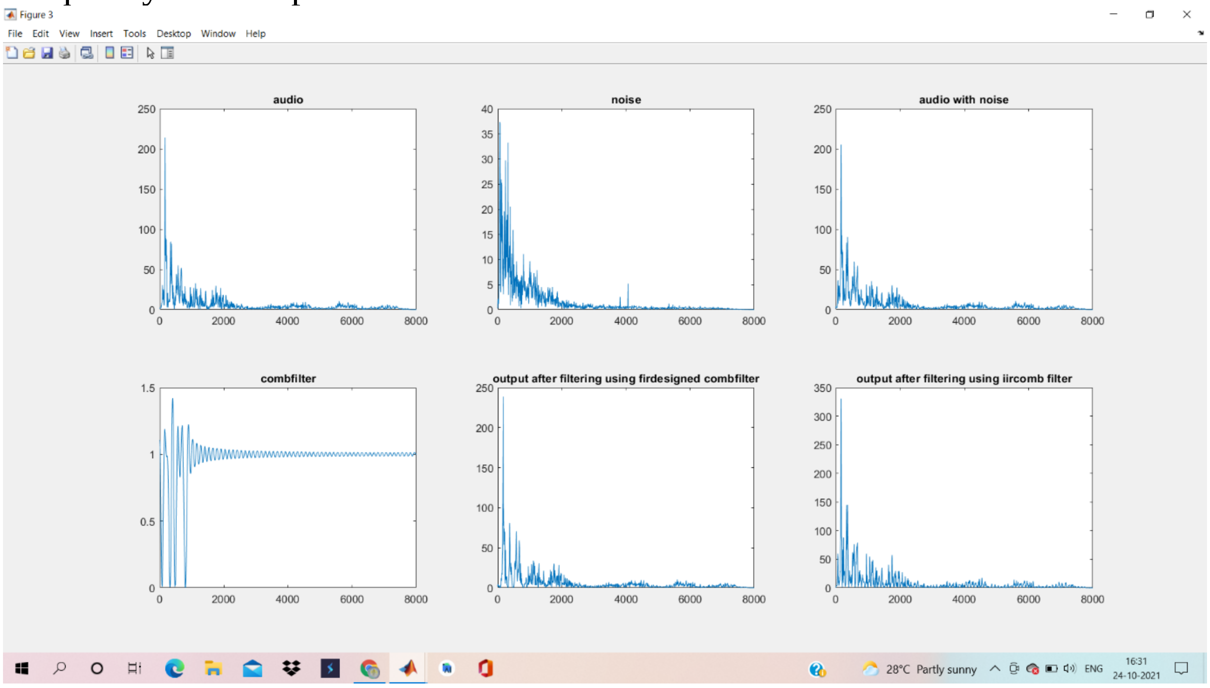
Time domain plots



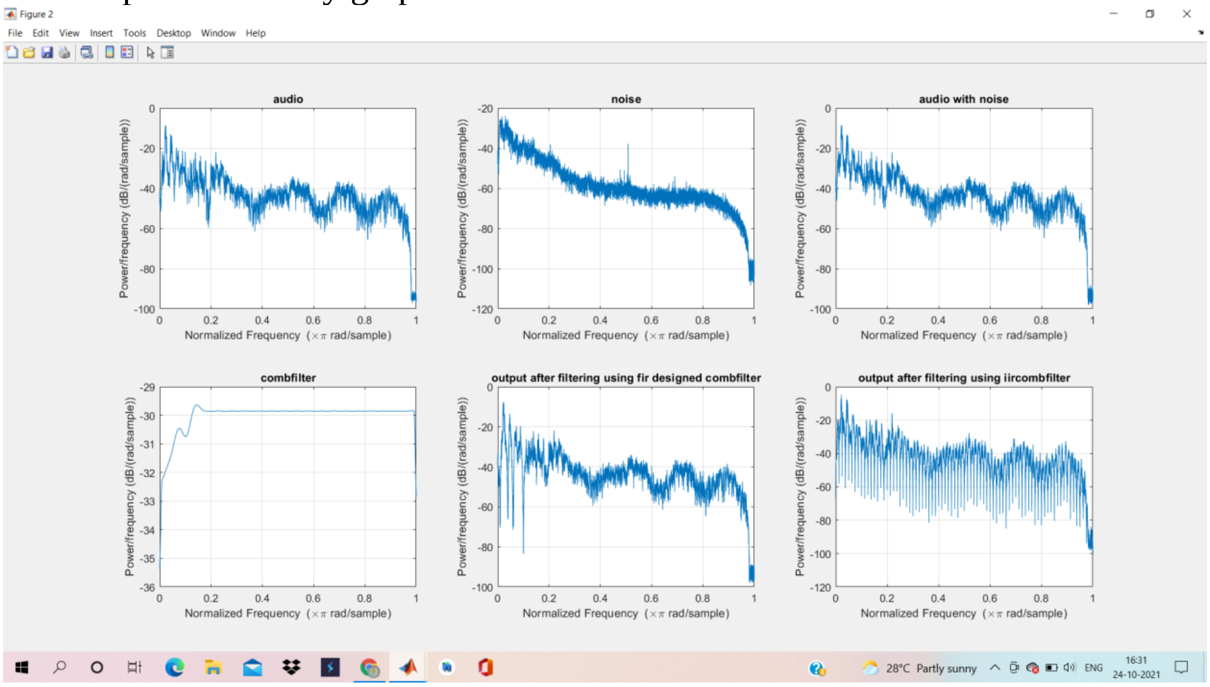
FIR comb filter



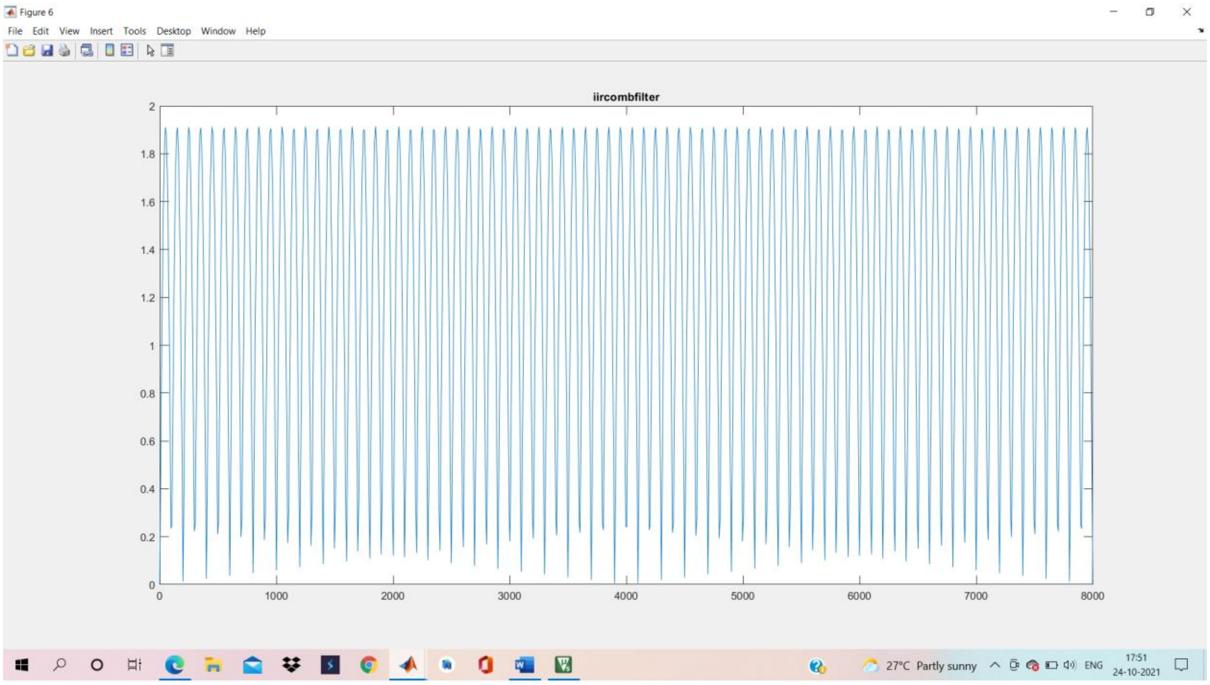
Frequency domain plots



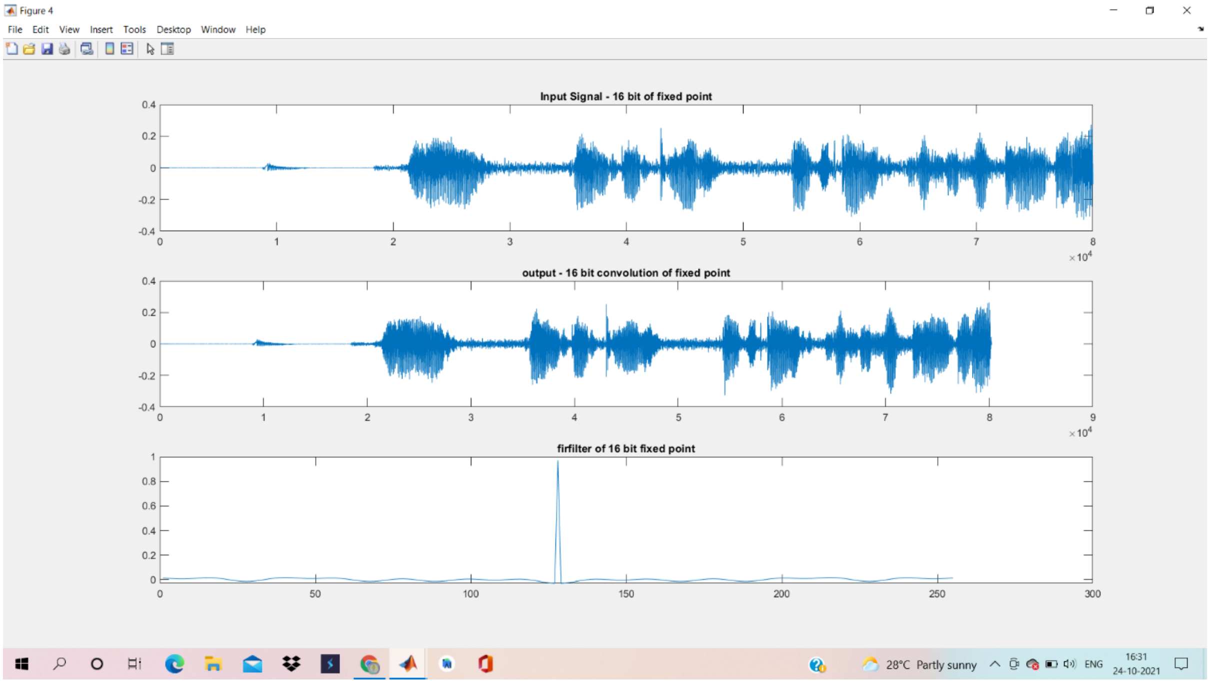
Power spectral density graphs



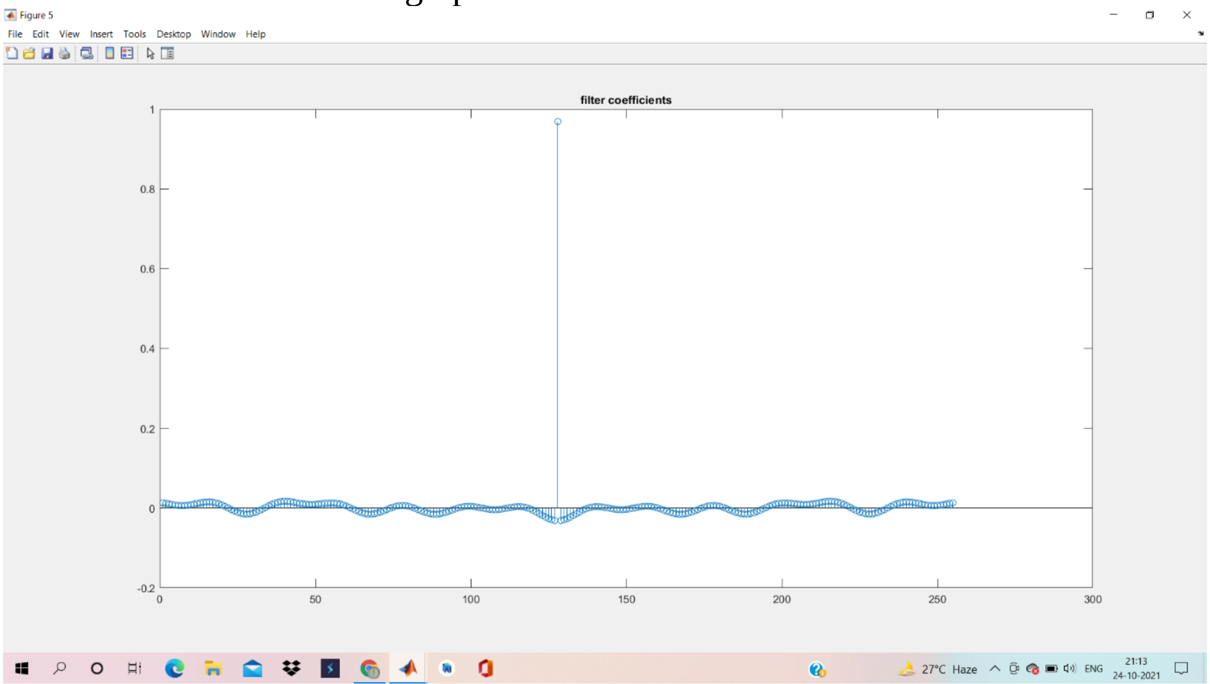
Iir Comb filter



Fixed point representation graphs:



Fircomb filter coefficients graph



Using stm32 :

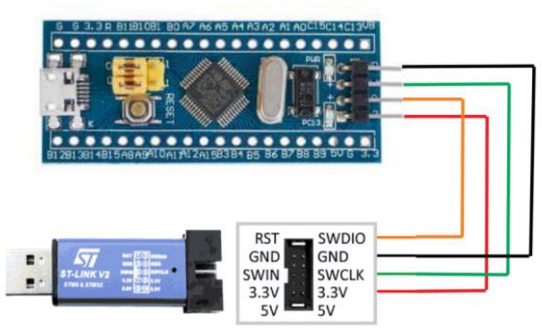
Convolution Code:

```
        AREA main, CODE, READONLY ;area is a directive
        EXPORT __main
        ENTRY
sig dcw #input signal coefficients
```

```
h dcw #output signal coefficients
f equ 8 ;length of fraction part of fixed point

l equ 510 ; Input length*2 for 2 byte alignment
hl equ 510 ; Filter length*2 for 2 byte alignment
ol equ 1018 ; Output length*2 for 2 byte alignment

;convlution code
;for(i=0;i<ol;i++){
;y[i]=0;
;for(j=0;i<hl;i++)
;(if(0<i-j<l){
;y[i]=y[i]+x[i-j]*h[j];
;};
;};
;};

__main
        ldr r0,=sig ;Address of input signal
        ldr r1,=h ;Address of filter coefficients
        mov r2,#0x0000 ;output signal memory lower byte
        movt r2,#0x2000 ;output signal memory upper byte 0x2000000000

        mov r3,#0 ;i=0
__outer_loop ;for(i=0;i<ol;i++)
        cmp r3,#ol
        bge __exit_outer_loop ;exit if i>=ol
        mov r4,#0 ;j=0
        mov r9,#0 ;accumulator=0
__inner_loop ;for(j=0;i<hl;i++)
        cmp r4,#hl
        bge __exit_inner_loop ;exit if j>=hl
        subs r5,r3,r4 ;r5=i-j
        bmi __skip ;skip if i-j<0
        cmp r5, #l
        bge __skip ;skip if i-j>=length(input signal)
        add r6, r5, r0 ;r6=&x[i-j], address pointer
        ldrsh r11,[r6] ;r11=x[i-j]
        add r7,r4,r1 ;r6=&h[j], address pointer
        ldrsh r12, [r7] ;r12=h[j]
        bl __mul ;r10= x[i-j]*h[j]
        add r9, r9, r10 ;accumulator=accumulator+x[i-j]*h[j]
```

```
__skip
	add r4, r4, #2 ;next filter coeff address
	b __inner_loop ;repeat inner loop

__exit_inner_loop
	add r8, r3, r2 ;r8=&y[i], address pointer
	strh r9,[r8] ;store y[i] in &y[i]
	;ldrsh r10,[r8] ;to check y[i]
	add r3, r3, #2 ;next input signal address
	b __outer_loop ;repeat outer loop

__exit_outer_loop
	b __exit_outer_loop ;convolution end

__mul  ;multiply two 16bit fixed point numbers in r11 & r12, product in r10.
result format A(8,8)
	mul r10, r11,r12
	asr r10,#f ;multiply with 2^-f, f=8, as fixed point multiplication
	cmp r10,#0
	bmi __neg
	movt r10,#0x0000 ;Extend upper 16bits if no is positive
	bx lr
__neg
	movt r10,#0xffff ;Extend upper 16bits if no is negative
	bx lr
	END
```

Connection diagram



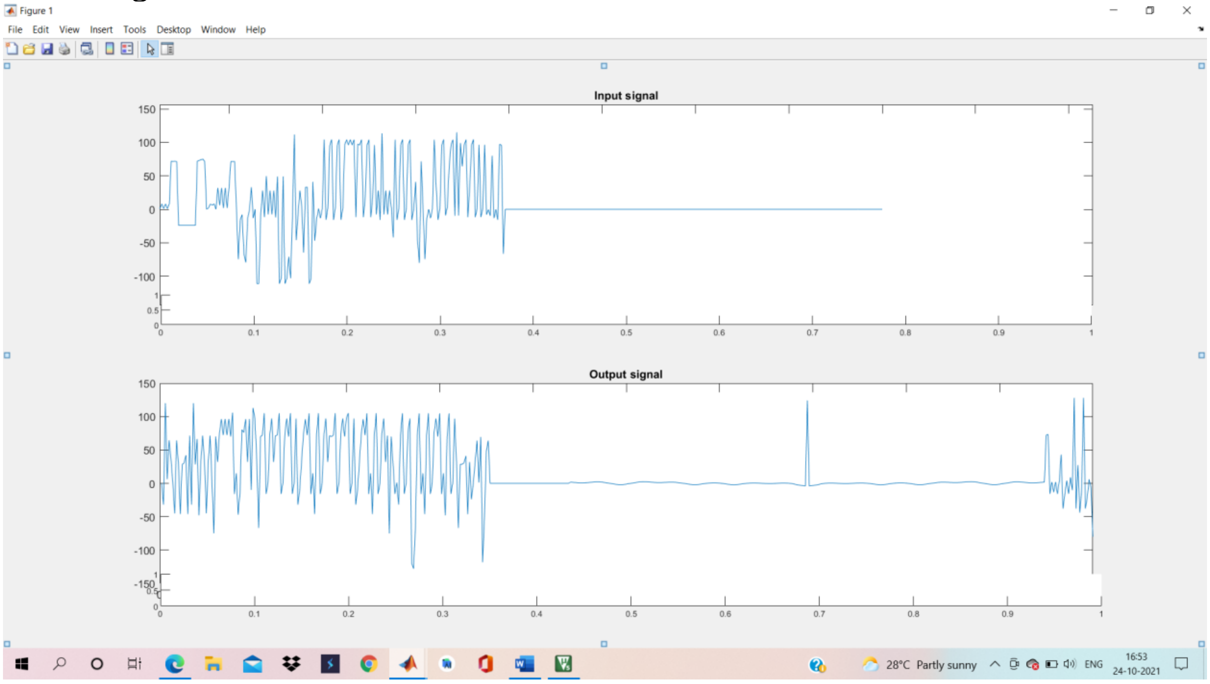
Matlab code to plot input and output audio signal graphs from stm32

```
iplen = 255;500
hlen = 255;
oplen = 255;500
```

```
fileID = fopen('inputsig.bin', 'r');
mat1 = fread(fileID);
fclose(fileID);

signalin = zeros(1, iplen);

for i=1:iplen
   if mat1(2*i) > 127
      signalin(1, i) = mat1(2*i)-256;
      signalin(1, i) = signalin(1, i) + mat1(2*i-1)/256;
   else
      signalin(1, i) = mat1(2*i);
      signalin(1, i) = signalin(1, i) + mat1(2*i-1)/256;
   end
end

fileID = fopen('outputsig.bin', 'r');
mat3 = fread(fileID);
fclose(fileID);


signal = zeros(1, oplen);

for i=1:oplen
   if mat3(2*i) > 127
      signal(1, i) = mat3(2*i)-256;
      signal(1, i) = signal(1, i) + mat3(2*i-1)/256;
   else
      signal(1, i) = mat3(2*i);
      signal(1, i) = signal(1, i) + mat3(2*i-1)/256;
   end
end

figure(1)
hold on;
plot(signalin);
title('Input signal');
hold off;
figure(2)
hold on;
plot(signal);
title('Output signal');
hold off;
```

Graphs:

Audio signal from stm32



Checked filter with sinusoidal signal