پروژه ی درس اصول مخابرا ت

رادیو نرمافزاری

افراد گروه:

محدثه غفوری ----------دکتر یزدیان

مهدیه احمدی----------دکتر مهدوی

مهیار عنصری----------دکتر یزدیان

دانشگاه صنعتی اصفهان

1398–1

مقدمه:

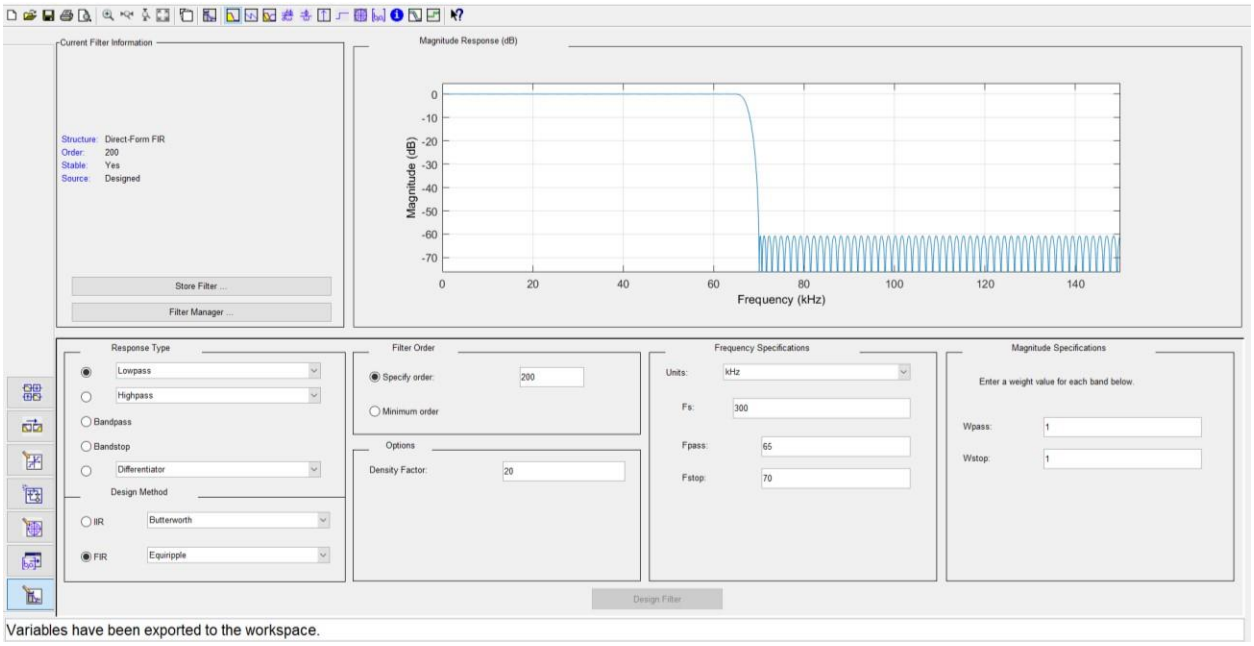
رادیوی نرم افزاری (Software-defined radio) یک سیستم رادیو مخابراتی است که مؤلفه هایی که معمولاً بصورت سخت افزاری هستند)برای نمونه میکسر، فیلتر ،مدولاسیو ن/دمودولاتور، آشکارساز و تقوی تکننده الکترونیکی) را به صورت نرم افزار
پیاده سازی میکند .از انواع آن میتوان RTL_SDR , HACKRF,LIMESDR , BLADERF نام برد.

دراین پروژه سعی بر آن شده که با استفاده از نرم افزار متلب یک رادیو نرمافزای را برنامه ریزی کرده و ساخت.

ابتدا باید فرکانس مرکزی نرخ نمونه برداری و تعداد نمونه ها در هر سمپل را مشخص کنیم.فرکانس مرکزی را بر روی 88KHz تنظیم و فرکانس نمونه برداری را بر روی 300KHz وتعداد نمونه در هر سمپل را 2^18 تعیین میکنیم.

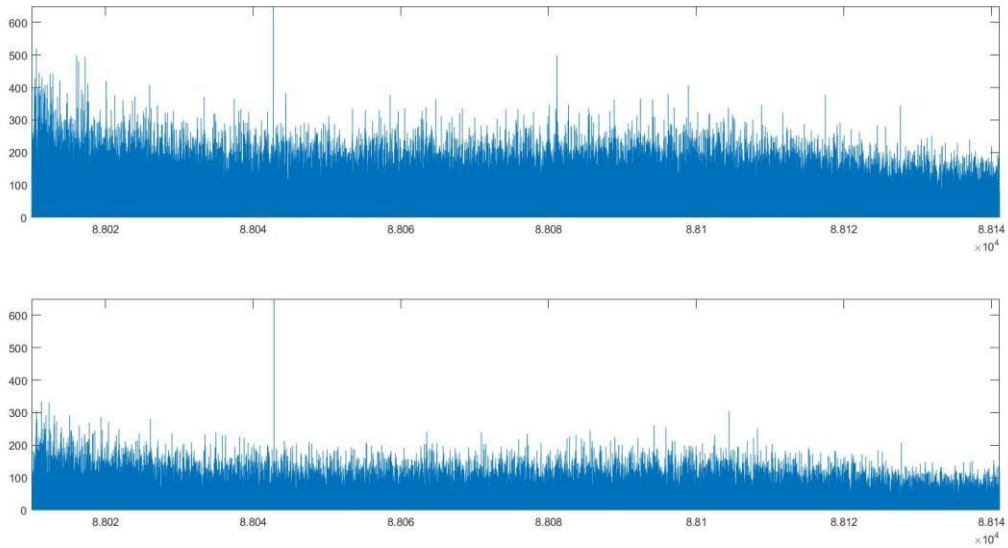
سیگنال ر ا ز sdr دریافت و بر روی متغییر sdr1 میریزیم. با استفاده از تا بع step(sdr1) سیگنال را دریافت میکنیم .سپس مقدار dc سیگنال را حذف و تبدیل فوریه نرمالیزه شدهای از سیگنال میگیریم .

فیلتری با نام Num به پهنای باند K300 که تا فرکانس 70K را عبور میدهد با اسفاده از filterDesigner طراحی و اعمال میکنیم و سیگنال اصلی و سیگنال فیلتر شده را مشاهده میکنیم.



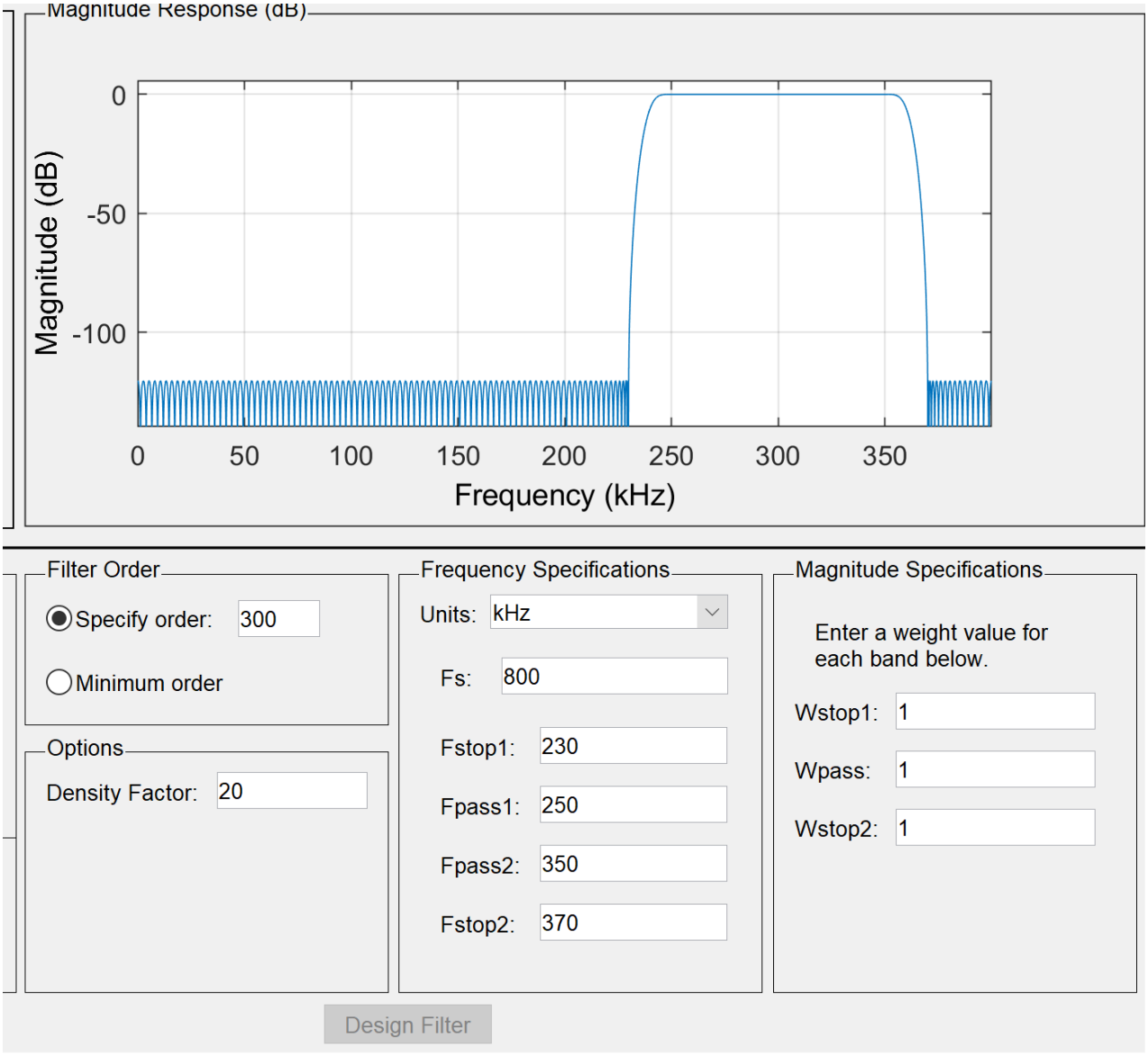

```
close all clc
centerfreq=88e6;
fs=300e3;
FrontEndSampleRate=2^18;
Fs=FrontEndSampleRate;  sdr1 =
comm.SDRRTLReceiver(...
    'CenterFrequency', centerfreq, ...
    'SampleRate',      fs, ...
    'SamplesPerFrame',    2^18,    ...
'OutputDataType',          'double');
sprintf('Initialization comlpleted');
figure
 N=1;
 L=FrontEndSampleRate*N;
 NFFT = 2^nextpow2(L); % Next power of 2 from length of y
f = Fs/2*linspace(0,1,NFFT/)1+2; if
~isempty(sdrinfo(sdr1.RadioAddress))     for count=1:5
fc=88e6+10e3;          sdr1.CenterFrequency=fc;
data=step(sdr1);          data = data - mean(data);
        Y=fft(data);
Y2=filter(Num,1,Y);
subplot(2,1,1);
        plot((f+fc)/1000,2*(Y(1:NFFT/2+1)));
ylim([0 650])
```

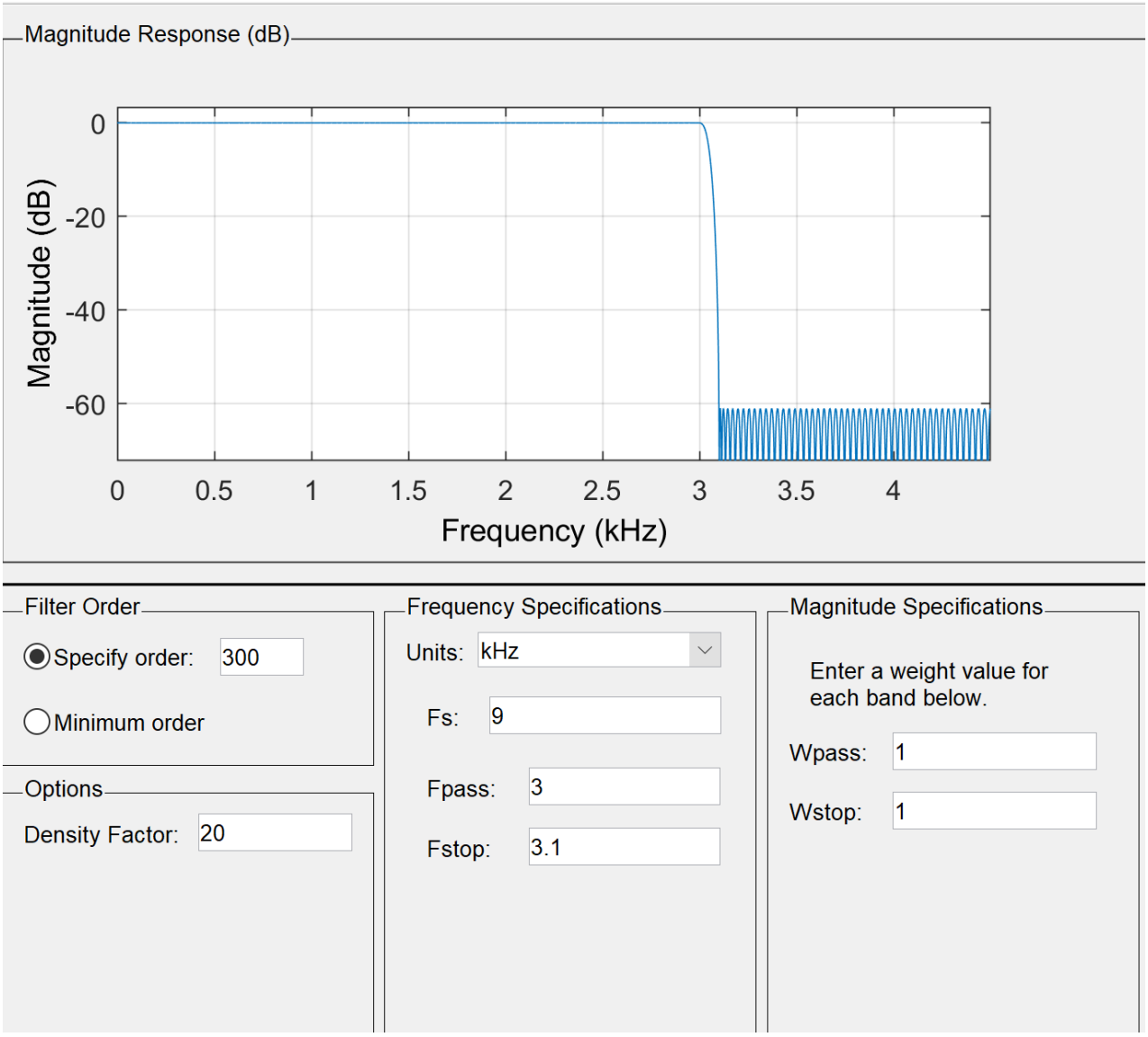
```
        xlim([min((f+fc)/1000) max((f+fc)/1000)])
subplot(2,1,2);
        plot((f+fc)/1000,2*(Y2(1:NFFT/2+1)));
ylim([0 650])          xlim([min((f+fc)/1000)
max((f+fc)/1000)])          drawnow     end else
     warning(message('SDR:sysobjdemos:MainLoop'))
end release(sdr1);
```



همانطور که مشاهده میشود با عمال فیلتر پایین گذ ر نویز کاهش یافته است.

در طول دمدولاسیون از دو نوع فیلتر میان گذر حول سیصد کیلو و فیلتر پایین گذر شبیه به دو فیلتر زیر استفاده میشود


Magnitude Response (dB)
Magnitude (dB)
0
-50
-100
0
50
100
150
200
250
300
350
Frequency (kHz)
Filter Order
Specify order:
300
Minimum order
Options
Density Factor:
20
Frequency Specifications
Units:
kHz
Fs:
800
Fstop1:
230
Fpass1:
250
Fpass2:
350
Fstop2:
370
Magnitude Specifications
Enter a weight value for each band below.
Wstop1:
1
Wpass:
1
Wstop2:
1
Design Filter



حال باید سیگنال فیلتر شده را آشکار کنیم.

با استفاده از روش اول:

یک واحد زما ن ی تاخیر می دهیم و در مزدوجش ضرب می کنیم .

```
m = Y2(1:end-1).*conj(Y2(2:end));
 v=angle(m);
```

حال باید نرخ نمونه برداری را کاهش ویک فیلتر پایین گذر مناسب بر روی سیگنال جدیدقرار دهیم.

پهنای باند سیگنال را کمتر از ده کیلو قرار می دهیم

```
n=decimate(v,20);
n=filter(Num2,1,n);
sound(abs(n),(fs/20));
```

استفاده از روش دوم

اشکار ساز پوش :با توجه به اینکه(s(t)=Ac*cos(2pi*fc*t+2*pi*k∫m(t) باگرفتن مشتق از این عبارت پوش عبارت حاوی سیگنال پیام میشود و با استفاده از دستور اشکار ساز پوش میتوان پوش حاوی پیام را با اعمال فیلتری با پهنای باند کمتر از ده کیلوهرتز دریافت کرد و با دادن نرخ نمونه برداری کاهش یافته به دستور sound رادیو را شنید

توجه شود که با دستور هیلبرت مولفه هم فاز و تربیعی باهم تولید میشوند و بصورت XI+jXQ ذخیره میشوند و با اعمال دستور اندازه روی ان پوش سیگنال ساخته میشود

```
y=diff(z);
v=abs(hilbert((real(y)));
n=decimate(v,25);
n=filter(Num2,1,n);
sound(abs(n),(fs/25));
```

استفاده از روش سوم

شمارش تعداد نقاط عبور از صفر سیگنال پیام : با توجه به این روش میدانیم هرچه فرکانس سیگنال پیام بیشتر باشد تعداد نقاط عبور از صفر نیز بیشتر است بنابراین با توجه به بلوک دیاگرام این روش ابتدا سیگنال از یک هارد لیمیتر عبور میکند که در متلب معادل با دستور sign میباشد سپس بوجود اوردن لبه با مشتق گرفتن از هارد لیمیتر انجام میشود و در نهایت با گرفتن اندازه ان و اعمال یک فیلتر پایین گذر که معادل با انتگرال گیر است تعداد نقاط عبور از صفر شمرده شده(این کار بوسیله یک انتگرال گیر که معادل با یک فیلتر پایین گذر است انجام میگیرد) و با کم کردن DC پیام استخراج میشود

```
v=abs(diff(sign(z)))-mean(abs(diff(sign(z))));
```

حال با کاهش نرخ نمونه برداری و اعمال فیلتر پایین گذر مناسب که پهنای باندی کمتر از ده کیلوهرتز برای جلوگیری از حضور نویز بصورت خش خش دارد میتوان صدای سیگنال را با دستور sound پخش کرد که بعلت کاهش نرخ نمونه برداری باید نرخ کاهش یافته را به این دستور بدهیم

```
n=filter(Num3,1,n);
n=decimate(v,30);
n=filter(Num2,1,n);
sound(abs(n),(fs/30));
```

فاز دوم :

با توجه به فاصله تقریبی یک و نیم مگ بین ایستگاه ها طول های گام پیمایش فرکانسی را یک و نیم مگ در بازه ی فرکانسی 88 مگ تا 108 مگ قرار میدهیم

در ابتدا با حرکت روی کل این بازه ها با گرفتن داده از SDR برنامه از ان تبدیل فوریه میگیرد و با استفاده از ان چگالی طیفی انرژی داده در هر طول گام محاسبه میشود و برای نرمالیزه شدن این چگالی طیفی به طول بازه تقسیم میشود و با متقارن بودن بودن چگالی طیفی سیگنال پیام ما که حقیقی است کافی است تنها نصفی از بازه چگالی طیفی انرژی را برای تولید چگالی طیفی توان استفاده و ذخیره میکنیم سپس حال با قرار دادن fc در ابتدای هر بازه و گام های یک و نیم مگ و به ازای هر پیمایش تمام چگالی طیفی هایی که در بازه یک و نیم مگی هست را محاسبه کرده و در یک ماتریس ذخیره میکند حال با محاسبه بیشترین چگالی طیفی در هر ایستگاه و ذخیره ان و ذخیره سماره سطری که چگالی طیفی در ان قرار دارد یک ماتریس تشکیل میدهیم که ماکسیمم چگالی طیفی توان هر ایستگاه را در خود داراست حال با قرار دادن یک ترشهلد که بسته به مکان دریافت سیگنال و ضعیف و قوی بودن ان متفیر است وارد یک حلقه ای میشویم که با نیم مگ حرکت از فرکانسی که در ابتدای برنامه از کاربر گرفته است رو به جلو حرکت میکند و با مقایسه چگالی طیفی سیگنال دریافتی با ماکسیمم چگالی طیفی ان ایستگاه که در محدوده حرکت قرار دارد درصورتی که چگالی طیفی سیگنال دریافتی به حد معینی از ماکسیمم ایستگاه رسیده باشد ان فرکانس را بعنوان ایستگاه اعلام میکند اما در حالت محاسبه ایستگاه قبلی تنها تفاوت گام های نیم مگ منفی و روبه عقب میباشد

```
clc
close all
clear all
f_min=88e6;
f_max=108e6;
l=2^18;
fs=3000000;
```

```
f_final=fs/2;
i_end=(f_max-f_min)/f_final;
f1=input('input CenterFreq: ');
for i=1:(fix(i_end)+1)
    fc=f_min+(i-1)*f_final;
    if fc > f_max
        break;
    end

sdr=comm.SDRRTLReceiver(...
    'CenterFrequency',fc,...
    'EnableTunerAGC',true,...
    'SampleRate',fs,...
    'SamplesPerFrame',l,...
    'OutputDataType','double');

data=step(sdr);
y=fft(data);
T=1/fs;
p2=abs(y/l).^2;
p1=p2(1:l/2+1);
f=fs*(0:(l/2))/l+fc;
[m,n]=size(p1);
main_mat(1:m,i)=p1;
f_mat(1:m,i)=f';
end

[max_of_den,m_max]=max(main_mat);
[max_of_energy,m_mmax]=max(max(main_mat));
[min_of_energy,m_mmin]=min(max(main_mat));
teresh=.99*max_of_den;
i=0.5e6;
while 1
    f1=f1+i;
    sdr = comm.SDRRTLReceiver(...
    'CenterFrequency', fc, ...
    'SampleRate',      fs, ...
    'SamplesPerFrame', 2^18, ...
    'OutputDataType',  'double');
    s=step(sdr);
    y=fft(s);
    y=(abs(y)).^2;
    if(y>teresh)
      f2=f1;
      break;
    end
    if(f1>108e6)
       f2=108e6;
       break
    end

end
```