ECE 251C Project

Name: Rishav Karki
PID: A59018992

Option 1:

Digitally down convert each channel to baseband and then filter the complex baseband sequence with a pair of FIR low pass filters that separates adjacent channels and down sample 8-to-1 to obtain output channels at 2-samples per symbol. Implement a FIR low-pass filter operating at 80 MHz sample rate with passbands of ±3.0 MHz and stopbands of ±7.0 MHz. Each channel is digitally translated from the 7 center frequencies, filtered and down sampled 8-to-1.

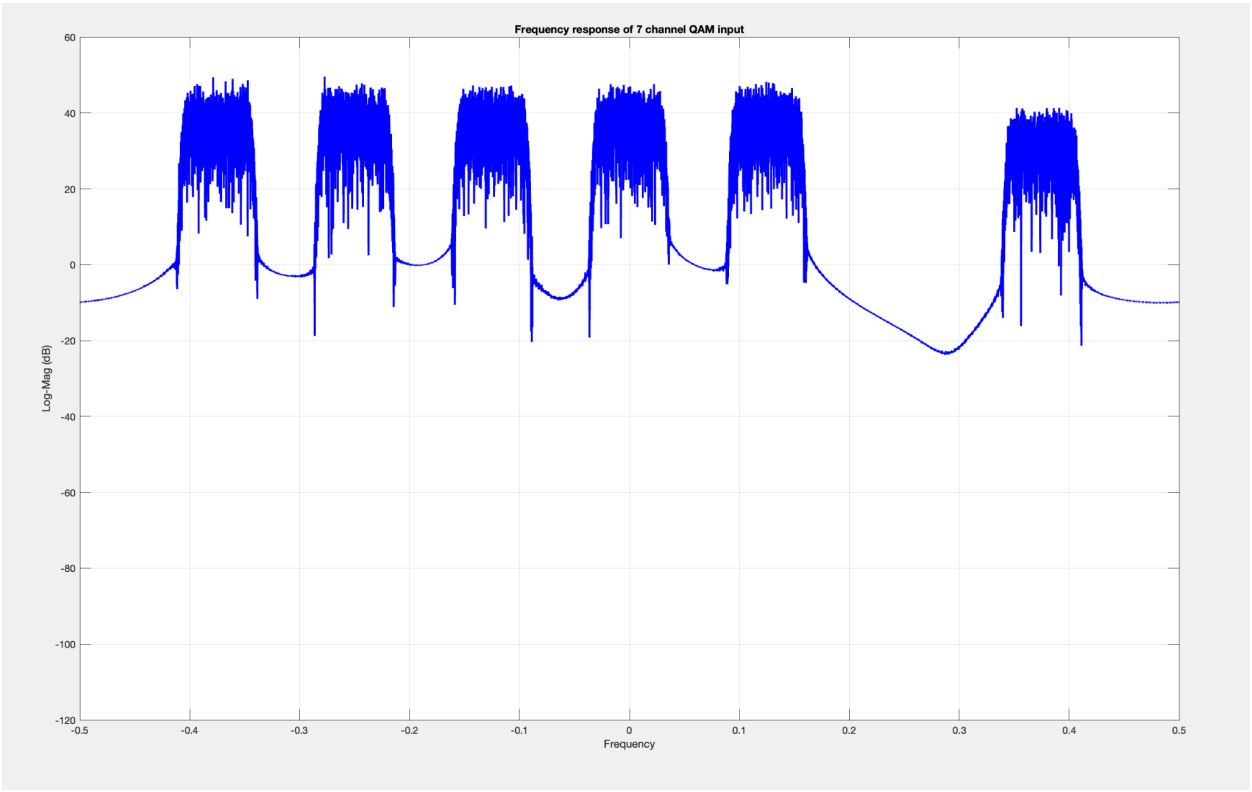
To down convert each channel to baseband, I used 7 complex heterodynes

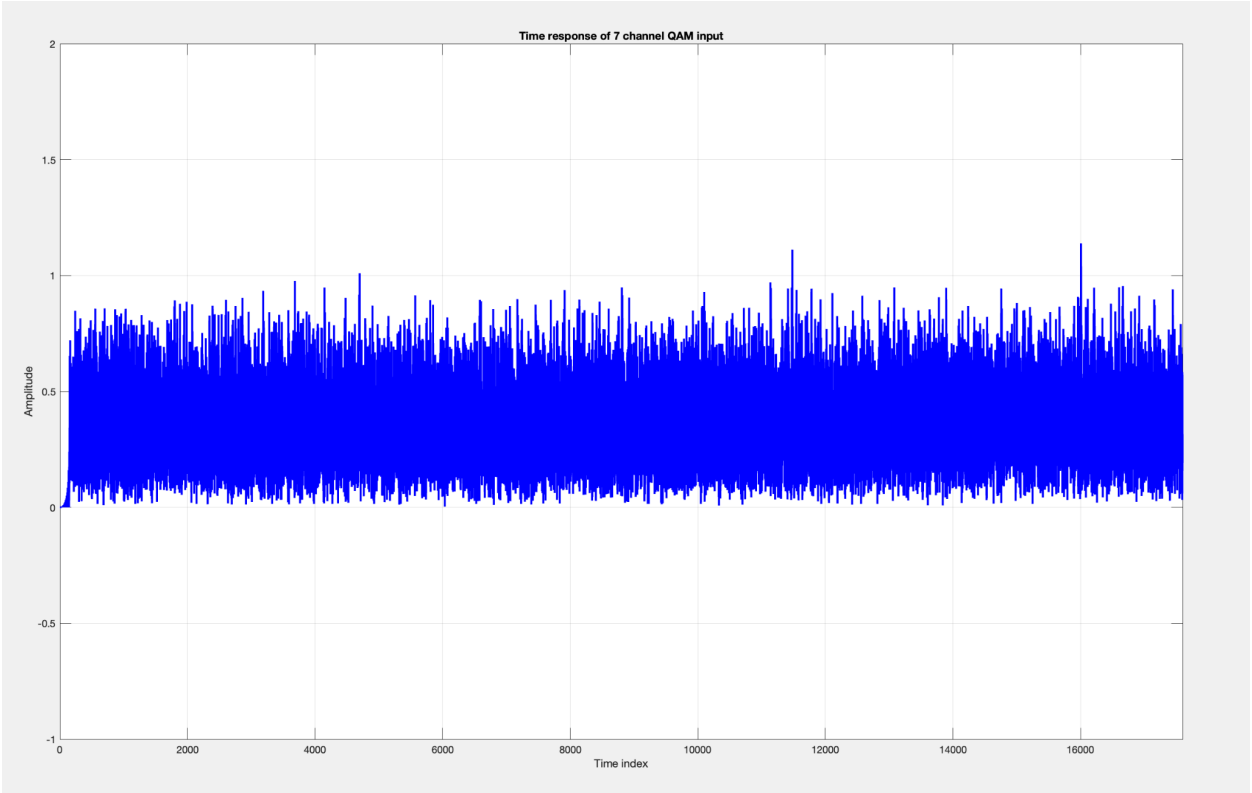
```
x3=zeros(7,16*N);
for k=1:7
   x3(k,:) = x2.*exp(-j*2*pi*(0:16*N-1)*ff(k));
End
```

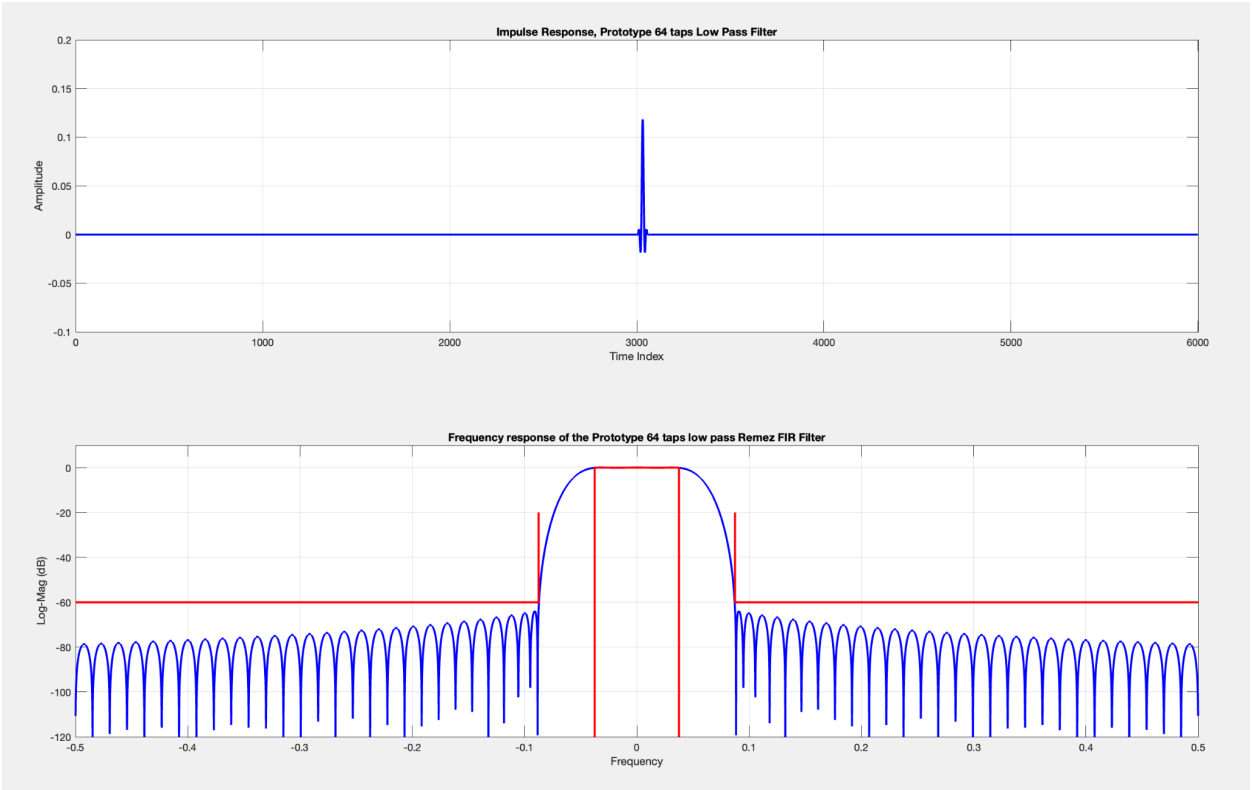
I designed a low pass FIR filter using the Remez algorithm, with N=64 taps to meet the desired filter specification.

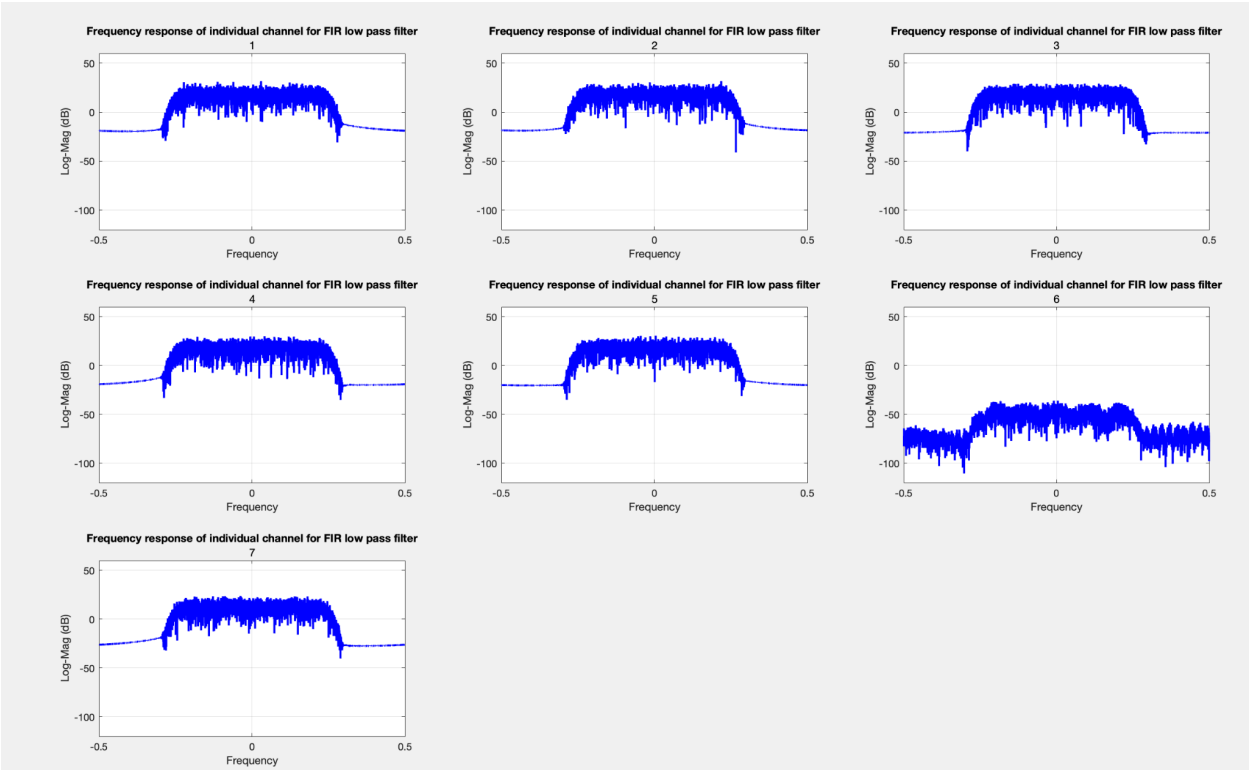
Coefficients :

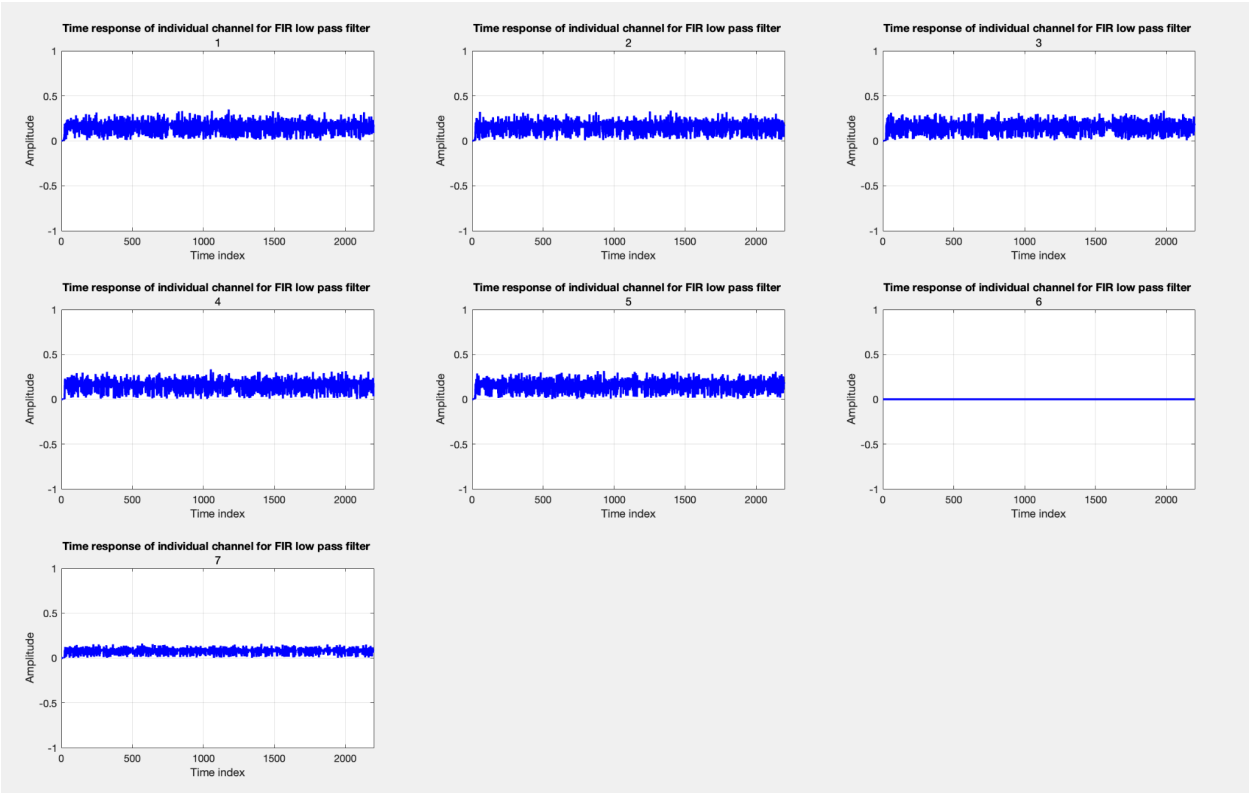
h = [-0.0002 -0.0004 -0.0006 -0.0007 -0.0006 -0.0003 0.0003 0.0013 0.0025
0.0037 0.0048 0.0055 0.0054 0.0042 0.0019 -0.0015 -0.0058 -0.0105 -0.0148
-0.0178 -0.0187 -0.0165 -0.0107 -0.0009 0.0126 0.0293 0.0480 0.0673 0.0855
0.1010 0.1122 0.1181 0.1181 0.1122 0.1010 0.0855 0.0673 0.0480 0.0293
0.0126 -0.0009 -0.0107 -0.0165 -0.0187 -0.0178 -0.0148 -0.0105 -0.0058 -0.0015
0.0019 0.0042 0.0054 0.0055 0.0048 0.0037 0.0025 0.0013 0.0003 -0.0003
-0.0006 -0.0007 -0.0006 -0.0004 -0.0002]


Frequency response of 7 channel QAM input
Log-Mag (dB)
Frequency
60
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5


Time response of 7 channel QAM input
Amplitude
Time index
2
1.5
1
0.5
0
-0.5
-1
0
2000
4000
6000
8000
10000
12000
14000
16000


Impulse Response, Prototype 64 taps Low Pass Filter
Amplitude
Time Index
0.2
0.15
0.1
0.05
0
-0.05
-0.1
0
1000
2000
3000
4000
5000
6000
Frequency response of the Prototype 64 taps low pass Remez FIR Filter
Log-Mag (dB)
Frequency
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5


Frequency response of individual channel for FIR low pass filter
1
Log-Mag (dB)
Frequency
Frequency response of individual channel for FIR low pass filter
2
Log-Mag (dB)
Frequency
Frequency response of individual channel for FIR low pass filter
3
Log-Mag (dB)
Frequency
Frequency response of individual channel for FIR low pass filter
4
Log-Mag (dB)
Frequency
Frequency response of individual channel for FIR low pass filter
5
Log-Mag (dB)
Frequency
Frequency response of individual channel for FIR low pass filter
6
Log-Mag (dB)
Frequency
Frequency response of individual channel for FIR low pass filter
7
Log-Mag (dB)
Frequency


Time response of individual channel for FIR low pass filter
1
Amplitude
Time index
Time response of individual channel for FIR low pass filter
2
Amplitude
Time index
Time response of individual channel for FIR low pass filter
3
Amplitude
Time index
Time response of individual channel for FIR low pass filter
4
Amplitude
Time index
Time response of individual channel for FIR low pass filter
5
Amplitude
Time index
Time response of individual channel for FIR low pass filter
6
Amplitude
Time index
Time response of individual channel for FIR low pass filter
7
Amplitude
Time index

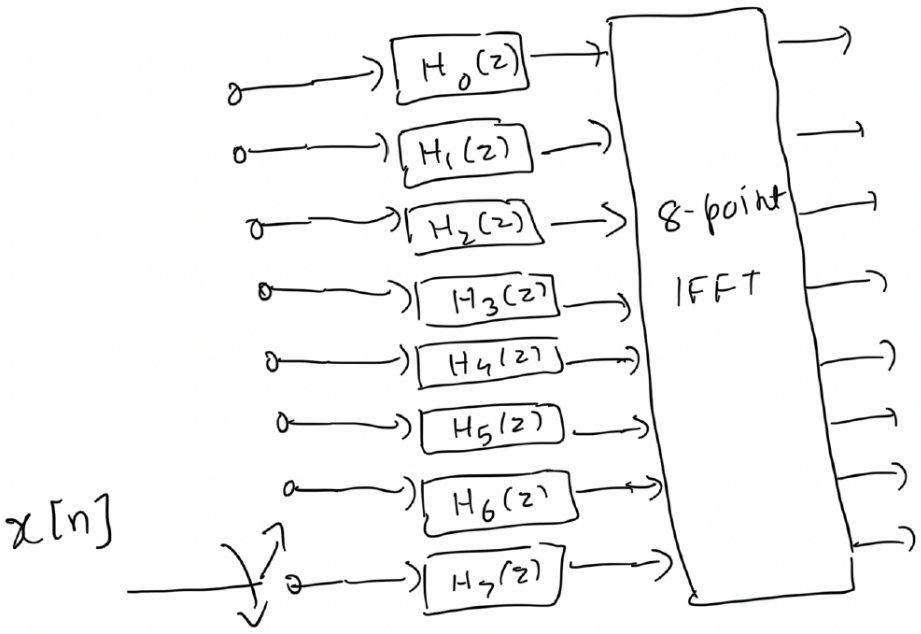
Option 2:

> Implement an 8-path polyphase FIR filter performing 8-to-1 down sampling. Each output from the 8-path filter is a down sampled version of time series aliased from multiples of 8 MHz and sampled at 10 MHz, 2-samples per symbol.

I designed a prototype low pass FIR filter using the Remez algorithm, with N=64 taps to meet the desired filter specification. I then partitioned the filter into an 8 path polyphase filter that performs 8-to-1 down sampling.

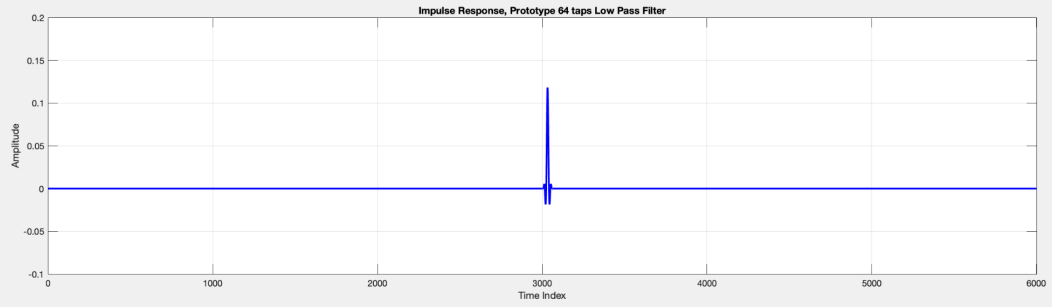
h = [-0.0002 -0.0004 -0.0006 -0.0007 -0.0006 -0.0003 0.0003 0.0013 0.0025
0.0037 0.0048 0.0055 0.0054 0.0042 0.0019 -0.0015 -0.0058 -0.0105 -0.0148
-0.0178 -0.0187 -0.0165 -0.0107 -0.0009 0.0126 0.0293 0.0480 0.0673 0.0855
0.1010 0.1122 0.1181 0.1181 0.1122 0.1010 0.0855 0.0673 0.0480 0.0293
0.0126 -0.0009 -0.0107 -0.0165 -0.0187 -0.0178 -0.0148 -0.0105 -0.0058 -0.0015
0.0019 0.0042 0.0054 0.0055 0.0048 0.0037 0.0025 0.0013 0.0003 -0.0003
-0.0006 -0.0007 -0.0006 -0.0004 -0.0002]

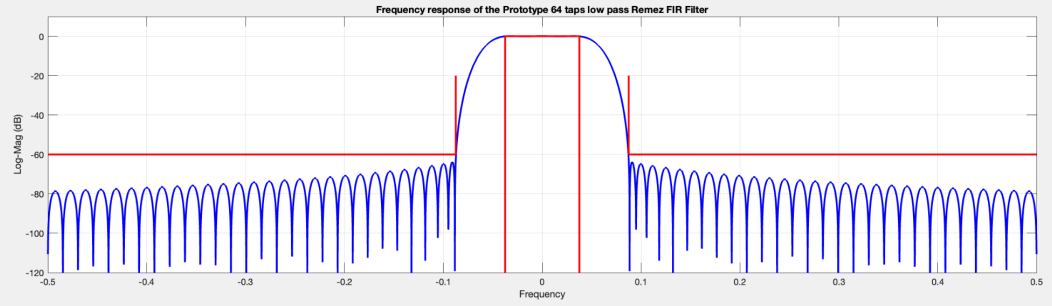
Using the Noble identity, I designed the 8-to-1 downsampler for the polyphase FIR filter. I used an IFFT on the outputs of each path to obtain the signals from each channel aliased to baseband.

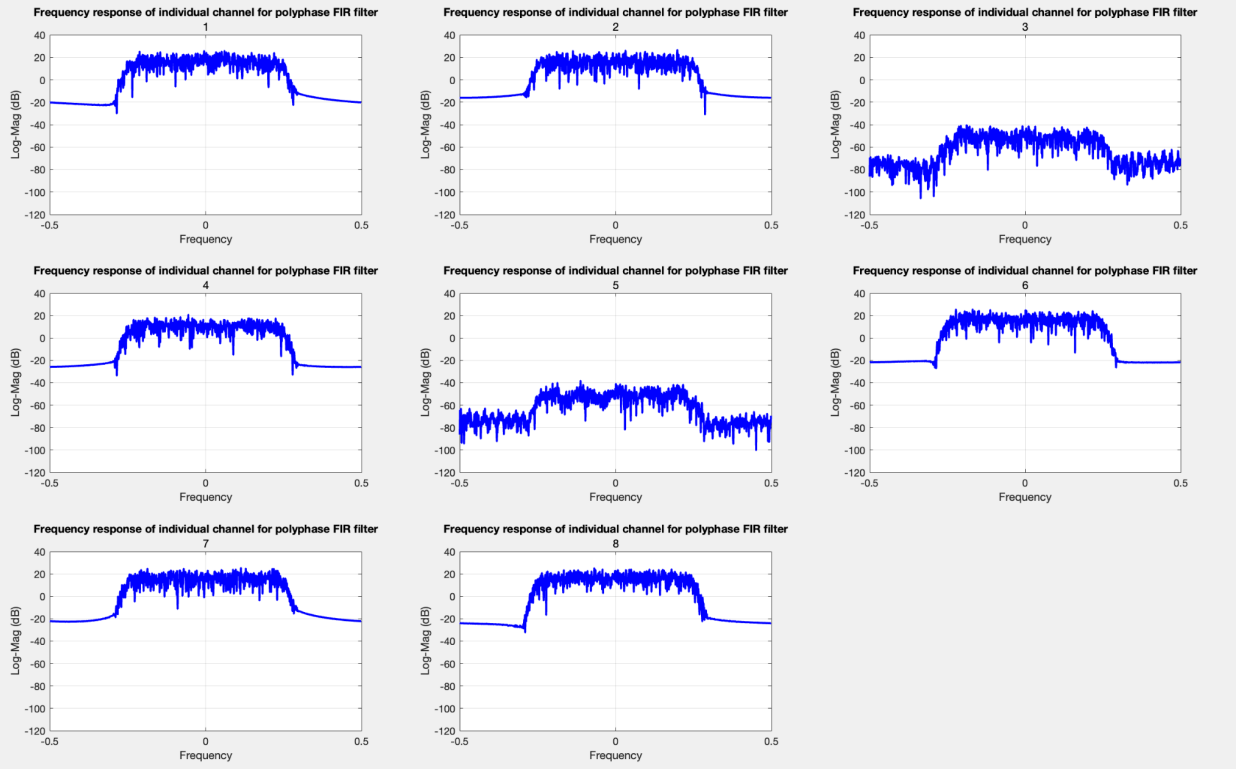



```
%filter QAM signal
x2 = x2_prev;
x2_size = size(x2);
x2_len = x2_size(2);
y4_fir_poly=zeros(8, floor(x2_len/8));
v1=zeros(8,floor(x2_len/8));
m=1;
for n=1:8:8*floor(x2_len/8)
    v0=fliplr(x2(n:n+7)).';
    reg=[v0 reg(:,1:(taps/8 - 1))];
        for k=1:8
          v1(k,m)=reg(k,:)*h2_8(k,:)';
```

```
        end
       %ifft
    %y0(m)=sum(v1(:,m));
    y4_fir_poly(:,m) = ifft(v1(:,m),8)';
    m=m+1;
end
```


Impulse Response, Prototype 64 taps Low Pass Filter
Amplitude
Time Index


Frequency response of the Prototype 64 taps low pass Remez FIR Filter
Log-Mag (dB)
Frequency

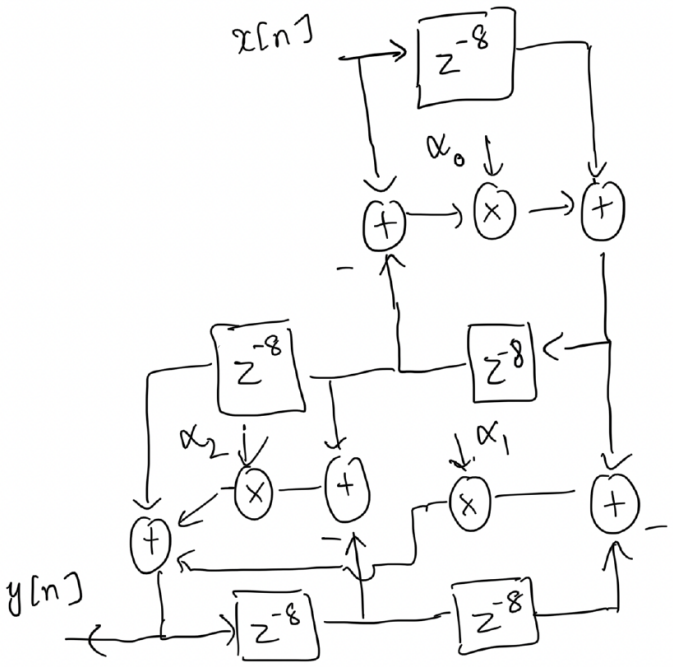

Frequency response of individual channel for polyphase FIR filter
1
2
3
4
5
6
7
8
Log-Mag (dB)
Frequency

Option 3: Digitally down convert each channel to baseband and then filter the complex baseband sequence with a pair of IIR linear phase low pass filters that separates adjacent channels and down sample 8-to-1 to obtain output channels at 2-samples per symbol.

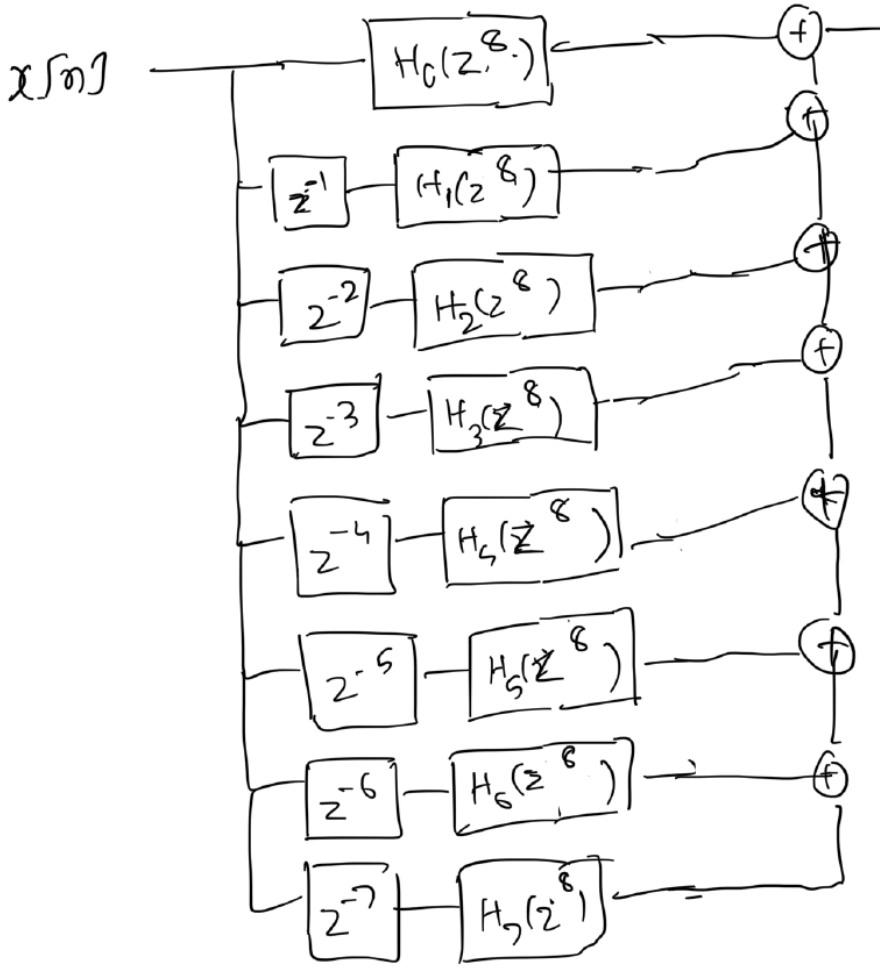
I used the lineardesign_2 script to design an 8 path linear polyphase recursive filter where each path has all pass filters with 3 coefficients (24 delays). The minimum normalized passband frequency for this script is 0.0625 (5 MHz), and I set the stopband at 0.0875 (7 MHz). I didn't implement the downsampler in this filter using Nobel's identity (Option 4), but I used the complex heterodyne operation to digitally down convert the signal for the 7 channels before sending it to 7 copies of the same IIR filter.

Coefficients of the filter :

```
% Denominator Polynomials, path-1
% branch 0 - 24 Delays,     Z^(-24)
% Denominator Polynomials, path-1
a_path(2,1,paths+1) = 0.3000581202376679;
a_path(2,2,paths+1) = -0.1900402095694197;
a_path(2,2,2*paths+1) = 0.02633139911243305;
% Denominator Polynomials, path-2
a_path(3,1,paths+1) = 0.4222035911739405;
a_path(3,2,paths+1) = -0.1989238080170467;
a_path(3,2,2*paths+1) = 0.03024784889553085;
% Denominator Polynomials, path-3
a_path(4,1,paths+1) = 0.5254281202136269;
a_path(4,2,paths+1) = -0.1852117598974453;
a_path(4,2,2*paths+1) =  0.02859219142615069;
% Denominator Polynomials, path-4
a_path(5,1,paths+1) = 0.6212350941176464;
a_path(5,2,paths+1) = -0.1599165480534598;
a_path(5,2,2*paths+1) = 0.02421203546513172;
% Denominator Polynomials, path-5
a_path(6,1,paths+1) = 0.714275602892003;
a_path(6,2,paths+1) = -0.1271271167937832;
a_path(6,2,2*paths+1) = 0.01846501616252262;
% Denominator Polynomials, path-6
a_path(7,1,paths+1) = 0.807212346911608;
a_path(7,2,paths+1) = -0.0885934163493368;
a_path(7,2,2*paths+1) = 0.01216766845066991;
% Denominator Polynomials, path-7
a_path(8,1,paths+1) = 0.9019330309014866;
a_path(8,2,paths+1) = -0.0462528555078691;
a_path(8,2,2*paths+1) = 0.005875680228912086;
```

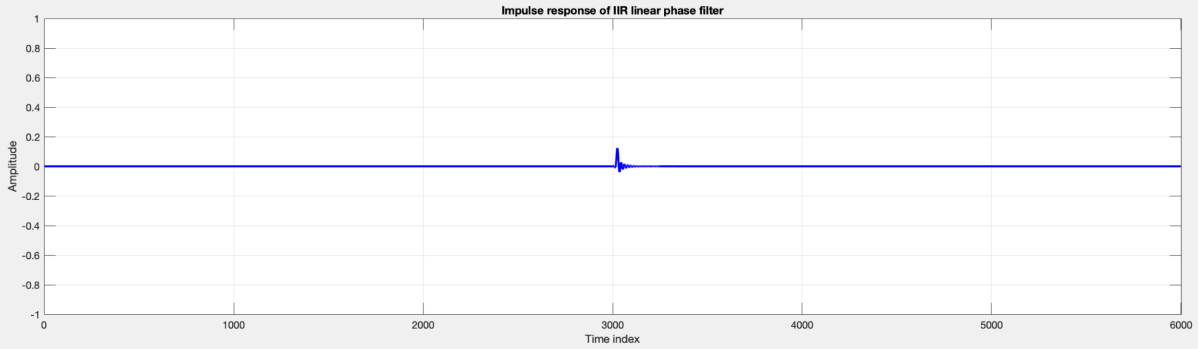


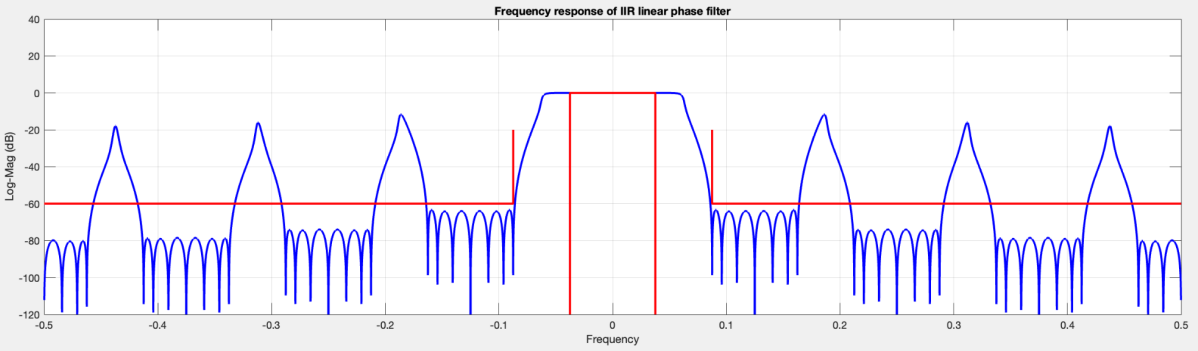
All Pass filter with
3 co-efficients $\alpha_0, \alpha_1, \alpha_2$.

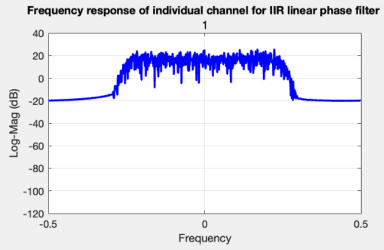


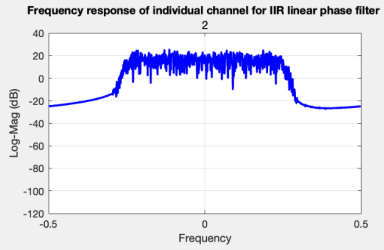
where $H_0(z^8)$ is z^{-24}

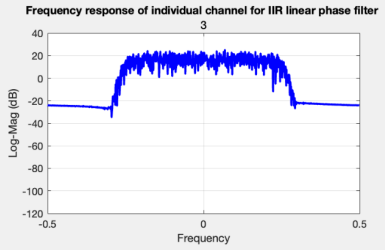
and $H_n(z^8)$ are all pass filters.

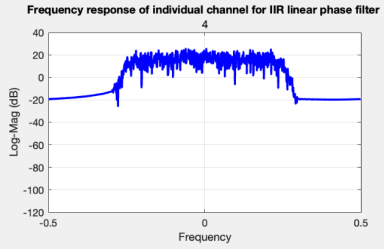

Impulse response of IIR linear phase filter
Amplitude
Time index

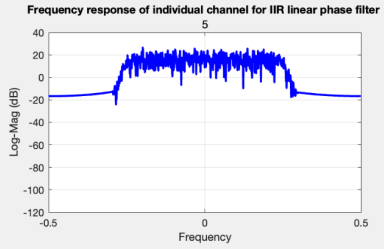

Frequency response of IIR linear phase filter
Log-Mag (dB)
Frequency

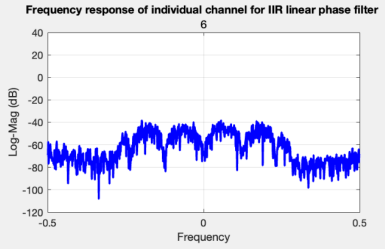

Frequency response of individual channel for IIR linear phase filter
1
Log-Mag (dB)
Frequency

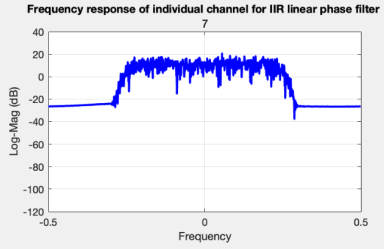

Frequency response of individual channel for IIR linear phase filter
2
Log-Mag (dB)
Frequency

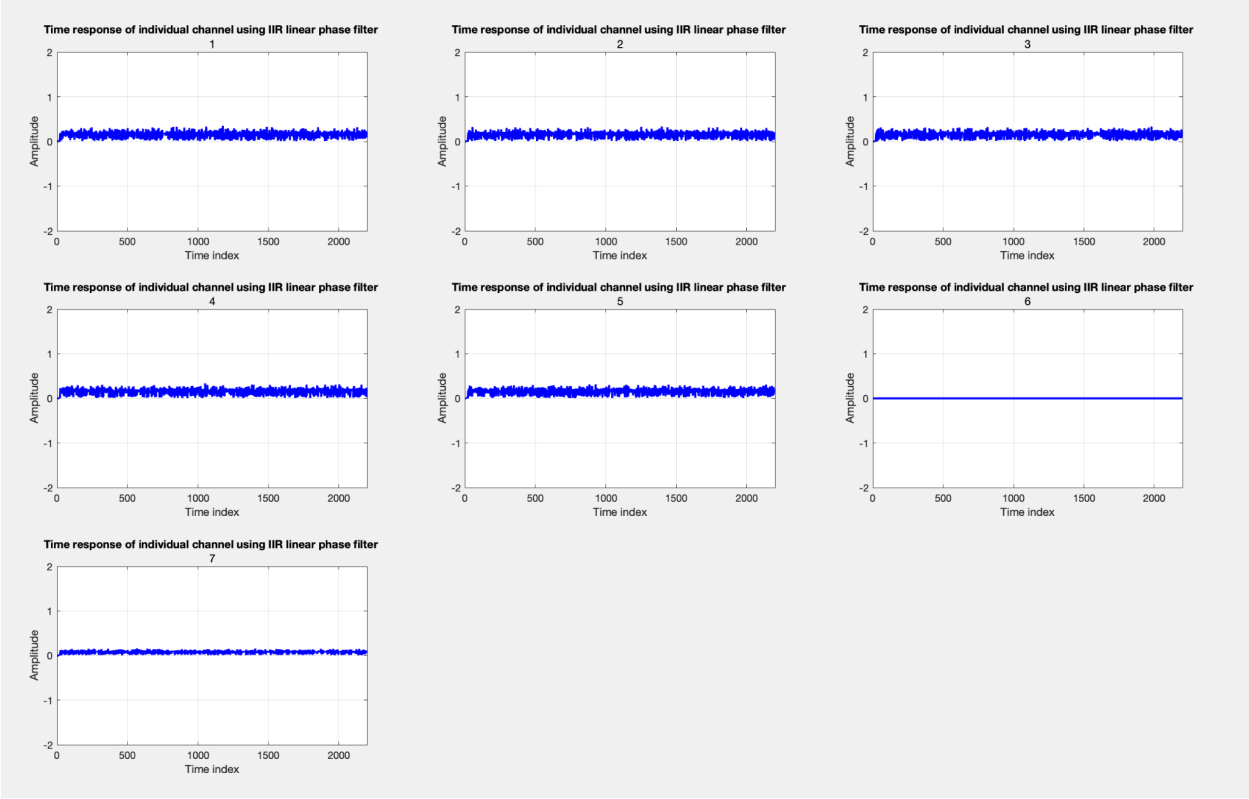

Frequency response of individual channel for IIR linear phase filter
3
Log-Mag (dB)
Frequency


Frequency response of individual channel for IIR linear phase filter
4
Log-Mag (dB)
Frequency


Frequency response of individual channel for IIR linear phase filter
5
Log-Mag (dB)
Frequency


Frequency response of individual channel for IIR linear phase filter
6
Log-Mag (dB)
Frequency


Frequency response of individual channel for IIR linear phase filter
7
Log-Mag (dB)
Frequency



Option 4: Implement an 8-path polyphase linear phase IIR filter performing 8-to-1 down sampling. Each output from the 8-path filter is a down sampled version of time series aliased from multiples of 8 MHz and sampled at 10 MHz, 2-samples per symbol.

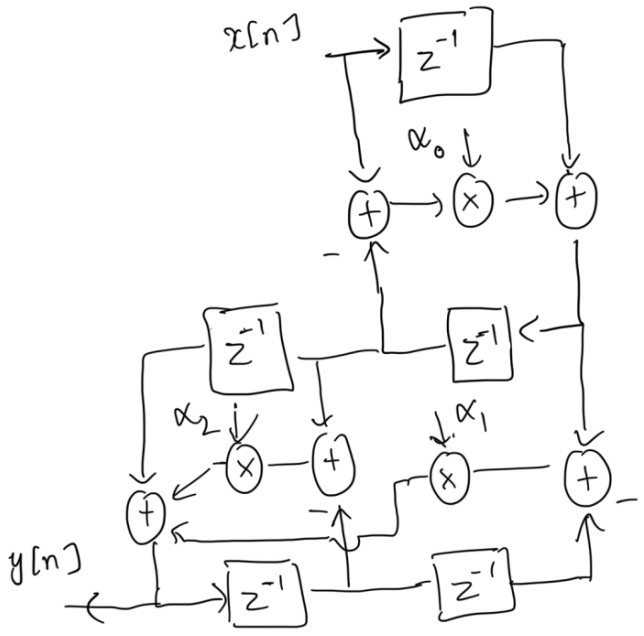
I used the lineardesign_2 script to design an 8 path linear polyphase recursive filter where each path has all pass filters with 3 coefficients (24 delays). The minimum normalized passband frequency for this script is 0.0625 (5 MHz), and I set the stopband at 0.0875 (7 MHz).

Coefficients of the filter:

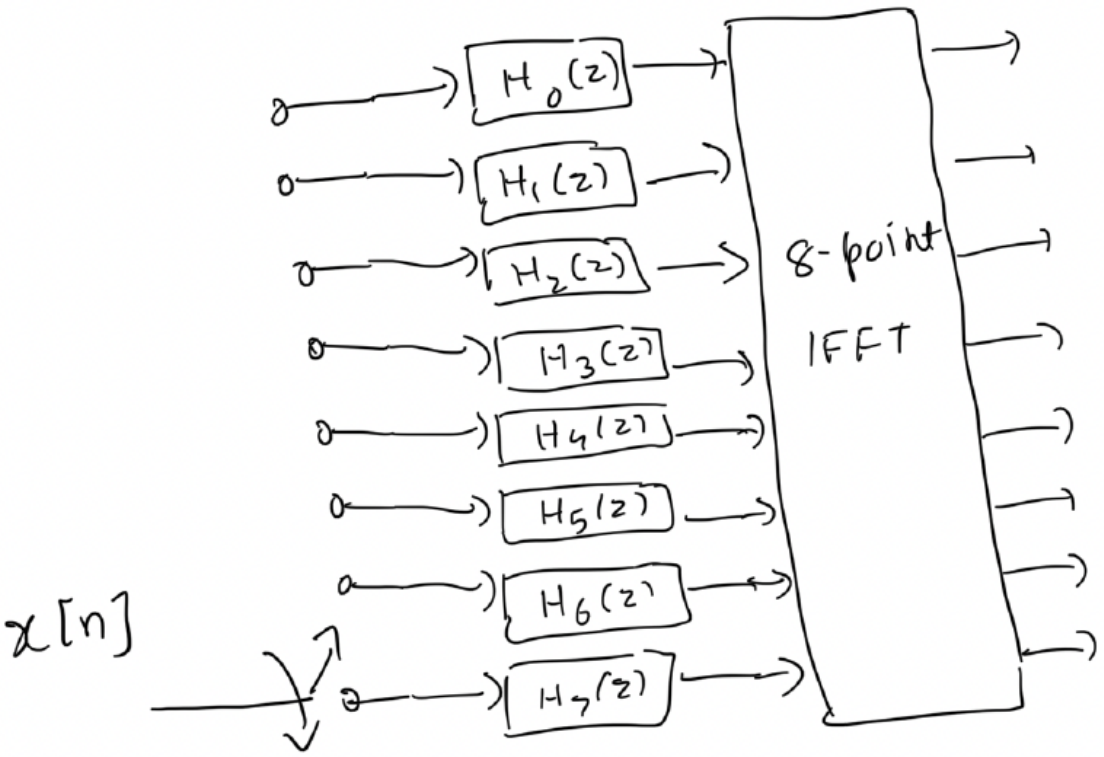
```
% Denominator Polynomials, path-1
% branch 0 - 24 Delays,     Z^(-24)
% Denominator Polynomials, path-1
a_path(2,1,paths+1) = 0.3000581202376679;
a_path(2,2,paths+1) = -0.1900402095694197;
```

```
a_path(2,2,2*paths+1) = 0.02633139911243305;
% Denominator Polynomials, path-2
a_path(3,1,paths+1) = 0.4222035911739405;
a_path(3,2,paths+1) = -0.1989238080170467;
a_path(3,2,2*paths+1) = 0.03024784889553085;
% Denominator Polynomials, path-3
a_path(4,1,paths+1) = 0.5254281202136269;
a_path(4,2,paths+1) = -0.1852117598974453;
a_path(4,2,2*paths+1) =  0.02859219142615069;
% Denominator Polynomials, path-4
a_path(5,1,paths+1) = 0.6212350941176464;
a_path(5,2,paths+1) = -0.1599165480534598;
a_path(5,2,2*paths+1) = 0.02421203546513172;
% Denominator Polynomials, path-5
a_path(6,1,paths+1) = 0.714275602892003;
a_path(6,2,paths+1) = -0.1271271167937832;
a_path(6,2,2*paths+1) = 0.01846501616252262;
% Denominator Polynomials, path-6
a_path(7,1,paths+1) = 0.807212346911608;
a_path(7,2,paths+1) = -0.08885934163493368;
a_path(7,2,2*paths+1) = 0.01216766845066991;
% Denominator Polynomials, path-7
a_path(8,1,paths+1) = 0.9019330309014866;
a_path(8,2,paths+1) = -0.0462528555078691;
a_path(8,2,2*paths+1) = 0.005875680228912086;
```

Using the Noble identity, I designed the 8-to-1 downsampler for the polyphase IIR filter. I used an IFFT on the outputs of each path to obtain the signals from each channel aliased to baseband.



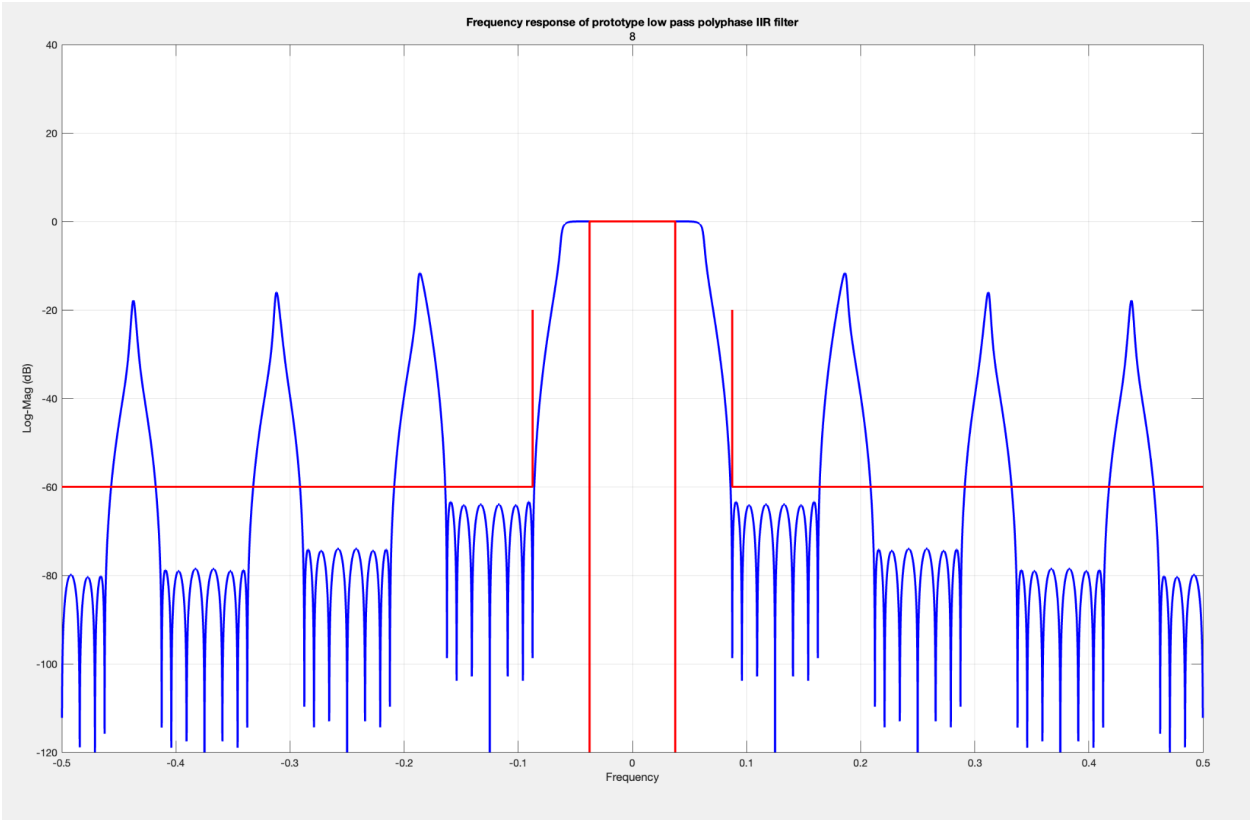
All Pass filter with
3 co-efficients $\alpha_0, \alpha_1, \alpha_2$.

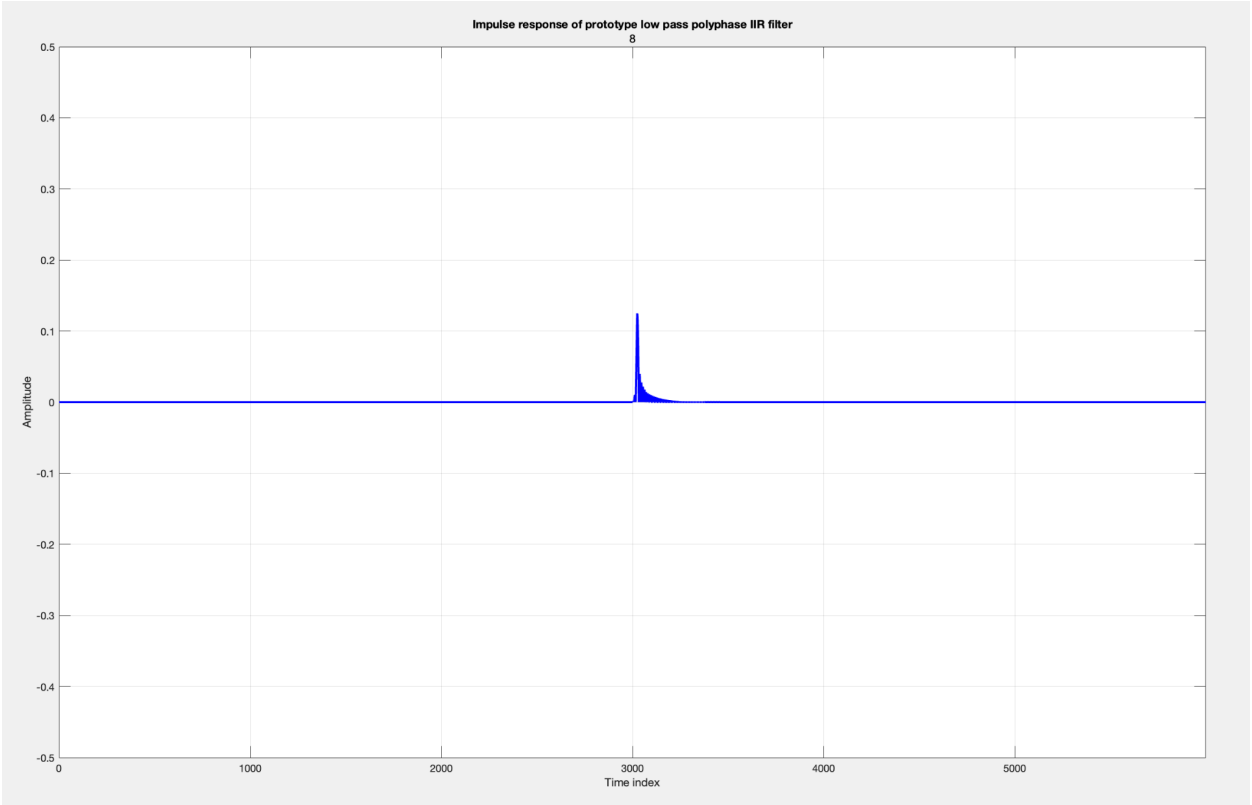


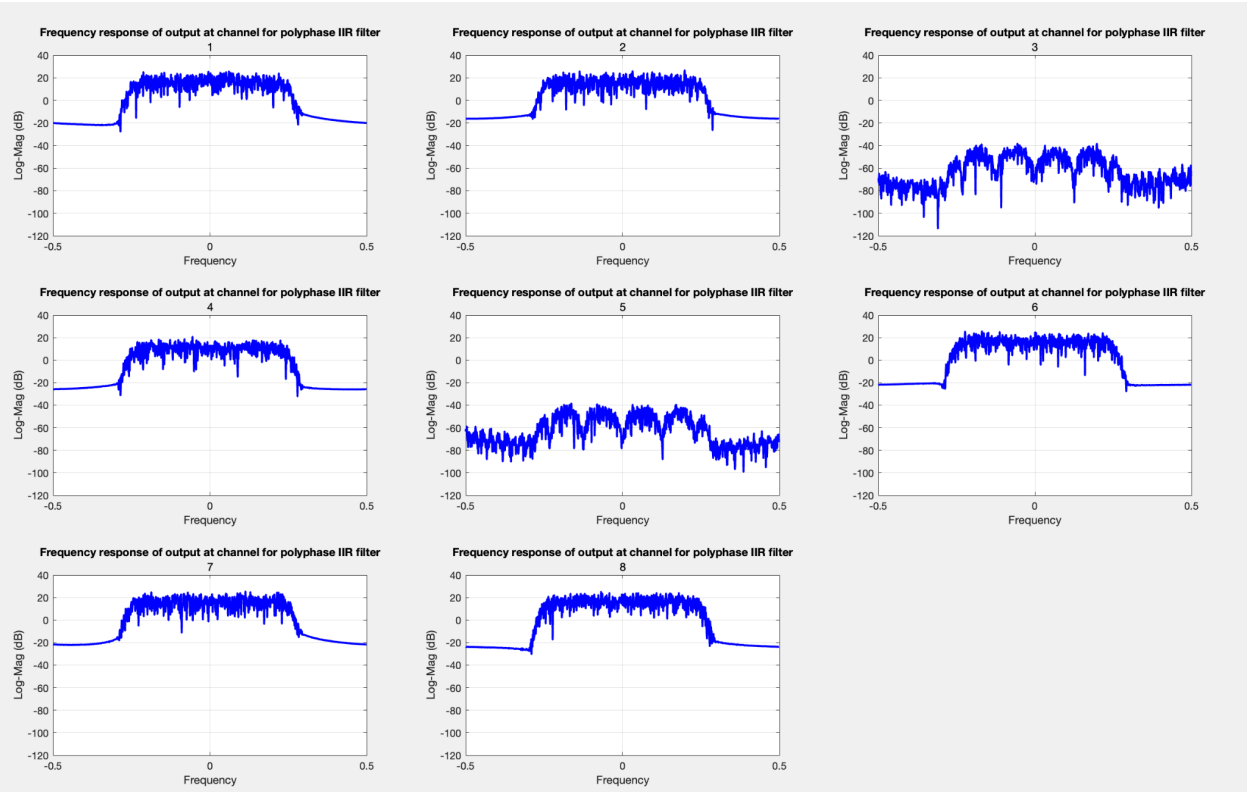
where $H_0(z)$ is z^{-3}

and $H_k(z)$ are all pass filters

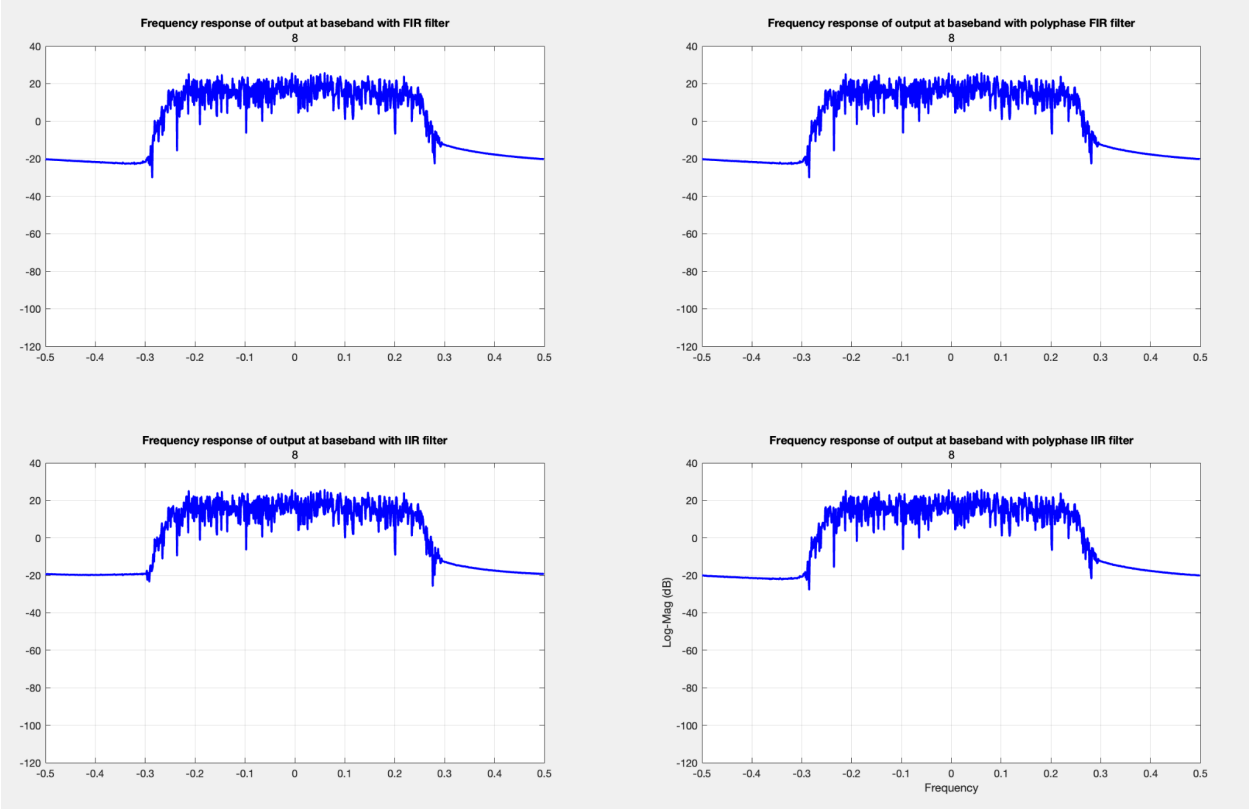
```
reg = zeros(paths,3,3*paths);
x0 = x2;
sm = zeros(paths,2);
x0_size = size(x0);
x0_len = x0_size(2);
len_y = floor(x0_len/paths);
y_iir_poly=zeros(paths,len_y);
m = 1;
for n=paths:paths:paths*len_y
   for k=2:1:paths
      sm(k,1)=(dly(k-1)-reg(k,2,1*paths))*a_path(k,1,(paths +
1))+reg(k,1,1*paths);
      sm(k,2)=(sm(k,1)-reg(k,3,2*paths))*a_path(k,2,(2*paths +
1))+(reg(k,2,1*paths)-reg(k,3,1*paths))*a_path(k,2,(paths +
1))+reg(k,2,2*paths);
   end
sm(1,2) = reg(1,1,3*paths);
y_iir_poly(:,m) = ifft(sm(:,2),paths)';

for k=2:paths
  reg(k,1,paths + 1:3*paths)= reg(k,1,1:2*paths);
  reg(k,2,paths + 1:3*paths)= reg(k,2,1:2*paths);
  reg(k,3,paths + 1:3*paths)= reg(k,3,1:2*paths);
  reg(k,1,paths)= dly(k-1);
  reg(k,2,paths)=sm(k,1);
  reg(k,3,paths)=sm(k,2);

 end
reg(1,1,paths+1:3*paths)= reg(1,1,1:2*paths);
%dly= [x0(n) dly(1:paths-2)];
%dly= fliplr(x0(n:n+paths-2));
dly= fliplr(x0(n-paths+1:n-1));
if n>paths
reg(1,1,paths)= x0(n-paths);
end
%reg(1,1,paths)= x0(n+paths-1);
m = m + 1;
end
```

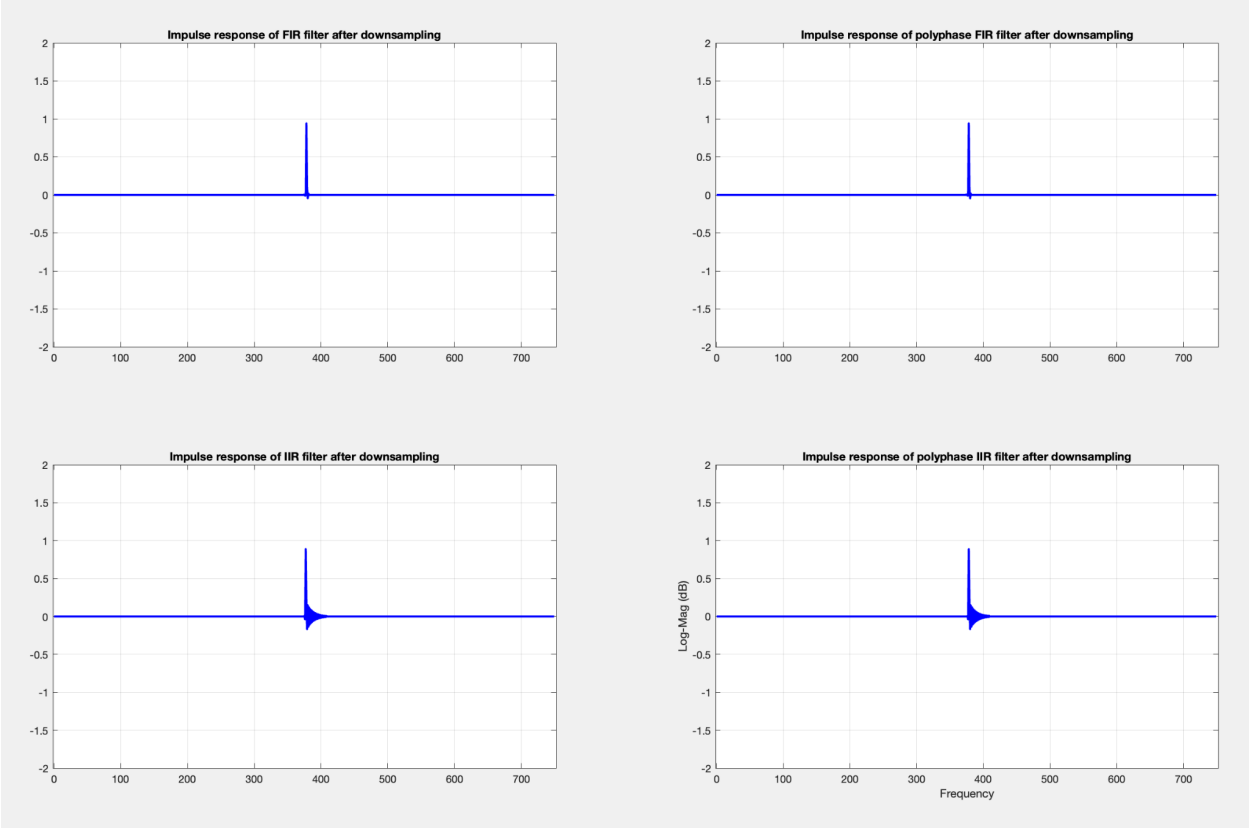

Frequency response of prototype low pass polyphase IIR filter
8
Log-Mag (dB)
Frequency
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5

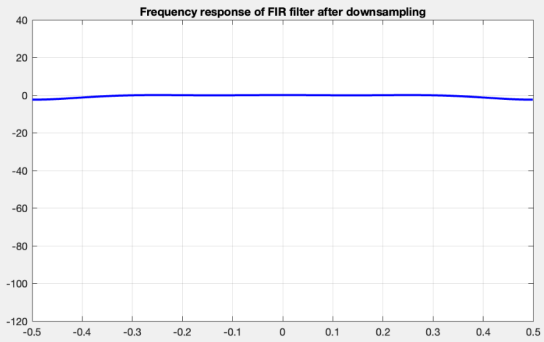

Impulse response of prototype low pass polyphase IIR filter
8
Amplitude
Time index
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
1000
2000
3000
4000
5000

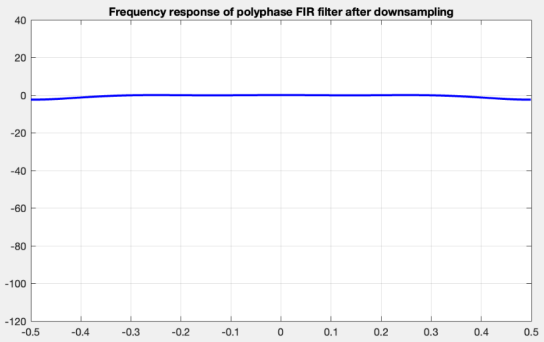

Frequency response of output at channel for polyphase IIR filter
1
Log-Mag (dB)
Frequency
Frequency response of output at channel for polyphase IIR filter
2
Log-Mag (dB)
Frequency
Frequency response of output at channel for polyphase IIR filter
3
Log-Mag (dB)
Frequency
Frequency response of output at channel for polyphase IIR filter
4
Log-Mag (dB)
Frequency
Frequency response of output at channel for polyphase IIR filter
5
Log-Mag (dB)
Frequency
Frequency response of output at channel for polyphase IIR filter
6
Log-Mag (dB)
Frequency
Frequency response of output at channel for polyphase IIR filter
7
Log-Mag (dB)
Frequency
Frequency response of output at channel for polyphase IIR filter
8
Log-Mag (dB)
Frequency

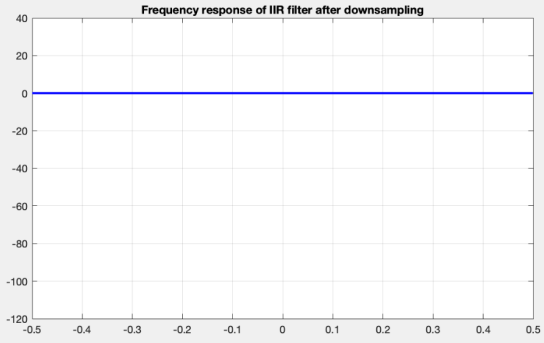
Clean Up:

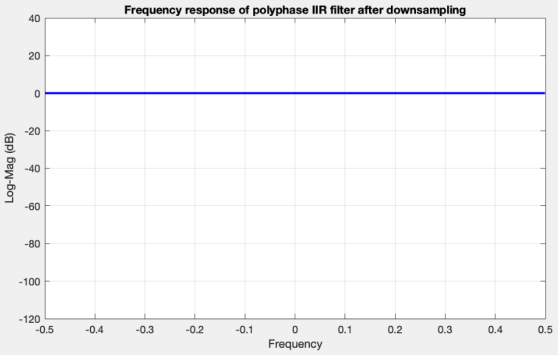

Frequency response of output at baseband with FIR filter
Frequency response of output at baseband with polyphase FIR filter
Frequency response of output at baseband with IIR filter
Frequency response of output at baseband with polyphase IIR filter
Log-Mag (dB)
Frequency

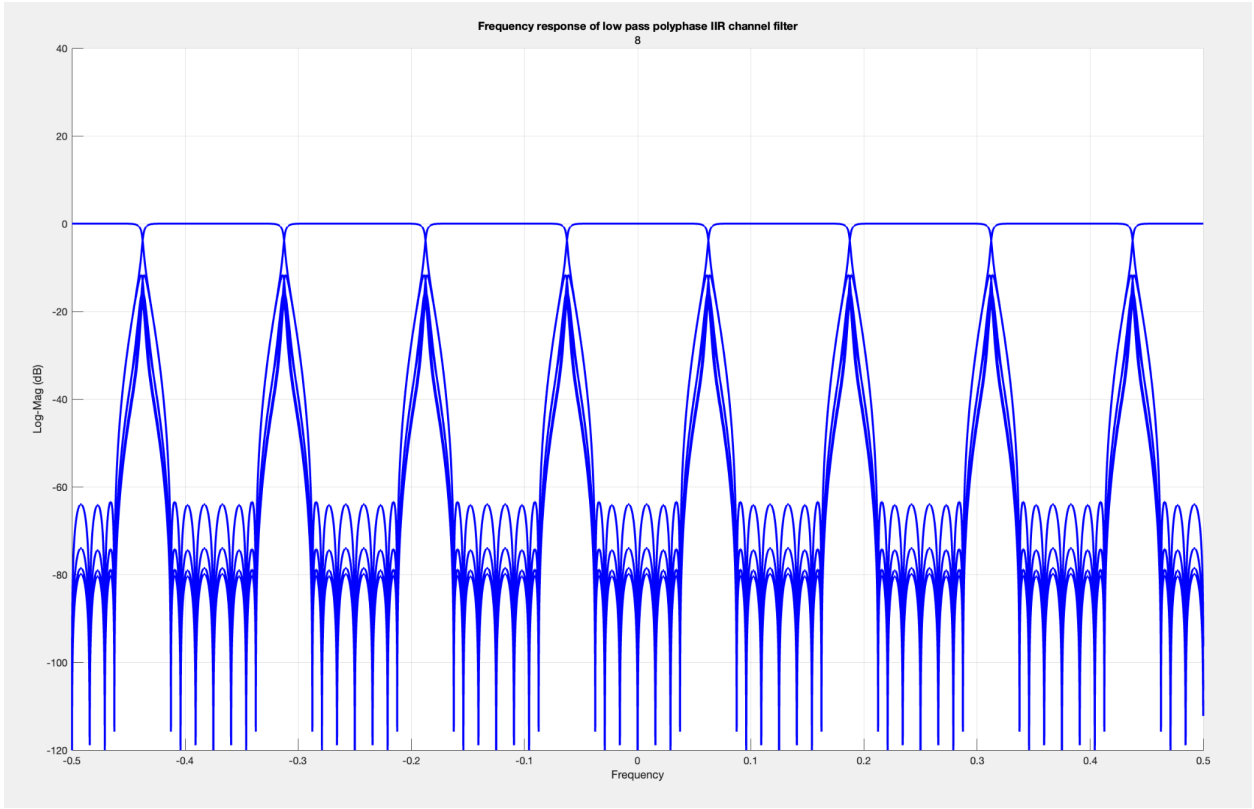

Impulse response of FIR filter after downsampling
Impulse response of polyphase FIR filter after downsampling
Impulse response of IIR filter after downsampling
Impulse response of polyphase IIR filter after downsampling
Log-Mag (dB)
Frequency

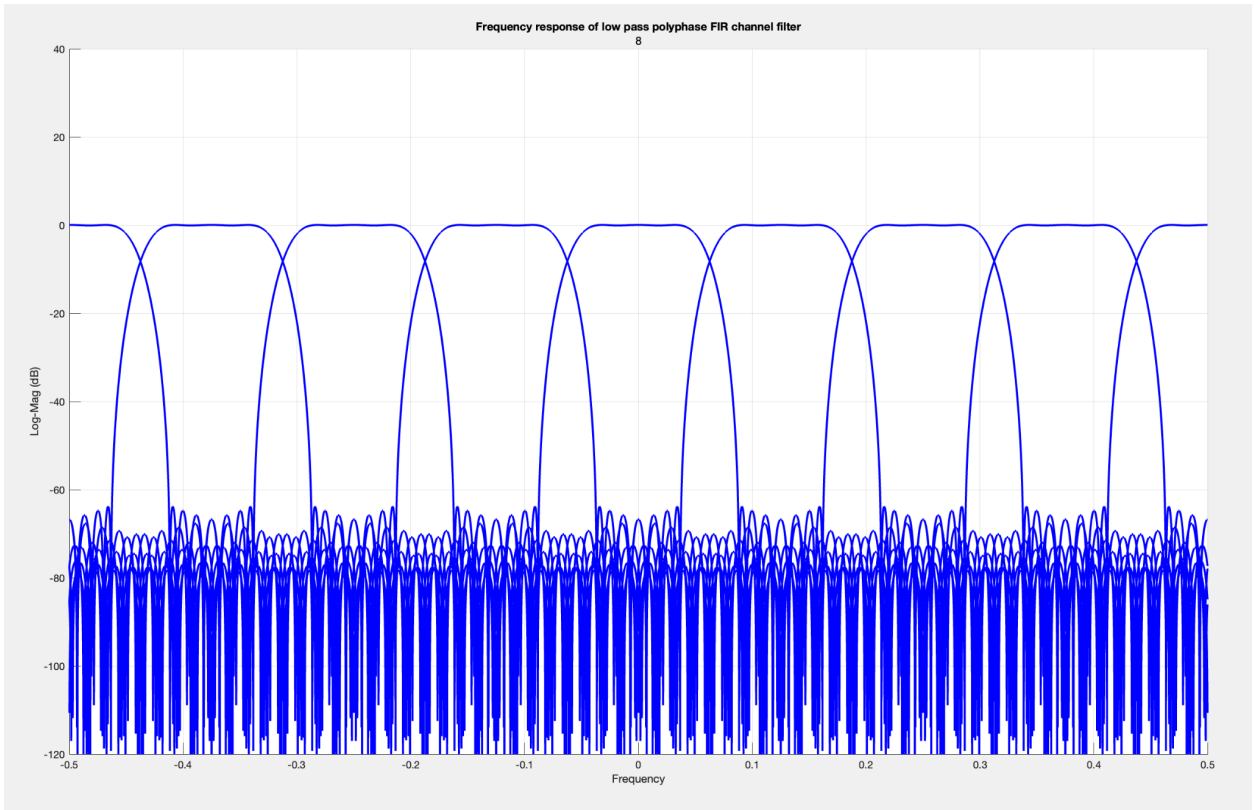

Frequency response of FIR filter after downsampling
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5


Frequency response of polyphase FIR filter after downsampling
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5


Frequency response of IIR filter after downsampling
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5


Frequency response of polyphase IIR filter after downsampling
Log-Mag (dB)
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5
Frequency


Frequency response of low pass polyphase IIR channel filter
8
Log-Mag (dB)
Frequency
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5


Frequency response of low pass polyphase FIR channel filter
8
Log-Mag (dB)
Frequency
40
20
0
-20
-40
-60
-80
-100
-120
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5

Computational workloads:

Option 1: We use 7 complex heterodynes to digitally down convert the signal to baseband and then use the low pass FIR filter 7 times on each baseband downconverted time series. Each FIR filter has 64 taps. Therefore, we perform 64*7 = 448 multiplications per input data point along with 7 multiplications for the heterodyne operation. The filter requires 64 multiplications per input data point per output channel. With the heterodyne operation, the filter requires 64+1 = 65 multiplications per input data point per output channel.

Option 2: We use an 8 path polyphase FIR filter where each path has 8 taps. Therefore, we perform 8 multiplications per input data point. For the 8 point IFFT, we would need 8*3 = 24 multiplications (NlogN) to get the 8 output channels, and we discard 1 channel to get the 7 desired channels. Therefore the filter requires 8/7 = 1.14 multiplications per input data point per output channel. With the IFFT operation, the filter requires 1.14 + (24)/(7*8) = 1.57 multiplications per input data point per output channel.

Option 3: We use an 8 path linear polyphase IIR filter where each path has all pass filters with 3 coefficients. We do not use the Noble identity to implement the down sampler in this case, and instead rely on the complex heterodyne operation to down convert the input. We use the same IIR filter 7 times on each baseband downconverted time series. Therefore, we perform 3*7*7 = 147 multiplications for each input sample point. The filter requires 3*7=21 multiplications per input data point per output channel. With the heterodyne operation, the filter requires 21+1 = 22 multiplications per input data point per output channel.

Option 4: We use an 8 path linear polyphase IIR filter where each path has all pass filters with 3 coefficients. Therefore, we perform 3*7 = 21 multiplications for 8 input sample points, or 2.7 multiplications per input sample point. For the 8 point IFFT, we would need 8*3 = 24 multiplications (NlogN) to get the 8 output channels, and we discard 1 channel to get the 7 desired channels. Therefore the filter requires 2.7/7 = 0.38 multiplications per input data point per output channel. With the IFFT operation, the filter requires 0.38 + (24)/(7*8) = 0.81 multiplications per input data point per output channel.

Therefore Option 1 is the least efficient and Option 4 is the most efficient.

Code:

```
h1=sinc(-10+1/16:1/16:10-1/16).*kaiser(319,6)';
x2=zeros(1,17600);
ff=[-3 -2 -1 0 1 2 3]/8;
aa=[ 1  1  1 1 1 0 0.5]/8;
N=1100;
for k=1:7
   x0=(floor(2*rand(1,N))-0.5)/0.5+j*(floor(2*rand(1,N))-0.5)/0.5;
```

```
   x1=zeros(1,16*N);
   x1(1:16:16*N)=x0;
   x1=filter(h1,1,x1);
   x2=x2+aa(k)*(x1.*exp(j*2*pi*(0:16*N-1)*ff(k)));
end
f_x2=fftshift(20*log10(abs(fft(x2,32768))));
figure (1)
plot((-0.5:1/32768:0.5-1/32768),f_x2,'b','linewidth',2)
grid on
axis([-0.5 0.5 -120 60])
title('Frequency response of 7 channel QAM input')
xlabel('Frequency')
ylabel('Log-Mag (dB)')
size_x2 = size(x2);
len_x2 = size_x2(2);
figure (2)
plot((0:len_x2-1),abs(x2),'b','linewidth',2)
grid on
axis([0 len_x2-1 -1 2])
title('Time response of 7 channel QAM input')
xlabel('Time index')
ylabel('Amplitude')
%%Option 1
%%Digitally down convert each channel to baseband
x3=zeros(7,16*N);
for k=1:7
   x3(k,:) = x2.*exp(-j*2*pi*(0:16*N-1)*ff(k));
end
%% Remez low pass FIR filter
fs = 80;
del_f = 4;
attenuation = 60;
taps = fs/del_f * (attenuation)/22;
taps = ceil(taps);
% Adjust taps to satisfy spec, taps is a multiple of 8 now %% 64
taps = taps + (8 - mod(taps,8)) + 8;
h = remez(taps-1,[0, 3, 7, 40]/40, {'myfrf', [1 1 0 0]}, [1 10]);
% Use Nyquist filter
%taps = 48;
%h=sinc(-2 + 1/(taps/4):1/(taps/4):2).*kaiser(taps,6)'/(sum(kaiser(taps,8))*1);
f_h=fftshift(20*log10(abs(fft(h,1024))));
size_h = size(h);
h_len = size_h(2);
%impulse response
x_impulse = [zeros(1,3000) 1 zeros(1,3000)];
%y_fir_impulse_len = 16*N+taps-1;
%y_fir_impulse =zeros(1,y_fir_impulse_len);
y_fir_impulse = zeros(1,6001);
reg = zeros(1,taps);
```

```
for n=1:1:6001
  reg=[x_impulse(n) reg(1:(taps - 1))];
  y_fir_impulse(n) = reg*h';
end
%y_fir_impulse = conv(h,x_impulse);
y_fir_impulse_size = size(y_fir_impulse);
y_fir_impulse_len = y_fir_impulse_size(2);
y_fir_impulse_down = zeros(1, floor(y_fir_impulse_len/8));
for k=1:floor(y_fir_impulse_len/8)
   y_fir_impulse_down(k) = y_fir_impulse(8*k);
end
figure (3)
subplot(2,1,1)
plot(0:6000,y_fir_impulse,'b','linewidth',2)
axis([-2 6002 -0.1 0.2])
grid on
title('Impulse Response, Prototype 64 taps Low Pass Filter')
xlabel('Time Index')
ylabel('Amplitude')
subplot(2,1,2)
plot((-0.5:1/8192:0.5-1/8192),fftshift(20*log10(abs(fft(y_fir_impulse,8192)))),
'b','linewidth',2)
hold on
plot([-3/80 -3/80 3/80 3/80],[-120 0 0 -120],'r','linewidth',2)
plot([-0.5 -7/80 -7/80],[-60 -60 -20],'r','linewidth',2)
plot([ 7/80 7/80 0.5],[-20 -60 -60],'r','linewidth',2)
hold off
grid on
axis([-0.5 0.5 -120 10])
title('Frequency response of the Prototype 64 taps low pass Remez FIR Filter')
xlabel('Frequency')
ylabel('Log-Mag (dB)')
y3_len = 16*N;
reg = zeros(1,taps);
y3=zeros(7,y3_len);
y3_down = zeros(7, floor(y3_len/8));
for k=1:7
   reg = zeros(1,taps);
   for n=1:1:y3_len
       reg=[x3(k,n) reg(1:(taps - 1))];
       y3(k,n) = reg*h';
   end
end
for k=1:floor(y3_len/8)
   y3_down(:,k) = y3(:,8*k);
end
figure(4)
for k=1:7
   subplot(3,3,k)
```

```
   %f_y3=fftshift(20*log10(abs(fft(x3(k,:),8192))));
   f_y3=fftshift(20*log10(abs(fft(y3_down(k,:),8192))));
   plot((-0.5:1/8192:0.5-1/8192),fliplr(f_y3),'b','linewidth',2)
   grid on
   axis([-0.5 0.5 -120 60])
   title('Frequency response of individual channel for FIR low pass filter',k)
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
end
figure(5)
for k=1:7
   subplot(3,3,k)
   plot((0:floor(y3_len/8)-1),abs(y3_down(k,:)),'b','linewidth',2)
   grid on
   axis([-1 floor(y3_len/8) -1 1])
   title('Time response of individual channel for FIR low pass filter ', k)
   xlabel('Time index')
   ylabel('Amplitude')
end
%%Option 2
%8-path polyphase FIR filter
h2 = remez(taps-1,[0, 3, 7, 40]/40, {'myfrf', [1 1 0 0]}, [1 10]);
% Use Nyquist filter
%h2=sinc(-2 + 1/16:1/16:2).*kaiser(64,6)'/sum(kaiser(64,6));
%taps = 48;
%h=sinc(-2 + 1/(taps/4):1/(taps/4):2).*kaiser(taps,6)'/(sum(kaiser(taps,8))*1);
f_h2=fftshift(20*log10(abs(fft(h2,1024))));
%8-path polyphase FIR filter, 8 to 1 down sampling
%size_h2 = size(h2);
%taps_h2 = size_h2(2);
h2_8=reshape(h2*8,8,taps/8);
x_impulse = [zeros(1,3000) 1 zeros(1,3000)];
x2_prev = x2;
x2 = x_impulse;
x2_size = size(x2);
x2_len = x2_size(2);
fir_poly_impulse=zeros(8, x2_len);
reg = zeros(8,taps-7);
v0 = zeros(8,1);
v1 = zeros(8,x2_len);
h2_8_insert_zeros = zeros(8,taps-7);
for k=1:8
   h2_8_insert_zeros(:,8*k-7) = h2_8(:,k);
end
for n=1:1:x2_len
  reg =[v0 reg(:,1:(taps - 8))];
  v0 = [x2(n) v0(1:7,1)']';
  for k=1:8
   v1(k,n) = reg(k,:)*h2_8_insert_zeros(k,:)';
```

```
  end
  fir_poly_impulse(:,n) = ifft(v1(:,n),8)';
end
figure (6)
subplot(2,1,1)
plot(0:6000,fir_poly_impulse(1,:),'b','linewidth',2)
axis([-2 6002 -0.1 0.2])
grid on
title('Impulse Response, Prototype 64 taps Low Pass Filter')
xlabel('Time Index')
ylabel('Amplitude')
subplot(2,1,2)
plot((-0.5:1/8192:0.5-1/8192),fftshift(20*log10(abs(fft(fir_poly_impulse(1,:),8
192)))),'b','linewidth',2)
hold on
plot([-3/80 -3/80 3/80 3/80],[-120 0 0 -120],'r','linewidth',2)
plot([-0.5 -7/80 -7/80],[-60 -60 -20],'r','linewidth',2)
plot([ 7/80 7/80 0.5],[-20 -60 -60],'r','linewidth',2)
hold off
grid on
axis([-0.5 0.5 -120 10])
title('Frequency response of the Prototype 64 taps low pass Remez FIR Filter')
xlabel('Frequency')
ylabel('Log-Mag (dB)')
figure(18)
hold on
for k=1:8
   %subplot(3,3,k)
   %f_y4_clean=fftshift(20*log10(abs(fft(y4_clean(1,:),1024))));
   f_y4=fftshift(20*log10(abs(fft(fir_poly_impulse(k,:),8192))));
   plot((-0.5:1/8192:0.5-1/8192),f_y4,'b','linewidth',2)
   grid on
   axis([-0.5 0.5 -120 40])
   title('Frequency response of low pass polyphase FIR channel filter',k)
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
end
hold off
%clean up filter ( fs=10, del_f =4, N=8)
%h2_clean_up = remez(32-1,[0, 3, 4, 5]/5, {'myfrf', [1 1 0 0]}, [1 10]);
%f_h_clean_up=fftshift(20*log10(abs(fft(h2_clean_up,1024))));
reg=zeros(8,taps/8);
v0=zeros(8,1);
%FIR polyphase filter impulse response
x_impulse = [zeros(1,3000) 1 zeros(1,3000)];
x2 = x_impulse;
x2_size = size(x2);
x2_len = x2_size(2);
fir_poly_impulse_down=zeros(8, floor(x2_len/8));
```

```
v1=zeros(8,floor(x2_len/8));
m=1;
for n=1:8:8*floor(x2_len/8)
    v0=fliplr(x2(n:n+7)).';
    reg=[v0 reg(:,1:(taps/8 - 1))];
        for k=1:8
          v1(k,m)=reg(k,:)*h2_8(k,:)';
        end
       %ifft
    %y0(m)=sum(v1(:,m));
    fir_poly_impulse_down(:,m) = ifft(v1(:,m),8)';
    m=m+1;
end
%filter QAM signal
x2 = x2_prev;
x2_size = size(x2);
x2_len = x2_size(2);
y4_fir_poly=zeros(8, floor(x2_len/8));
v1=zeros(8,floor(x2_len/8));
m=1;
for n=1:8:8*floor(x2_len/8)
    v0=fliplr(x2(n:n+7)).';
    reg=[v0 reg(:,1:(taps/8 - 1))];
        for k=1:8
          v1(k,m)=reg(k,:)*h2_8(k,:)';
        end
       %ifft
    %y0(m)=sum(v1(:,m));
    y4_fir_poly(:,m) = ifft(v1(:,m),8)';
    m=m+1;
end
%h2_clean_up_size = size(h2_clean_up);
%h2_clean_up_len = h2_clean_up_size(2);
%y4_clean = zeros(8,h2_clean_up_len+m-2);
%for k=1:8
%  y4_clean(k,:) = conv(y4(k,:),h2_clean_up);
%end
%y1 = conv(h2,x2);
figure(7)
for k=1:8
   subplot(3,3,k)
   %f_y4_clean=fftshift(20*log10(abs(fft(y4_clean(1,:),1024))));
   f_y4_fir_poly=fftshift(20*log10(abs(fft(y4_fir_poly(k,:),1024))));
   plot((-0.5:1/1024:0.5-1/1024),f_y4_fir_poly,'b','linewidth',2)
   %plot((-0.5:1/1024:0.5-1/1024),f_h_clean_up,'b','linewidth',2)
   grid on
   axis([-0.5 0.5 -120 40])
   title('Frequency response of individual channel for polyphase FIR filter',k)
   xlabel('Frequency')
```

```
   ylabel('Log-Mag (dB)')
end
%Option 3, IIR low pass filter with linear phase
%%Digitally down convert each channel to baseband
x3=zeros(7,16*N);
for k=1:7
   x3(k,:) = x2.*exp(-j*2*pi*(0:16*N-1)*ff(k));
end
% Denominator Polynomials, path-0
% branch 0 - 24 Delays,     Z^(-24)
% Denominator Polynomials, path-1
%   (a0 Z^2  +  a1 Z^1  +  a2)
%a_path = zeros(paths,2,2*paths+1);
paths = 8;
a_path = zeros(8,2,17);
a_path(2,1,paths+1) = 0.3000581202376679;
a_path(2,2,paths+1) = -0.1900402095694197;
a_path(2,2,2*paths+1) = 0.02633139911243305;
% Denominator Polynomials, path-2
a_path(3,1,paths+1) = 0.4222035911739405;
a_path(3,2,paths+1) = -0.1989238080170467;
a_path(3,2,2*paths+1) = 0.03024784889553085;
% Denominator Polynomials, path-3
a_path(4,1,paths+1) = 0.5254281202136269;
a_path(4,2,paths+1) = -0.1852117598974453;
a_path(4,2,2*paths+1) =  0.02859219142615069;
% Denominator Polynomials, path-4
a_path(5,1,paths+1) = 0.6212350941176464;
a_path(5,2,paths+1) = -0.1599165480534598;
a_path(5,2,2*paths+1) = 0.02421203546513172;
% Denominator Polynomials, path-5
a_path(6,1,paths+1) = 0.714275602892003;
a_path(6,2,paths+1) = -0.1271271167937832;
a_path(6,2,2*paths+1) = 0.01846501616252262;
% Denominator Polynomials, path-6
a_path(7,1,paths+1) = 0.807212346911608;
a_path(7,2,paths+1) = -0.08885934163493368;
a_path(7,2,2*paths+1) = 0.01216766845066991;
% Denominator Polynomials, path-7
a_path(8,1,paths+1) = 0.9019330309014866;
a_path(8,2,paths+1) = -0.0462528555078691;
a_path(8,2,2*paths+1) = 0.005875680228912086;
reg = zeros(paths,3,3*paths);
%reg = zeros(paths,4,5*paths);
dly=zeros(1,paths-1);
x0=[zeros(1,3000) 1 zeros(1,3000)];
%x0 = x2;
sm = zeros(paths,2);
x0_size = size(x0);
```

```
x0_len = x0_size(2);
len_y = floor(x0_len/paths);
y=zeros(1,paths*len_y);
y0=zeros(1,paths*len_y);
for n=1:1:paths*len_y
   for k=2:1:paths
      sm(k,1)=(dly(k-1)-reg(k,2,1*paths))*a_path(k,1,(paths +
1))+reg(k,1,1*paths);
      sm(k,2)=(sm(k,1)-reg(k,3,2*paths))*a_path(k,2,(2*paths +
1))+(reg(k,2,1*paths)-reg(k,3,1*paths))*a_path(k,2,(paths +
1))+reg(k,2,2*paths);
   end
sm(1,2) = reg(1,1,3*paths);

y(n) = 1/8 * (sm(1,2) + sm(2,2) + sm(3,2) + sm(4,2) + sm(5,2) + sm(6,2) +
sm(7,2) + sm(8,2));
for k=2:paths
  reg(k,1,2:3*paths)= reg(k,1,1:3*paths-1);
  reg(k,2,2:3*paths)= reg(k,2,1:3*paths-1);
  reg(k,3,2:3*paths)= reg(k,3,1:3*paths-1);
   reg(k,1,1)= dly(k-1);
  reg(k,2,1)=sm(k,1);
  reg(k,3,1)=sm(k,2);

 end
dly= [x0(n) dly(1:paths-2)];
reg(1,1,2:3*paths)= reg(1,1,1:3*paths-1);
reg(1,1,1)= x0(n);
end
out_iir_linear_impulse = y;
out_iir_linear_impulse_down = zeros(1,len_y);
for k=1:len_y
   out_iir_linear_impulse_down(k) = out_iir_linear_impulse(8*k);
end
figure(8)
subplot(2,1,1)
plot((0:paths*len_y-1),out_iir_linear_impulse(1,:),'b','linewidth',2)
grid on
axis([-1 paths*len_y+1 -1 1])
title('Impulse response of IIR linear phase filter')
xlabel('Time index')
ylabel('Amplitude')
subplot(2,1,2)
f_out_iir_linear_impulse=fftshift(20*log10(abs(fft(out_iir_linear_impulse(1,:),
8192))));
plot((-0.5:1/8192:0.5-1/8192),f_out_iir_linear_impulse,'b','linewidth',2)
hold on
plot([-3/80 -3/80 3/80 3/80],[-120 0 0 -120],'r','linewidth',2)
plot([-0.5 -7/80 -7/80],[-60 -60 -20],'r','linewidth',2)
```

```
plot([ 7/80 7/80 0.5],[-20 -60 -60],'r','linewidth',2)
hold off
grid on
axis([-0.5 0.5 -120 40])
title('Frequency response of IIR linear phase filter')
xlabel('Frequency')
ylabel('Log-Mag (dB)')
%% Filter the 7 QAM signal
x0 = x3(1,:);
x0_size = size(x0);
len_y = x0_size(2);
paths = 8;
out_iir_linear_qam_down = zeros(7,floor(len_y/8));
y = zeros(7,8*floor(len_y/8));
for p=1:7
   x0 = x3(p,:);
   reg = zeros(paths,3,3*paths);
   dly=zeros(1,paths-1);
   sm = zeros(paths,2);
   for n=1:1:len_y
     for k=2:1:paths
        sm(k,1)=(dly(k-1)-reg(k,2,1*paths))*a_path(k,1,(paths +
1))+reg(k,1,1*paths);
        sm(k,2)=(sm(k,1)-reg(k,3,2*paths))*a_path(k,2,(2*paths +
1))+(reg(k,2,1*paths)-reg(k,3,1*paths))*a_path(k,2,(paths +
1))+reg(k,2,2*paths);
     end
   sm(1,2) = reg(1,1,3*paths);

   y(p,n) = 1/8 * (sm(1,2) + sm(2,2) + sm(3,2) + sm(4,2) + sm(5,2) + sm(6,2) +
sm(7,2) + sm(8,2));
   for k=2:paths
       reg(k,1,2:3*paths)= reg(k,1,1:3*paths-1);
       reg(k,2,2:3*paths)= reg(k,2,1:3*paths-1);
       reg(k,3,2:3*paths)= reg(k,3,1:3*paths-1);
       reg(k,1,1)= dly(k-1);
       reg(k,2,1)=sm(k,1);
       reg(k,3,1)=sm(k,2);
   end
   dly= [x0(n) dly(1:paths-2)];
   reg(1,1,2:3*paths)= reg(1,1,1:3*paths-1);
   reg(1,1,1)= x0(n);
   end
end
out_iir_linear_qam = y;
for k=1:floor(len_y/8)
   out_iir_linear_qam_down(:,k) = out_iir_linear_qam(:,8*k);
end
figure(9)
```

```
for k=1:7
   subplot(3,3,k)

f_out_iir_linear_qam_down=fftshift(20*log10(abs(fft(out_iir_linear_qam_down(k,:
),1024))));
   plot((-0.5:1/1024:0.5-1/1024),f_out_iir_linear_qam_down,'b','linewidth',2)
   grid on
   axis([-0.5 0.5 -120 40])
   title('Frequency response of individual channel for IIR linear phase filter
', k)
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
end
figure(10)
for k=1:7
   subplot(3,3,k)

plot((0:floor(len_y/8)-1),(abs(out_iir_linear_qam_down(k,:))),'b','linewidth',2
)
   grid on
   axis([-1 floor(len_y/8)+1 -2 2])
   title('Time response of individual channel using IIR linear phase filter ',
k)
   xlabel('Time index')
   ylabel('Amplitude')
end
%Option 4: 8 path IIR linear polyphase filter (8 to 1 downsample)
% Denominator Polynomials, path-0
% branch 0 - 24 Delays,     Z^(-24)
% Denominator Polynomials, path-1
%   (a0 Z^2  +  a1 Z^1  +  a2)
%a_path = zeros(paths,2,2*paths+1);
paths = 8;
a_path = zeros(8,2,17);
a_path(2,1,paths+1) = 0.3000581202376679;
a_path(2,2,paths+1) = -0.1900402095694197;
a_path(2,2,2*paths+1) = 0.02633139911243305;
% Denominator Polynomials, path-2
a_path(3,1,paths+1) = 0.4222035911739405;
a_path(3,2,paths+1) = -0.1989238080170467;
a_path(3,2,2*paths+1) = 0.03024784889553085;
% Denominator Polynomials, path-3
a_path(4,1,paths+1) = 0.5254281202136269;
a_path(4,2,paths+1) = -0.1852117598974453;
a_path(4,2,2*paths+1) =  0.02859219142615069;
% Denominator Polynomials, path-4
a_path(5,1,paths+1) = 0.6212350941176464;
a_path(5,2,paths+1) = -0.1599165480534598;
a_path(5,2,2*paths+1) = 0.02421203546513172;
```

```
% Denominator Polynomials, path-5
a_path(6,1,paths+1) = 0.714275602892003;
a_path(6,2,paths+1) = -0.1271271167937832;
a_path(6,2,2*paths+1) = 0.01846501616252262;
% Denominator Polynomials, path-6
a_path(7,1,paths+1) = 0.807212346911608;
a_path(7,2,paths+1) = -0.08885934163493368;
a_path(7,2,2*paths+1) = 0.01216766845066991;
% Denominator Polynomials, path-7
a_path(8,1,paths+1) = 0.9019330309014866;
a_path(8,2,paths+1) = -0.0462528555078691;
a_path(8,2,2*paths+1) = 0.005875680228912086;
reg = zeros(paths,3,3*paths);
%reg = zeros(paths,4,5*paths);
dly=zeros(1,paths-1);
x0=[zeros(1,3000) 1 zeros(1,3000)];
%x0 = x2;
sm = zeros(paths,2);
x0_size = size(x0);
x0_len = x0_size(2);
len_y = floor(x0_len/paths);
y=zeros(paths,paths*len_y);
y0=zeros(1,paths*len_y);
m = 1;
for n=1:1:paths*len_y
   for k=2:1:paths
      sm(k,1)=(dly(k-1)-reg(k,2,1*paths))*a_path(k,1,(paths +
1))+reg(k,1,1*paths);
      sm(k,2)=(sm(k,1)-reg(k,3,2*paths))*a_path(k,2,(2*paths +
1))+(reg(k,2,1*paths)-reg(k,3,1*paths))*a_path(k,2,(paths +
1))+reg(k,2,2*paths);
   end
sm(1,2) = reg(1,1,3*paths);

y(:,n) = ifft(sm(:,2),paths)';
%if n==8800
%    reg(1,1,1*paths)
%    reg(1,1,2*paths)
%    reg(1,1,3*paths)
%    %sm(:,2)
%    %y(:,n)
%end
for k=2:paths
  reg(k,1,2:3*paths)= reg(k,1,1:3*paths-1);
  reg(k,2,2:3*paths)= reg(k,2,1:3*paths-1);
  reg(k,3,2:3*paths)= reg(k,3,1:3*paths-1);
   reg(k,1,1)= dly(k-1);
  reg(k,2,1)=sm(k,1);
  reg(k,3,1)=sm(k,2);
```

```
 end
dly= [x0(n) dly(1:paths-2)];
reg(1,1,2:3*paths)= reg(1,1,1:3*paths-1);
reg(1,1,1)= x0(n);
end
figure(11)
%for k=1:paths
   %subplot(3,3,k)
   %f_y4_clean=fftshift(20*log10(abs(fft(y4_clean(1,:),1024))));
   f_y4=fftshift(20*log10(abs(fft(y(1,:),8192))));
   plot((-0.5:1/8192:0.5-1/8192),f_y4,'b','linewidth',2)
   grid on
   axis([-0.5 0.5 -120 40])
   hold on
   plot([-3/80 -3/80 3/80 3/80],[-120 0 0 -120],'r','linewidth',2)
   plot([-0.5 -7/80 -7/80],[-60 -60 -20],'r','linewidth',2)
   plot([ 7/80 7/80 0.5],[-20 -60 -60],'r','linewidth',2)
   hold off
   title('Frequency response of prototype low pass polyphase IIR filter',k)
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
%end
figure(17)
hold on
for k=1:paths
   %subplot(3,3,k)
   %f_y4_clean=fftshift(20*log10(abs(fft(y4_clean(1,:),1024))));
   f_y4=fftshift(20*log10(abs(fft(y(k,:),8192))));
   plot((-0.5:1/8192:0.5-1/8192),f_y4,'b','linewidth',2)
   grid on
   axis([-0.5 0.5 -120 40])
   title('Frequency response of low pass polyphase IIR channel filter',k)
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
end
hold off
figure(12)
%for k=1:paths
%   subplot(3,3,k)
   plot((0:paths*len_y - 1),abs(y(1,:)),'b','linewidth',2)
   grid on
   axis([-1 paths*len_y - 1 -0.5 0.5])
   title('Impulse response of prototype low pass polyphase IIR filter',k)
   xlabel('Time index')
   ylabel('Amplitude')
%end
y_down_ref = y(1,2:paths:paths*len_y)*8;
%Downsampling
```

```
reg = zeros(paths,3,3*paths);
%reg = zeros(paths,4,5*paths);
dly=zeros(1,paths-1);
%impulse response
x0=[zeros(1,3000) 1 zeros(1,3000)];
sm = zeros(paths,2);
x0_size = size(x0);
x0_len = x0_size(2);
len_y = floor(x0_len/paths);
y_iir_poly_impulse_down = zeros(paths,len_y);
m = 1;
for n=paths:paths:paths*len_y
   for k=2:1:paths
      sm(k,1)=(dly(k-1)-reg(k,2,1*paths))*a_path(k,1,(paths +
1))+reg(k,1,1*paths);
      sm(k,2)=(sm(k,1)-reg(k,3,2*paths))*a_path(k,2,(2*paths +
1))+(reg(k,2,1*paths)-reg(k,3,1*paths))*a_path(k,2,(paths +
1))+reg(k,2,2*paths);
   end
sm(1,2) = reg(1,1,3*paths);
y_iir_poly_impulse_down(:,m) = ifft(sm(:,2),paths)';
for k=2:paths
  reg(k,1,paths + 1:3*paths)= reg(k,1,1:2*paths);
  reg(k,2,paths + 1:3*paths)= reg(k,2,1:2*paths);
  reg(k,3,paths + 1:3*paths)= reg(k,3,1:2*paths);
  reg(k,1,paths)= dly(k-1);
  reg(k,2,paths)=sm(k,1);
  reg(k,3,paths)=sm(k,2);

 end
reg(1,1,paths+1:3*paths)= reg(1,1,1:2*paths);
dly= fliplr(x0(n-paths+1:n-1));
if n>paths
reg(1,1,paths)= x0(n-paths);
end
m = m + 1;
end
%filter QAM signal
reg = zeros(paths,3,3*paths);
x0 = x2;
sm = zeros(paths,2);
x0_size = size(x0);
x0_len = x0_size(2);
len_y = floor(x0_len/paths);
y_iir_poly=zeros(paths,len_y);
m = 1;
for n=paths:paths:paths*len_y
   for k=2:1:paths
```

```
      sm(k,1)=(dly(k-1)-reg(k,2,1*paths))*a_path(k,1,(paths +
1))+reg(k,1,1*paths);
      sm(k,2)=(sm(k,1)-reg(k,3,2*paths))*a_path(k,2,(2*paths +
1))+(reg(k,2,1*paths)-reg(k,3,1*paths))*a_path(k,2,(paths +
1))+reg(k,2,2*paths);
   end
sm(1,2) = reg(1,1,3*paths);
y_iir_poly(:,m) = ifft(sm(:,2),paths)';
 %if m==1101
 %   reg(1,1,1*paths)
 %   reg(1,1,2*paths)
 %   reg(1,1,3*paths)
 %   %sm(:,2)
 %   %y_down(:,m)/8
 %end
for k=2:paths
  reg(k,1,paths + 1:3*paths)= reg(k,1,1:2*paths);
  reg(k,2,paths + 1:3*paths)= reg(k,2,1:2*paths);
  reg(k,3,paths + 1:3*paths)= reg(k,3,1:2*paths);
  reg(k,1,paths)= dly(k-1);
  reg(k,2,paths)=sm(k,1);
  reg(k,3,paths)=sm(k,2);

 end
reg(1,1,paths+1:3*paths)= reg(1,1,1:2*paths);
%dly= [x0(n) dly(1:paths-2)];
%dly= fliplr(x0(n:n+paths-2));
dly= fliplr(x0(n-paths+1:n-1));
if n>paths
reg(1,1,paths)= x0(n-paths);
end
%reg(1,1,paths)= x0(n+paths-1);
m = m + 1;
end
%figure(5)
%subplot(2,1,1)
%plot(1:len_y,y_down(1,1:len_y),'b','linewidth',2)
%grid on
%axis([0 len_y -0.2 0.6])
%title('Impulse Rresponse')
%xlabel('Time Index')
%ylabel('Amplitude')
%text(20, 0.2,'9-th Order Filter, 4 Multiplies')
figure(13)
for k=1:paths
   subplot(3,3,k)
   %f_y4_clean=fftshift(20*log10(abs(fft(y4_clean(1,:),1024))));
   f_y_iir_poly=fftshift(20*log10(abs(fft(y_iir_poly(k,:),1024))));
   plot((-0.5:1/1024:0.5-1/1024),f_y_iir_poly,'b','linewidth',2)
```

```
   grid on
   axis([-0.5 0.5 -120 40])
   title('Frequency response of output at channel for polyphase IIR filter',k)
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
end
figure(14)
%for k=1:8
   %subplot(3,3,k)
   %f_y4_clean=fftshift(20*log10(abs(fft(y4_clean(1,:),1024))));
   f_y3_fir=fftshift(20*log10(abs(fft(y3_down(4,:),1024))));
   f_y4_fir_poly=fftshift(20*log10(abs(fft(y4_fir_poly(1,:),1024))));

f_out_iir_linear_qam_down=fftshift(20*log10(abs(fft(out_iir_linear_qam_down(4,:
),1024))));
   f_y_iir_poly=fftshift(20*log10(abs(fft(y_iir_poly(1,:),1024))));
   subplot(2,2,1)
   plot((-0.5:1/1024:0.5-1/1024),fliplr(f_y3_fir),'b','linewidth',2)
   title('Frequency response of output at baseband with FIR filter',k)
   grid on
   axis([-0.5 0.5 -120 40])
   subplot(2,2,2)
   plot((-0.5:1/1024:0.5-1/1024),f_y4_fir_poly,'b','linewidth',2)
   title('Frequency response of output at baseband with polyphase FIR
filter',k)
   axis([-0.5 0.5 -120 40])
   grid on
   subplot(2,2,3)

plot((-0.5:1/1024:0.5-1/1024),fliplr(f_out_iir_linear_qam_down),'b','linewidth'
,2)
   title('Frequency response of output at baseband with IIR filter',k)
   grid on
   axis([-0.5 0.5 -120 40])
   subplot(2,2,4)
   plot((-0.5:1/1024:0.5-1/1024),f_y_iir_poly,'b','linewidth',2)
   title('Frequency response of output at baseband with polyphase IIR
filter',k)
   %plot((-0.5:1/1024:0.5-1/1024),f_y4_clean,'b','linewidth',2)
   %plot((-0.5:1/1024:0.5-1/1024),f_h_clean_up,'b','linewidth',2)
   grid on
   axis([-0.5 0.5 -120 40])
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
%end
figure(15)
%for k=1:8
   %subplot(3,3,k)
   subplot(2,2,1)
```

```
   plot((0:749),8*y_fir_impulse_down,'b','linewidth',2)
   title('Impulse response of FIR filter after downsampling')
   grid on
   axis([-2 752 -2 2])
   subplot(2,2,2)
   plot((0:749),8*fir_poly_impulse_down(1,:),'b','linewidth',2)
   title('Impulse response of polyphase FIR filter after downsampling')
   axis([-2 752 -2 2])
   grid on
   subplot(2,2,3)
   plot((0:749),8*out_iir_linear_impulse_down,'b','linewidth',2)
   title('Impulse response of IIR filter after downsampling')
   grid on
   axis([-2 752 -2 2])
   subplot(2,2,4)
   plot((0:749),8*y_iir_poly_impulse_down(1,:),'b','linewidth',2)
   title('Impulse response of polyphase IIR filter after downsampling')
   %plot((-0.5:1/1024:0.5-1/1024),f_y4_clean,'b','linewidth',2)
   %plot((-0.5:1/1024:0.5-1/1024),f_h_clean_up,'b','linewidth',2)
   grid on
   axis([-2 752 -2 2])
   xlabel('Frequency')
   ylabel('Log-Mag (dB)')
%end
figure(16)
%for k=1:8
   %subplot(3,3,k)
   %f_y4_clean=fftshift(20*log10(abs(fft(y4_clean(1,:),1024))));
   f_y3_fir=fftshift(20*log10(8*abs(fft(y_fir_impulse_down,1024))));

f_y4_fir_poly=fftshift(20*log10(8*abs(fft(fir_poly_impulse_down(1,:),1024))));

f_out_iir_linear_impulse=fftshift(20*log10(8*abs(fft(out_iir_linear_impulse_dow
n,1024))));

f_y_iir_poly=fftshift(20*log10(8*abs(fft(y_iir_poly_impulse_down(1,:),1024))));
   subplot(2,2,1)
   plot((-0.5:1/1024:0.5-1/1024),f_y3_fir,'b','linewidth',2)
   title('Frequency response of FIR filter after downsampling')
   grid on
   axis([-0.5 0.5 -120 40])
   subplot(2,2,2)
   plot((-0.5:1/1024:0.5-1/1024),f_y4_fir_poly,'b','linewidth',2)
   title('Frequency response of polyphase FIR filter after downsampling')
   axis([-0.5 0.5 -120 40])
   grid on
   subplot(2,2,3)
   plot((-0.5:1/1024:0.5-1/1024),f_out_iir_linear_impulse,'b','linewidth',2)
   title('Frequency response of IIR filter after downsampling')
```

```
    grid on
    axis([-0.5 0.5 -120 40])
    subplot(2,2,4)
    plot((-0.5:1/1024:0.5-1/1024),f_y_iir_poly,'b','linewidth',2)
    title('Frequency response of polyphase IIR filter after downsampling')
    %plot((-0.5:1/1024:0.5-1/1024),f_y4_clean,'b','linewidth',2)
    %plot((-0.5:1/1024:0.5-1/1024),f_h_clean_up,'b','linewidth',2)
    grid on
    axis([-0.5 0.5 -120 40])
    xlabel('Frequency')
    ylabel('Log-Mag (dB)')
%end
```