## 1. <u>FIR – FILTRY.</u>

$$y(n)=\sum_{m=0}^{M} b_m x(n-m)=\sum_{m=0}^{M} h_m x(n-m)$$

$$\begin{bmatrix} b=[b_0, b_1, \ldots] \\ a=1 \end{bmatrix}$$

$$y = \mathit{filter}(b, a, x)$$

h – odpowiedź impulsowa filtru

$$H(z)=\frac{Y(z)}{X(z)}=\frac{B(z)}{A(z)}=\sum_{m=0}^{M} h_m z^{-m}$$

SPRAWDZENIE

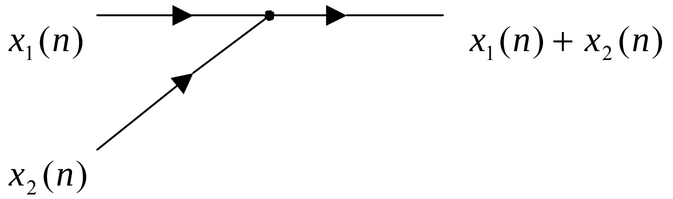
$$Y(z)=\left(\sum_{m=0}^{M} h_m z^{-m}\right)X(z)$$

STĄD

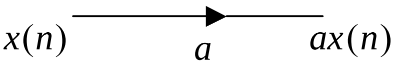
$$y(n)=\sum_{m=0}^{M} h_m x(n-m)$$

**GRAFICZNE PRZEDSTAWIENIE FILTRÓW CYFROWYCH**

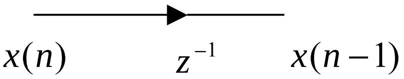
Sumator •



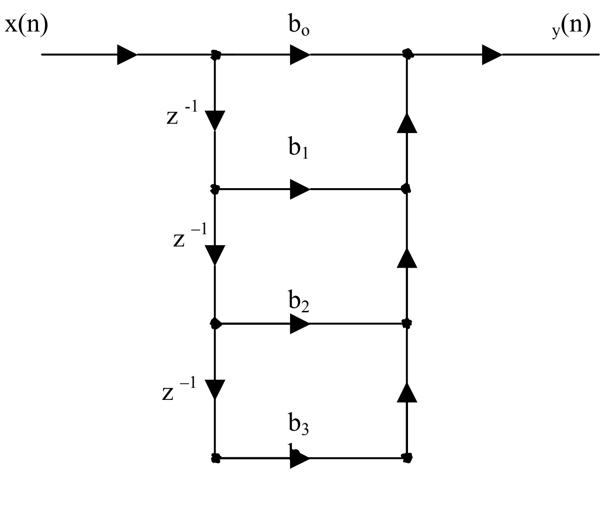

Mnożnik (wzmocnienie)




Opóźnienie




POSTAĆ BEZPOŚREDNIA FILTRU FIR

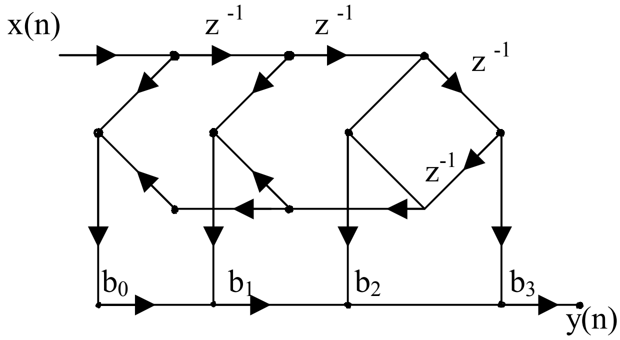


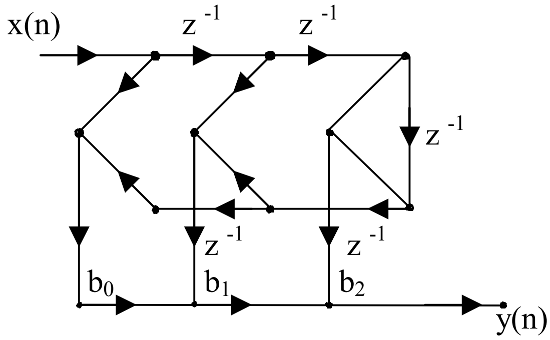

$Y(n)=b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + \ldots$
$y(n)=b_0x(n) + b_1x(n-1) + b_2x(n-2) + \ldots$

RZĄD FILTRU = LICZBA WSPÓŁCZYNNIKÓW = M+1.

Jeśli filtr FIR ma mieć liniową fazę to współczynniki filtru są symetryczne lub antysymetryczne.

Filtr FIR o liniowej fazie



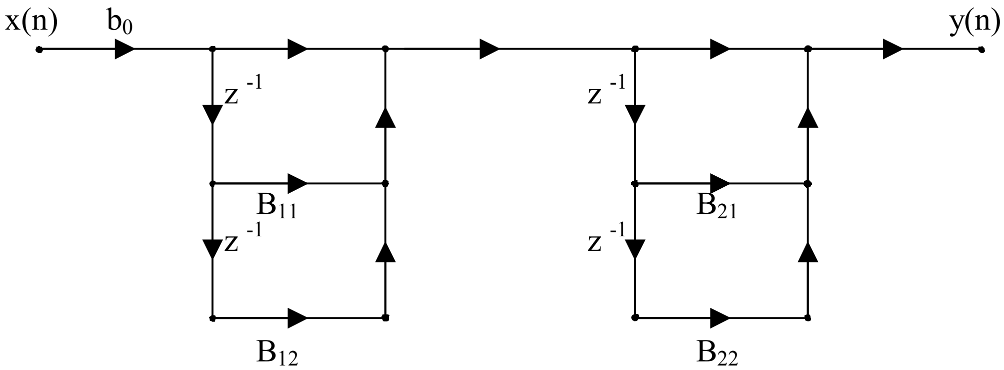




$b_0\ b_1\ b_2\ b_3\ b_2\ b_1\ b_0$ $\qquad$ $b_0\ b_1\ b_2\ b_2\ b_1\ b_0$

KASKADOWA REALIZACJA FILTRU FIR

$$H(z)=b_0+b_1z^{-1}+b_2z^{-2}+\ldots+b_{M-1}z^{-M+1}=b_0(1+\frac{b_1}{b_0}z^{-1}+\frac{b_2}{b_0}z^{-2}+\ldots+\frac{b_{M-1}}{b_0}z^{-M+1})=$$

$$b_0\prod_{k=1}^{M}(1+B_{k1}z^{-1}+B_{k2}z^{-2})$$




%Procedura zmiany formy bezpośredniej na kaskadową (BI-KWADRATOWĄ)

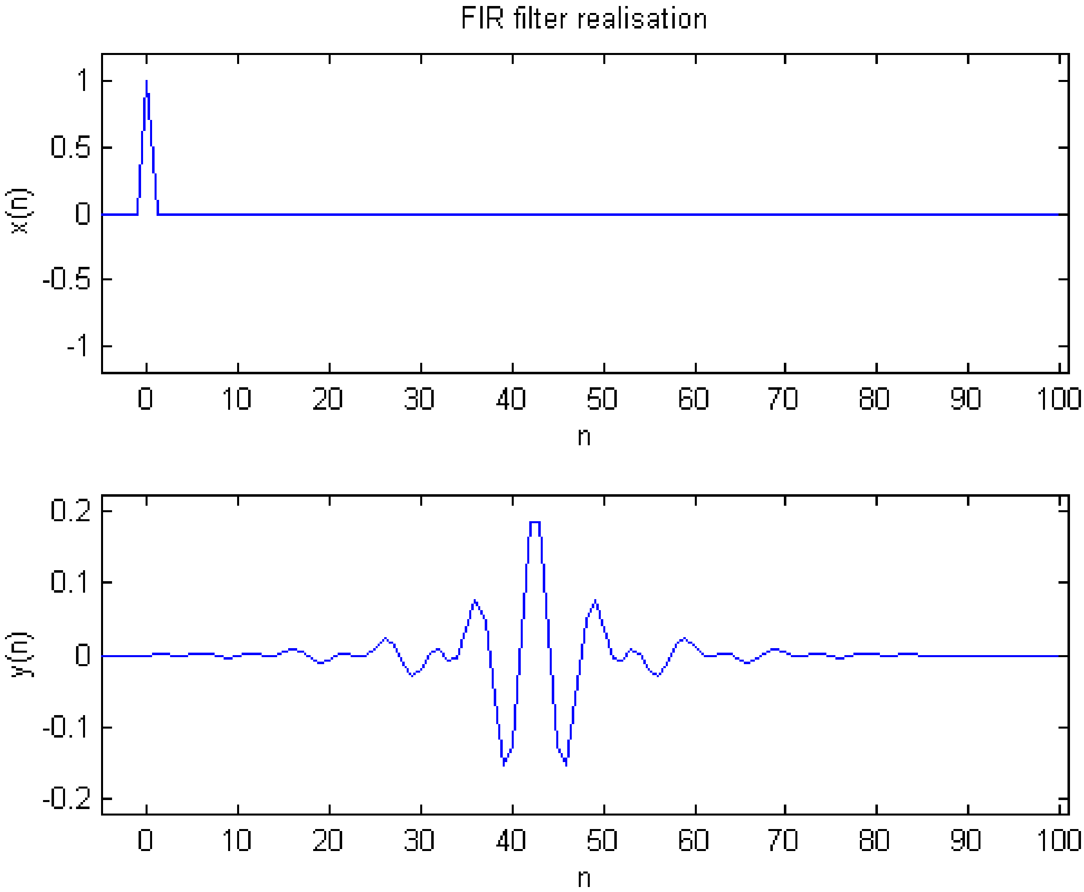
b=[ $b_0$, ...];

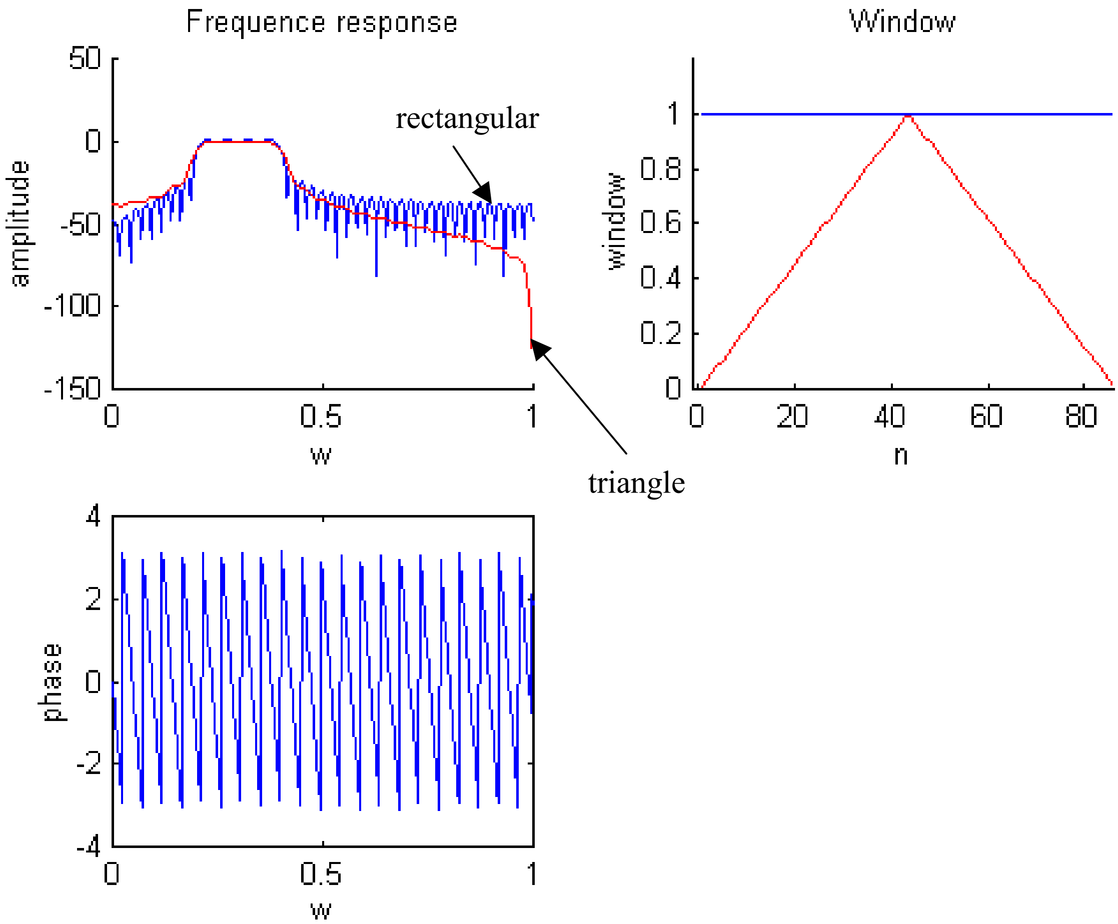
a=[1];

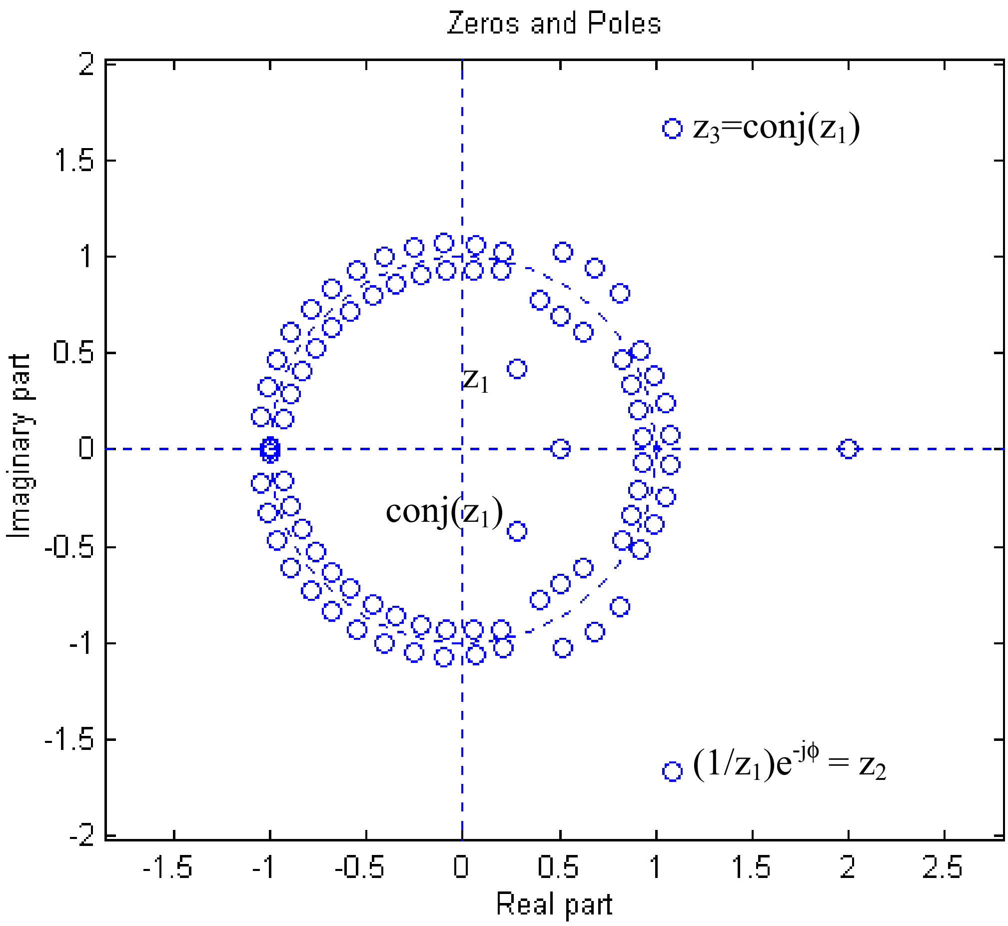
[ $b_0$, B, A]=dir2cas(b,a);

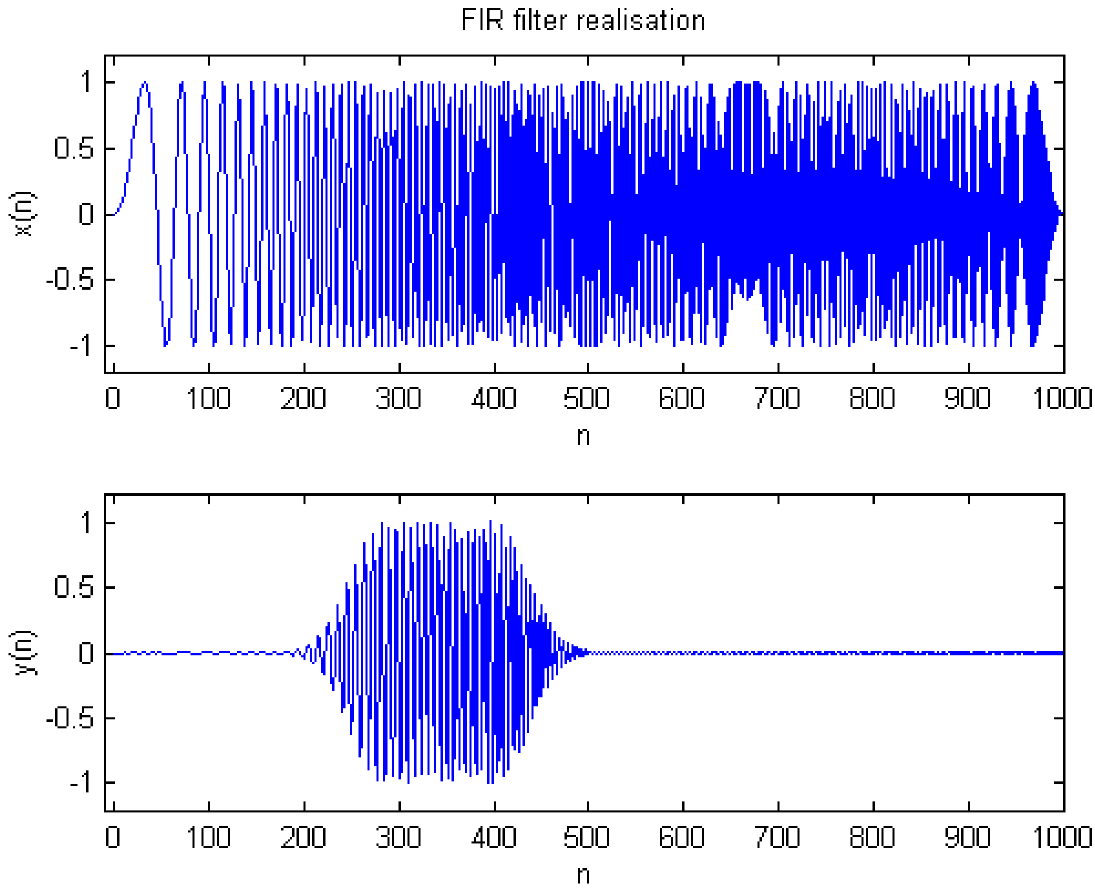
```
% fir_try.m                        % Badanie filtrów typu FIR
close;
a=1;
%b=[ones(1,10)]/10;                % average sinx/x low pass filter
%b=[0,0,0,0,1,-1,0,0,0,0];         % differentiator
%b=[0.5,0,0,0,0,0,0,0,0,-0.5];     % comb filter

% FIR Filter Design: coeficient calculation -----------------------
close;
Nord=85;
subplot(2,2,1);
hold on;
b=fir1(Nord,[0.2,0.4],boxcar(Nord+1))  % rectangular window method
[H,w] =freqz(b,a,500);
plot(w/pi,20*log10(abs(H)))
%b=fir1(Nord,[0.2,0.4],hamming(Nord+1))  % Hamming window
%b=fir1(Nord,[0.2,0.4],hanning(Nord+1))  % Hanning window
%b=fir1(Nord,[0.2,0.4],blackman(Nord+1)) % Blackman window
b=fir1(Nord,[0.2,0.4],triang(Nord+1))    % rectangular window
%b=remez(Nord,[ 0 0.18 0.2 0.38 0.4 1],[0 0.001 1 1 0.001 0])  %Parks-McClellan: equiripples
[H,w] =freqz(b,a,500);
plot(w/pi,20*log10(abs(H)),'r')
hold off;
xlabel('w');ylabel('amplitude');
title('Frequence response');
subplot(2,2,3);plot(w/pi,angle(H))
axis([0,1,-4,4]);
xlabel('w');ylabel('phase');
subplot(2,2,2);
hold on;
plot(boxcar(Nord+1))
plot(triang(Nord+1),'r')
axis([-1,Nord+1,0,1.2]);
xlabel('n');ylabel('window');
hold off;
title('Window');
pause;

% FIR filter impulse response
close;
n=[-10:200];
x=[(n-0)==0];
y=filter(b,a,x);
stem(n,y,'.')
axis([-10,length(n),-abs(max(y)*1.1),abs(max(y)*1.1)]);
xlabel('n');ylabel('h(n)');
title('FIR Filter Impulse Response');
pause;

%FIR Filter and chirp input signal
close;
n=[0:1000];
x=sin(2*pi/4000.*n.*n);
s=filter(b,a,x);
subplot(2,1,1);plot(n,x)
xlabel('n');ylabel('input(n)');
axis([0,1000,-1.2,1.2])
title('Chirp Response');
subplot(2,1,2);plot(n,s)
axis([0,1000,-1.2,1.2])
xlabel('n');ylabel('output(n)');
pause;

%z-plane: zeros and poles
```

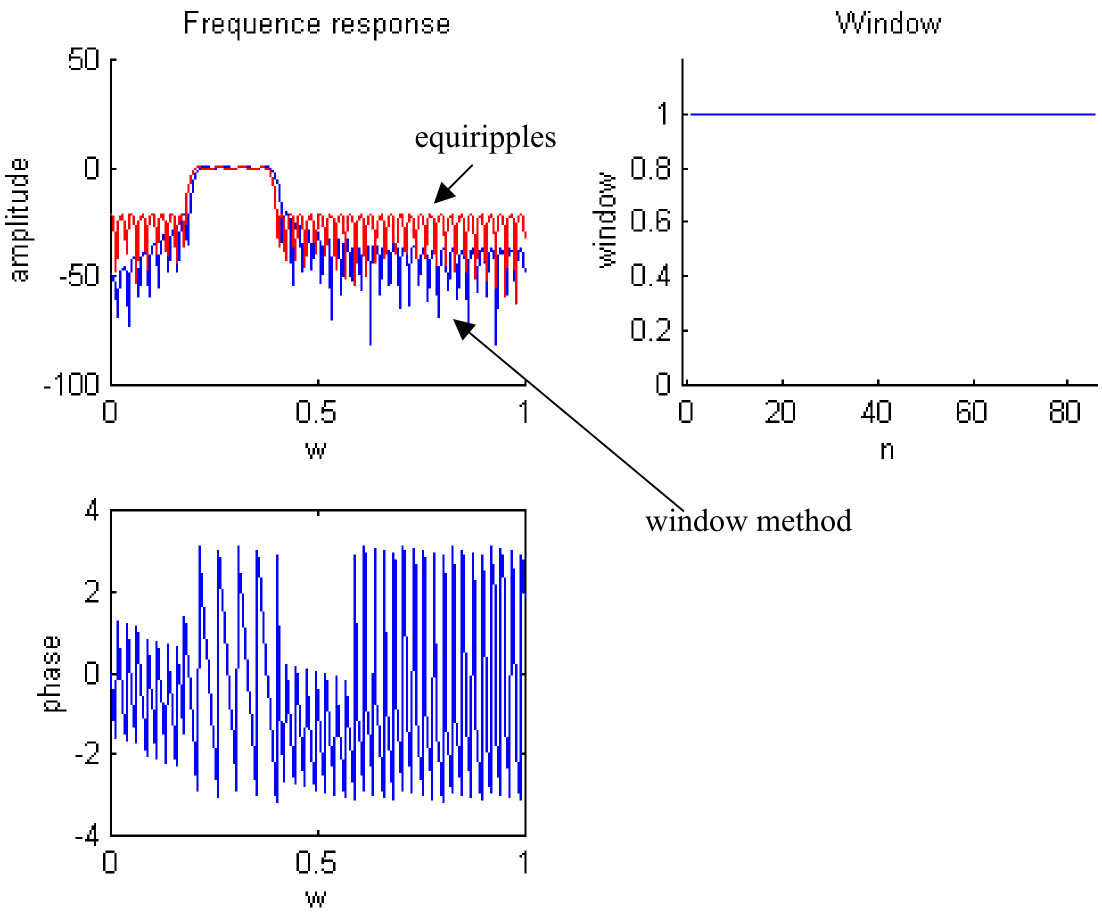
```
close;
zplane(b,a);
title('Zeros and Poles');
pause;

%-------------------------------------------------------------------
% FIR algorithm realisation
close;
n=[0:1000];
%x=randn(1,1001);         %szum
x=sin(2*pi/4000.*n.*n);  %chirp
%n=[-5:100]; x=[(n==0)];  %pulse
y=zeros(1,length(x));    % x - input ; y - output ; hist - input history
N=length(b);
hist=zeros(1,N+1);
coef=b;                  % coef - FIR filyrt coeficients
for j=1:length(x),
   hist(N+1)=x(j);
   out=0;
   for i=1:N,
      hist(i)=hist(i+1);
      out=out+hist(i+1)*coef(N-i+1);
   end;
   y(j)=out;
end;
subplot(2,1,1);plot(n,x);
axis([-5,length(n)+n(1),-abs(max(x)*1.2),abs(max(x)*1.2)])
xlabel('n');ylabel('x(n)');
title('FIR filter realisation');
subplot(2,1,2);plot(n,y);
axis([-5,length(n)+n(1),-abs(max(y)*1.2),abs(max(y)*1.2)])
xlabel('n');ylabel('y(n)');
pause;
sound(x,2000);
pause;
sound(y,2000);

%--------------------------------produce coeficient table
disp(b)
fid=fopen('coef.dat','w');
fprintf(fid,' #define N  %d\r\n',N);
fprintf(fid,' const  float h[N] = {\r\n');
j=1;
for i=1:N,
      j=j+1;
      fprintf(fid,'%12.6g, ',b(i));
      if j==6
         fprintf(fid,'\r\n');
         j=1;
      end;
   end;
end;
fprintf(fid,' };\r\n');
fclose(fid);
```


FIR filter realisation
x(n)
1
0.5
0
-0.5
-1
0 10 20 30 40 50 60 70 80 90 100
n
y(n)
0.2
0.1
0
-0.1
-0.2
0 10 20 30 40 50 60 70 80 90 100
n


Frequence response
amplitude
w
rectangular
triangle
Window
window
n
phase
w


Zeros and Poles
Imaginary part
Real part
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
$z_3$=conj($z_1$)
$z_1$
conj($z_1$)
$(1/z_1)e^{-j\phi} = z_2$


FIR filter realisation
x(n)
y(n)
n
n
1
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700
800
900
1000


Frequence response
amplitude
equiripples
w
Window
window
n
window method
phase
w