**FIR FILTERS**

1. Aim: To implement FIR LPF filter for the following specifications using DSP processor

Pass band edge frequency: 2000Hz

Pass band ripple: 2dB

Stop band edge frequency: 4000Hz

Stop band ripple: 40dB

Sampling frequency: 20000Hz

// Program to design FIR digital low pass filter

#include<stdio.h>

#include<math.h>

void main()

{

int N,i,k;

double fp=2000,fs=4000,Fs=20000;

double rp=2,rs=40,hd[25],Wn[25],h[45],pi=3.1415926,fc,TW;

fc=(fp+fs)/(2\*Fs);

TW=fabs((fs-fp))/Fs;

N=ceil(4/TW);

if(N%2==0)

N=N+1;

k=(N-1)/2;

for(i=0;i<k;i++)

{

Wn[i]=0.5\*(1-cos((2\*pi\*(double)i)/(N-1)));

hd[i]=(sin(2\*pi\*fc\*(double)(i-k)))/(pi\*(i-k));

h[i]=hd[i]\*Wn[i];

h[N-i-1]=h[i];

}

h[i]=fc\*(1-cos((2\*pi\*(double)i)/(N-1)));

}

1. AIM: To implement FIR HPF filter for the following specifications using DSP processor.

Pass band edge frequency: 4000Hz

Pass band ripple: 2dB

Stop band edge frequency: 2000Hz

Stop band ripple: 40dB

Sampling frequency: 20000Hz

// program to design FIR digital High pass filter

#include<stdio.h>

#include<math.h>

void main()

{

int N,i,k;

double fp=4000,fs=2000,Fs=20000;

double rp=2,rs=40,hd[25],Wn[25],h[45],pi=3.1415926,fc,TW;

fc=((fp+fs)/(2\*Fs));

TW=fabs((fs-fp))/Fs;

N=ceil(4/TW);

if(N%2==0)

N=N+1;

k=(N-1)/2;

for(i=0;i<k;i++)

{

Wn[i]=0.5\*(1-cos((2\*pi\*(double)i)/(N-1)));

hd[i]=-(sin(2\*pi\*fc\*(double)(i-k)))/(pi\*(i-k));

h[i]=hd[i]\*Wn[i];

h[N-i-1]=h[i];

}

h[i]=(1-2\*fc)\*0.5\*(1-cos((2\*pi\*(double)i)/(N-1)));

}

**Filtering of noisy signal using FIR filter(Processor)**

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

void main()

{

int N,i,k,j,n=15;

double fp=2000,fs=5500,Fs=20000;

double hd[14],Wn[14],h[28],pi=3.1415926,fc,TW;

fc=((fp+fs)/(2\*Fs));

TW=fabs((fs-fp))/Fs;

N=ceil(4/TW);

if(N%2==0)

N=N+1;

k=(N-1)/2;

// To design FIR Low pass filter

for(i=0;i<k;i++)

{

Wn[i]=0.5\*(1-cos((2\*pi\*(double)i)/(N-1)));

hd[i]=(sin(2\*pi\*fc\*(double)(i-k)))/(pi\*(i-k));

h[i]=hd[i]\*Wn[i];

h[N-i-1]=h[i];

}

h[i]=fc\*(1-cos((2\*pi\*(double)i)/(N-1)));

// To generate signal with random noise

for(i=0;i<=n;i++)

{

number=rand();

x[i]=sin(2\*pi\*10/100\*i)+0.00001\*number;

y[i]=x[i];

}

for(j=10;j<25;j++)

{

y[j]=0.0;

for(i=0;i<=j;i++)

y[j]+=h[i]\*x[j-i];

}

}