Name: MANSI UNIYAL
Roll No.: 19EE10039
Experiment 4
Title - Digital Highpass FIR filter implementation on Arduino Hardware

---

Objective:
To Program ATMEGA328p in Arduino Uno to filter the input signal received in the ADC port of the microcontroller and gives a filtered output which we can observe in the serial plotter as a virtual output.

Requirements:
1. Arduino Uno Board
2. USB A to USB B cable
3. Arduino IDE

Methodology:
Build the required digital filter parameters in MATLAB and use the parameters in assembly code to program the ATMEGA328p microcontroller in Arduino.

MATLAB code:-
```
%Filter Specifications
M=32;
fs=2000;
fc=500; % high pass filter
wc=2*fc/fs;
h = fir1(M,wc, 'high'); % for highpass
h_fixed=fi(h, 1, 8, 7);

%Plot filter gain vs frequency response
fvtool(h);
title('Designed filter');

fvtool(h_fixed);
title('Fixed point filter');

%Input Signal
L=500; f1=50; f2=500;
sig=zeros(L,1);
for i=1:L
   sig(i)=sin(2*pi*f1*i/fs)+sin(2*pi*f2*i/fs);
end
```

```
sig_fixed=fi(sig, 1, 8, 7);

%Do convolution to get output signal
y=conv(sig,h);
y_fixed=conv(sig_fixed,h_fixed);

%Plot input & output signal
figure(1)
subplot(2,2,1)
plot(sig)
title('Input Signal');
subplot(2,2,2)
plot(sig_fixed)
title('Fixed point Input Signal');
subplot(2,2,3)
plot(y)
title('High Pass Filtered Signal');
subplot(2,2,4)
plot(y_fixed)
title('Fixed point High Pass Filtered Signal');

%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)
   hh=h_fixed(i)*2^8;
   if i<length(h_fixed)
      fprintf(file1, '%d, ', hh);
   else
      fprintf(file1, '%d', hh);
   end
end
fclose(file1);
file2=fopen('Input Signal Data from MATLAB.txt', 'w');
for i=1:length(sig_fixed)
   si=sig_fixed(i)*2^8;
   if i<length(sig_fixed)
      fprintf(file2, '%d, ', si);
   else
      fprintf(file2, '%d', si);
   end
end
fclose(file2);
```

This code generates two files that contain input signal data(50Hz) with noise(500Hz) and the filter parameters. Now we can write the C++ code in Arduino IDE for the ATMEGA328p microcontroller to perform the digital filtering on the provided input signal mixed with high-frequency noise.

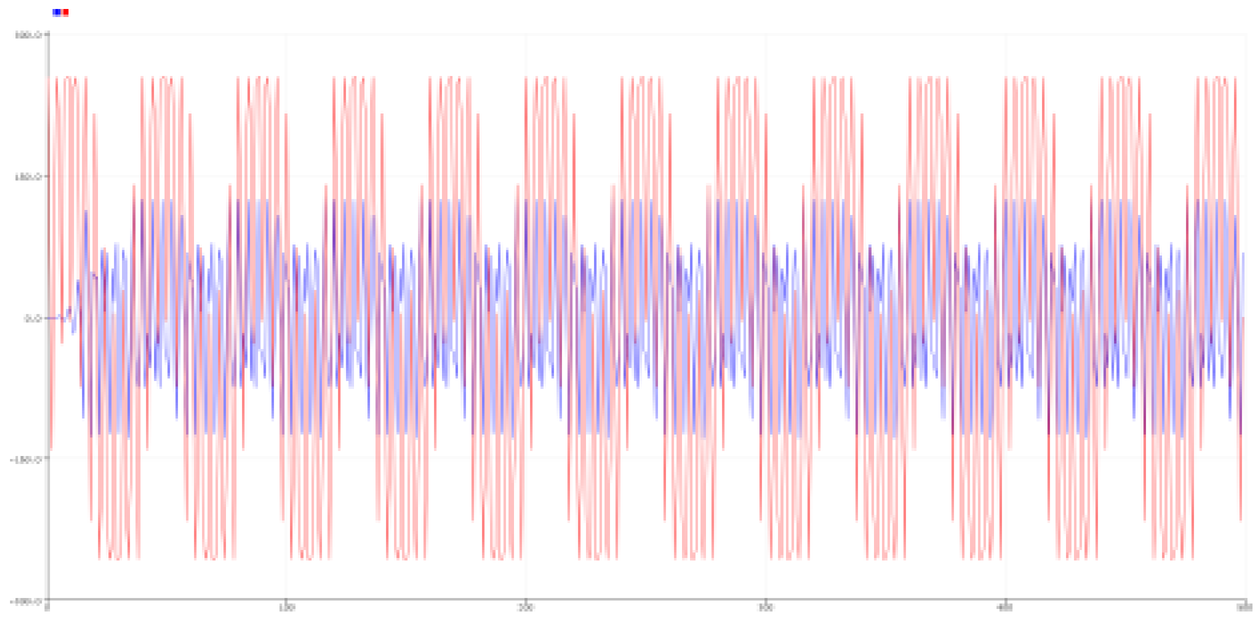
Code:

Arduino IDE code:-

```
int m = 500, n = 30;
int x[500] = {254, 80, -140, 150, 254, 208, -28, 244, 254, 254, -4, 244, 254, 208, -74,
150, 254, 80, -216, 0, 216, -80, -256, -150, 74, -208, -256, -244, 4, -256, -256, -244, 28,
-208, -256, -150, 140, -80, -256, 0, 254, 80, -140, 150, 254, 208, -28, 244, 254, 254, -4,
244, 254, 208, -74, 150, 254, 80, -216, 0, 216, -80, -256, -150, 74, -208, -256, -244, 4,
-256, -256, -244, 28, -208, -256, -150, 140, -80, -256, 0, 254, 80, -140, 150, 254, 208,
-28, 244, 254, 254, -4, 244, 254, 208, -74,
 150, 254, 80, -216, 0, 216, -80, -256, -150, 74, -208, -256, -244, 4, -256, -256, -244, 28,
-208, -256, -150, 140, -80, -256, 0, 254, 80, -140, 150, 254, 208, -28, 244, 254, 254, -4,
244, 254, 208, -74, 150, 254, 80, -216, 0, 216, -80, -256, -150, 74, -208, -256, -244, 4,
-256, -256, -244, 28, -208, -256, -150, 140, -80, -256, 0, 254, 80, -140, 150, 254, 208,
-28, 244, 254, 254, -4, 244, 254, 208, -74, 150, 254, 80, -216, 0, 216, -80, -256, -150,
74, -208, -256, -244, 4, -256, -256, -244, 28, -208, -256, -150, 140, -80, -256, 0, 254, 80,
-140, 150, 254, 208, -28, 244, 254, 254, -4, 244, 254, 208, -74, 150, 254, 80, -216, 0,
216, -80, -256, -150, 74, -208, -256, -244, 4, -256, -256, -244, 28, -208, -256, -150, 140,
-80, -256, 0, 254, 80, -140, 150, 254, 208, -28, 244, 254, 254, -4, 244, 254, 208, -74,
150, 254, 80, -216, 0, 216, -80, -256, -150, 74, -208, -256, -244, 4, -256, -256, -244, 28,
-208, -256, -150, 140, -80, -256, 0, 254, 80, -140, 150, 254, 208, -28, 244, 254, 254, -4,
244, 254, 208, -74, 150, 254, 80, -216, 0, 216, -80, -256, -150, 74, -208, -256, -244, 4,
-256, -256, -244, 28, -208, -256, -150, 140, -80, -256, 0, 254, 80, -140, 150, 254, 208,
-28, 244, 254, 254, -4, 244, 254, 208, -74, 150, 254, 80, -216, 0, 216, -80, -256, -150,
74, -208, -256, -244, 4, -256, -256, -244, 28, -208, -256, -150, 140, -80, -256, 0, 254, 80,
-140, 150, 254, 208, -28, 244, 254, 254, -4, 244, 254, 208, -74, 150, 254, 80, -216, 0,
216, -80, -256, -150, 74, -208, -256, -244, 4, -256, -256, -244, 28, -208, -256, -150, 140,
-80, -256, 0, 254, 80, -140, 150, 254, 208, -28, 244, 254, 254, -4, 244, 254, 208, -74,
150, 254, 80, -216, 0, 216, -80, -256, -150, 74, -208, -256, -244, 4, -256, -256, -244, 28,
-208, -256, -150, 140, -80, -256, 0, 254, 80, -140,
 150, 254, 208, -28, 244, 254, 254, -4, 244, 254, 208, -74, 150, 254, 80, -216, 0, 216,
-80, -256, -150, 74, -208, -256, -244, 4, -256, -256, -244, 28, -208, -256, -150, 140, -80,
-256, 0, 254, 80, -140, 150, 254, 208, -28, 244, 254, 254, -4, 244, 254, 208, -74, 150,
254, 80, -216, 0};

int h[33] = {0, 0, 0, 0, 0, 2, 0, -4, 0, 8, 0, -12, 0, 26, 0, -80, 128, -80, 0, 26, 0, -12, 0, 8, 0,
-4, 0, 2, 0, 0, 0, 0, 0}; //highpass
```

```
long int y = 0;

void setup() {
  // initialize serial communication with 9600 bits per second:
  Serial.begin(115200);

  //convolution
  for (int i=0; i<m+n; i++){
    y = 0;
    for (int j=0; j<n; j++){
      if ((i-j)>=0 && (i-j) < m)
        y = y + (h[j]*x[i-j])/256;
    }
    Serial.print(y);
    Serial.print(", ");
    if (i < m)
      Serial.println(x[i]);
    delayMicroseconds(500); //0.5ms delay
    }
}

void loop() {

}
```

Results:
The simulated output as observed in the lab is produced. The Arduino output can be observed in the serial plotter, which shows a filtered output signal of 500Hz frequency with very little noise in the output.



Discussions:

In the case of the High-pass filter, of the two signals(50 Hz and 500 Hz) the 50 Hz signal is blocked. The cutoff frequency of the filter being 500Hz, the 500Hz signal is allowed to pass