For: - Avantari Technologies Pvt. Ltd.

AIM

When working with analog signals, one of the most pervasive noise elements is the power line noise. This signal (link), sampled at 200Hz, is an ECG signal, which is corrupted by power line frequency (50 Hz). The expected ECG signal should look like this.

We need you to write code, to read input signal from a file, clean the signal of the power line noise, detect the peaks, and output a clean ECG Signal as well as the heart rate detected from the signal. Also save clean ECG Signal into another file.

TASKS

a) Remove power line noise from given ECG signal.

b) Calculate average heart rate of filtered ECG signal

SOLUTION

As per the required aim and task at hand, after analyzing the problem statement and the basics of ECG measurements below is the proposed solution.

Note that the attached ecg_processing.c file has all the functions and code required for the same.

Now this is a typical system of DSP where the DAQ stage has been performed and the raw data is provided to us for further processing including the filtering and calculation of respectable output i.e. the heart rate in our case.

Since the sample frequency is 200Hz, the maximum input frequency possible to avoid error free sampling in accordance with the Nyquist formula is less than 100Hz. Assuming that the higher frequency components are noise and the removing the 50Hz power noise we can get seemingly clear wave for further processing and heart rate calculation.

Procedure:-

Designing a Low pass filter to remove high frequency noises.

Band stop filter to stop the 50Hz frequency (45-55Hz in our case)

Calculation of heart rate.

Void RemovePowerlineNoise(float *inputsignal, float *outputsignal, unsigned int size, unsigned int samplefreq)

```
/* coefficients for the filter:-

float b0 = 0.8;

float b1 = 0;

float b2 = 0.8;

float a1 = 0.2;

float a2 = 0.8;
```

Note that these values are generated by designing a butterworth filter with notch frequency 45-55Hz and sample frequency of 200Hz

Python code to get coefficients:-

```
fs = 200

f1 = 45

f2 = 55

sos = signal.butter(2,[f1/fs*2,f2/fs*2],'stop',output='sos')
```

```
*/

{

        Int x1, x2, y1, y2 = 0;

        for(int i = 0; i<size ; i++)

        {

                out[i] = c_b0*in[i] + c_b1*m_x1 + c_b2*m_x2 - c_a1*m_y1 - c_a2*m_y2;

                m_x2 = m_x1;

                m_y2 = m_y1;

                m_x1 = in[i];

                m_y1 = out[i];

        }

}
```

This function gives output array which can be stored in a csv file for further usage.

Note:- I was not able to get exact values as shown in the output file however the graphs were as follows.

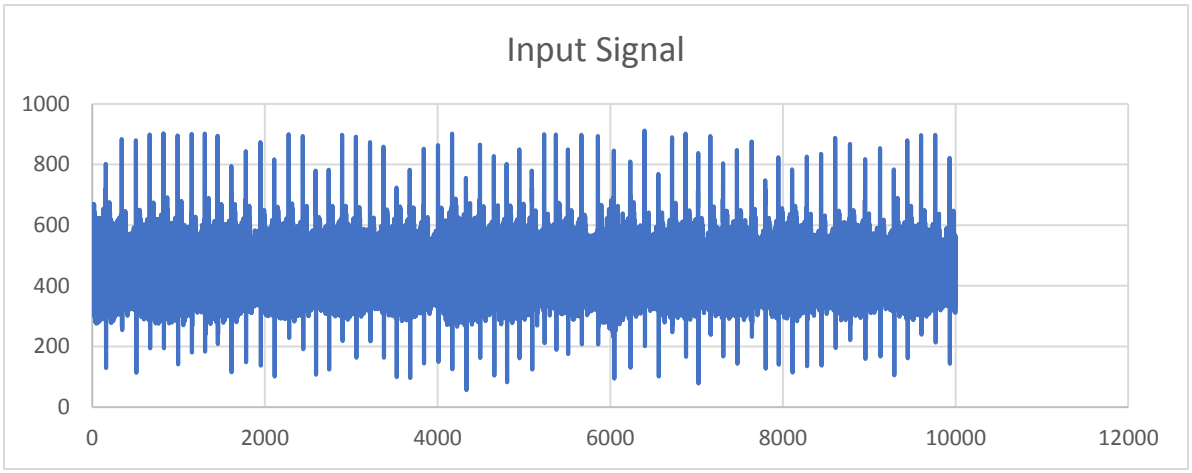

Fig :- The Input signal of raw data plotted on MS-Excel

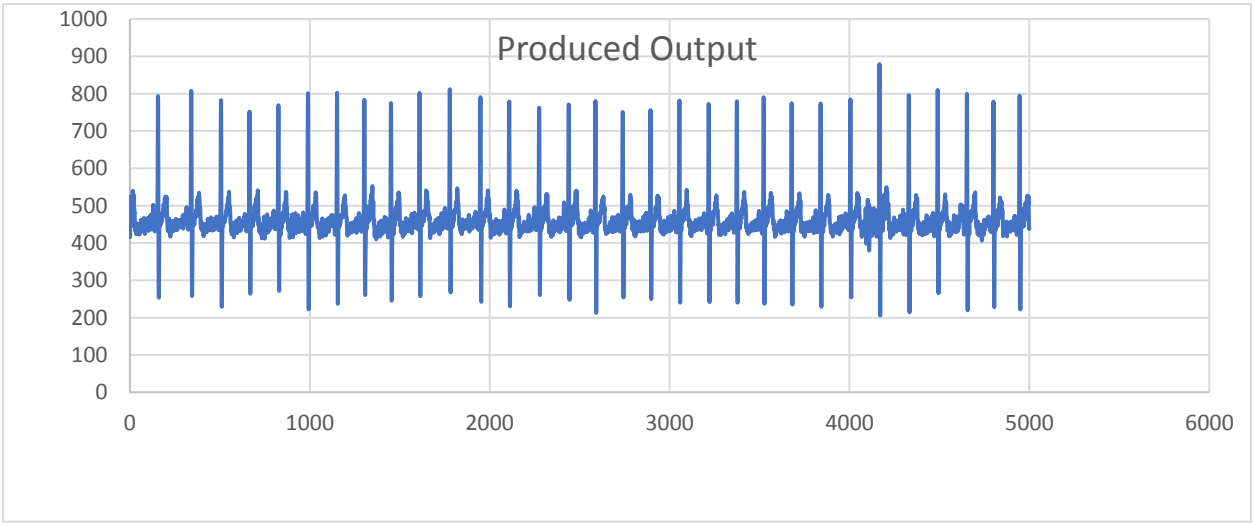

Fig:- The Produced output after applying band reject filter

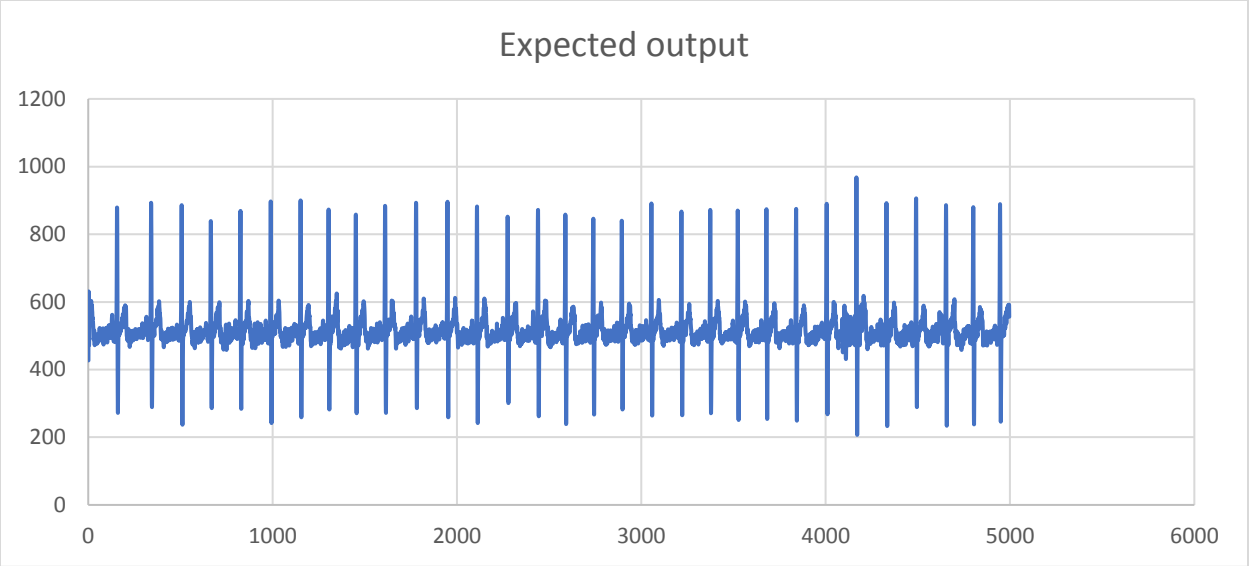

Fig:- The expected curve as plotted using data on excel

```
float ProcessEcgSignal( float *ecgSignal, unsigned int size, unsigned int samplingFreq)
{
   unsigned long sum;
   float avg = 0.0;
   float avg_diff = 0.0;
   float beat = 0;
   unsigned int sum_diff = 0;
   unsigned int prev = 0;
   int count = 0;
   for (int i=0; i<size ; i++)
   sum = sum +ecgSignal[i];
   avg = sum/size;
   for (int i=0; i<size; i++)
   {
        if((ecgSignal[i]-avg)>200)
     {
```

```
        sum_diff = sum_diff + i - prev;
        prev = i;
        i+=5;
        count++;
      }
}

avg_diff = sum_diff / count;
beat = (samplingFreq / (float)avg_diff)*60;
   printf("\nthe heart beat is    %f",beat);
   printf("\nthe average is    %f",avg);
   printf("\nthe average diff is    %f",avg_diff);
}
```

Notes:- ***The code has been checked using a CLI based g++ compiler with below version details:***

Compiler Version = g++ (Ubuntu 7.4.0-1ubuntu1~18.04.1) 7.4.0 Copyright (C) 2017 Free Software Foundation, Inc.

Machine = Linux machine (Ubuntu 18.04)

The output of the same was:-

the heart beat is   75.000000

the average is   367.000000

the average diff is   160.000000