RESEARCH PAPER | Engineering | Volume : 5 | Issue : 10 | October 2015 | ISSN - 2249-555X

Heart Beat Scanner for Smart Phones

KEYWORDS: Heart Beat Scanner, MATLAB, Arduino microcontroller, IR sensor, pulse, signal filter.

Rohan Kelkar

J-Block, VIT Men's Hostel, VIT University, Vellore

ABSTRACT *In today's world of mobile technology, every other person owns a smart phone using which he/she carries out numerous number of tasks in a day. Estimates show that almost 2 billion consumers will own smart phones by 2016[1]. Since smart phones are always carried by almost all individuals, certain health related technologies can be installed in the devices as standard such as a heart beat scanner which can prove to be life-saving in certain scenarios of emergency. This led to the prototyping of a cheap and easy to build heart beat scanner using an IR sensor interfaced with an Arduino Uno microcontroller, 'Processing' software and MATLAB. This prototype has been designed to detect change of blood flow in our veins caused due to the beating of the heart, and thus give a reading of our pulse rate after undergoing the necessary calculations and filtration techniques.*

INTRODUCTION

This project attempts to successfully create a cheap pulse rate calculator system for smart phones which uses IR sensors to detect the fluctuation of blood flow in the veins which takes place according to the heart beat and further uses MATLAB and Processing to graphically display the pulse graph as well as the reading in BPM on the computer screen.

The generated signal from the IR sensor is made to pass through two low pass filters connected in series which successfully eliminate the low frequency noise introduced in the signal due to various factors such as temperature fluctuations etc. The filtered signal is then transferred via a USB interface to the Processing software which plots the graph of the input signal on the screen.

MATLAB is also made to run in parallel and fetches the same signal via the USB interface. A peak detection algorithm is run in MATLAB which counts the number of peaks generated in the signal in a predefined amount of time. Thus the pulse rate is calculated and displayed.

HARDWARE

The IR sensor is designed by using a standard IR emitter and detector. Whenever the detector receives radiation emitted by the emitter, a voltage fluctuation is observed at the output terminal of the detector[2]. This signal consists of a lot of noise due to various unavoidable natural problems such as temperature fluctuations etc. These fluctuations cannot be contained in the real world therefore various filtering techniques need to be applied in order to obtain a cleaner, low noise signal.

This signal is made to pass through LM386 amplifier IC which amplifies the amplitude of the signal making the fluctuations easy to detect. Pin1 and Pin8 of the IC are coupled with a 1uF capacitor in order to ensure uniform amplification over a long range of frequencies. The signal received from the LM386 amplifier is made to pass through two passive RC low pass filters connected in series in order to eliminate the low frequency noise and thus leave us with a cleaner signal which can be further processed for peak detection[3].

We have further used an Arduino microcontroller to interface the circuit with the computer via a USB interface[4]. The amplified and the filtered signal from the previous circuit is taken as input in one of the analog pins of the Arduino microcontroller. A baud rate of 9600 bits per second is set as a serial communication parameter initially. The baud rate of the transmitter, which is the Arduino, and that of the receiver, which is the Processing software, must be set to the same value. If the baud rate of the receiver differs from the baud rate of the transmitter, loss of data may occur and successful transfer of observed signal will not take place. The signal is then transferred to the Processing software to plot a graph.

SOFTWARE

Processing is the software used for mapping the fluctuations of the input signal on the screen. Importing the serial library was necessary as building a serial communication interface had to be done in order to receive the input signal from the microcontroller[5]. A serial event handler was used which is called every time data is received from the serial interface. It calls a draw() function which is basically responsible for mapping the graph on the screen based on the amplitudes received.

MATLAB is responsible for calculating the pulse rate from the signal received. It too needs to receive the signal from the serial port and thus it is connected to the same COM port as the Arduino with baud rate set to 9600 bits per second. This software also has a serial event handler which is called every time data is received[6]. Its main purpose is to note the number of peaks in the input signal with respect to time and thus calculate the pulse rate as every peak corresponds to a diastole or a systole of the heart.

WORKING

The IR sensor is placed in such a way under the tip of our finger, that IR radiations enter the vein at an oblique angle and reflect back due to the presence of blood flow[7]. Whenever there is a random change in the blood flow, the intensity of the reflected radiation changes which helps in marking a heartbeat. Whenever the heart beats, the signal received at the receiver end fluctuates in an unusual manner. This can either be a systole or a diastole of the heart but is considered as one beat.

Volume : 5 | Issue : 10 | October 2015 | ISSN - 2249-555X

The fluctuations at the receiver end are of a very low intensity. Due to this, it becomes very difficult to differentiate between the actual signal and noise. Therefore the signal is made to pass through the LM386 amplifier IC[8]. This amplifier amplifies all the fluctuations present in the signal and thus helps us get a stronger signal. Every minute detail now becomes very clear and thus the signal becomes relatively easy to process. But, this amplifier IC cannot differentiate between the noise signal and the actual signal. The noise introduced by the passive components in the circuit is of a particular low frequency.

To eliminate the noise present in the circuit, two passive RC low pass filters are used in series[9]. This ensures that the low frequency noise be eliminated and we thus obtain a cleaner signal which can be easily processed further in order to calculate the pulse rate. The low pass filters introduce a voltage drop in the signal but it is not a significant drop and thus it does not reduce the probability and the ease of finding the peaks.

After amplification and filtration of the signal, it is then made to pass into an Arduino microcontroller. The Arduino microcontroller receives the analog signal at one of its analog input pins. The microcontroller is programmed to fetch this input stream of analog data and assign it to the serial port. An input stream event handler function is written which on receiving any data on the input pin, writes it to the assigned serial port and keeps doing it until no data is received. The baud rate is set to 9600 bits per second and thus the input signal is sampled according to this rate and is further transferred to the Processing software via the serial interface.

The Processing software is responsible for mapping the input signal stream in the form of a graph[10]. Initially the software creates a window in which all the drawing takes place. In the Setup() function, various parameters such as the size of the window, thickness of the brush, color of the brush, color of the background etc are set. Next, an event handler function is written which runs every time data is received via the serial port. This function calculates the amplitude of the received signal and assigns it to the y-axis variable. The signal is plotted with respect to time and therefore, time is assigned to the x-axis variable. The time variable depends on the sampling rate of the system. After assigning values to the x-axis and the y-axis variables, the draw() function is called and these parameters are passed. This function draws the lines connecting all the points. Since the sampling rate of the system is very high, a high resolution signal with a smooth graph is obtained.

MATLAB runs simultaneously and is responsible for calculating the pulse rate from the input signal. Since the calculations done in MATLAB are dependent on the same input signal, this software too receives data from the same serial port. It is also set to receive data at 9600 bits per second. An inbuilt function called findpeaks() is used in order to calculate the number of peaks experienced. Certain parameters such as minimum peak height and frequency range were set so that unnecessary fluctuations could be ignored and the target portion of the signal could be focused upon. Initially a delay of two seconds is introduced which acts as a buffer time for the signal to settle and attain uniformity in its fluctuations. Next, the number of peaks are calculated for every 1 second. Thus, after multiplying the figure by 60 we estimate the number of peaks occurring in a minute. This amounts to the number of times the heart beats in a second. This figure is then displayed on the screen. The findpeaks() function is called after every 5 seconds and thus it keeps displaying a refreshed amount.

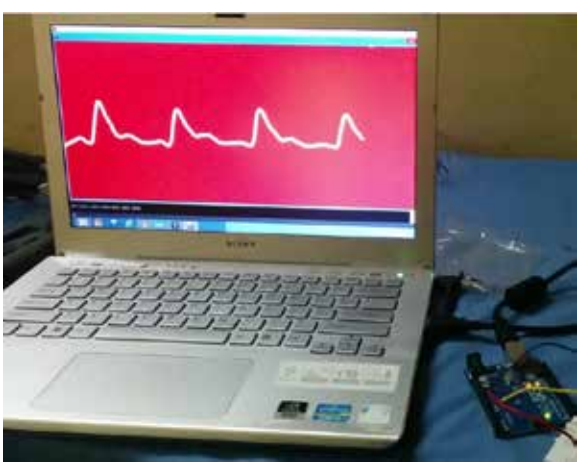
RESULTS:

Trail Number 1

No. of peaks	6
Bits per Sample	8
Buffer Length	0.1000
DeviceID	-1
Number Of Buffers	10
Number of Channels	1
Running	'on'
Sample Rate	9600
Timer Period	5 sec
BPM	72

Trial Number 2

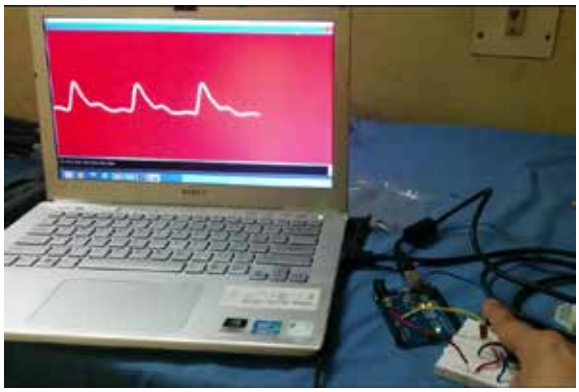
No. of peaks	5
Bits per Sample	8
Buffer Length	0.1000
DeviceID	-1
Number Of Buffers	10
Number of Channels	1
Running	'on'
Sample Rate	9600
Timer Period	5 sec
BPM	60



Trial Number 3

No. of peaks	7
Bits per Sample	8
Buffer Length	0.1000
DeviceID	-1
Number Of Buffers	10
Number of Channels	1
Running	'on'
Sample Rate	9600
Timer Period	5 sec
BPM	84

Volume : 5 | Issue : 10 | October 2015 | ISSN - 2249-555X



APPLICATIONS

- This prototype can be implemented in modern day smart phones. People with high blood pressure will find this feature extremely handy as they will be able to check if their pulse rate has elevated or not with great ease.
- During cases of emergency in which there is a threat of a person dying, this feature will prove extremely useful as we can easily check the availability of a pulse.
- This system can also be installed on smart watches and thus there will be no need to place the finger on the sensor as the sensor will be placed right under the vein on our wrist.

FUTURE PROSPECTS

- If this system is connected through internet, a registered doctor can immediately be notified or an ambulance can immediately be summoned at the time of an emergency.
- If the system detects a problem, it can also send text messages to people who are closest to the devices current location and thus can call for help.

REFERENCE [1 http://www.emarketer.com/Article/2-Billion-Consumers-Worldwide-Smartphones-by-2016/1011694 | [2http://www.elprocus.com/infrared-ir-sensor-circuit-and-working/ | [3]http://www.electronics-tutorials.ws/filter/filter_2.html | [4] https://www.arduino.cc/en/Reference/Serial | [5] https://processing.org/reference/libraries/serial/ | [6]http://in.mathworks.com/help/matlab/matlab_external/events-and-callbacks.html | [7]http://vtc.internshala.com/course/content.php?topic_id=15&module_id=2&course=robotics101&demo=true | [8] http://www.ti.com/lit/ds/symlink/lm386.pdf | [9] https://en.wikipedia.org/wiki/Low-pass_filter | [10] https://processing.org/reference/map_.html