--- Documentation ---

Project Zieg

Polygraph - The Lie Detector Machine

Team Wiara

Team ID: 1

(By Kiran Dhope and Karan Taneja)

Links:

GitHub Repository:

- **https://github.com/KrnTneja/PolygraphZieg**

Videos:

- **https://www.youtube.com/watch?v=kgVUFpzo6Ig**
- **To be updated**

About the final product:

Polygraph Machine can be divided into 2 parts:

- Electronics and Hardware
- Graphic User Interface

Electronics and Hardware can also be further divided into 4 parts:

- Skin conductivity measurement.
- Pulse measurement.
- Respiration rate measurement.
- Blood pressure measurement (that couldn't be completed).

Graphic User Interface can be further divided into 3 parts:

- Serial communications with Arduino.
- Processing of raw data to graph data at back end.
- Plotting of graph and other control features.

Electronics and Hardware

Skin conductivity measurement:

A voltage divider is used to measure the body resistance (and not skin conductivity implicitly). A known resistance of 560k is connected to VCC and the body. Other side of body is connected to ground in the circuit. The voltage across body is fed to analog pin of Arduino.

Pulse measurement:

Component Required: IC LM324 (quad op-amp), IR LED, IR Sensor, a red LED, resistors, and capacitors.

Note: The technique used is called Photoplethysmography (PPG).

There is an IR LED whose light is reflected by the skin and received by an IR sensor. The changes in volume of blood over the time changes the amount of reflection which is detected by the IR sensor and it is amazing how we can filter and amplify to get the pulse in desired range.

There are two types of unwanted voltage signals in the signal received from input: the constant DC voltage and the noise from surroundings. For removing the DC voltage, there is (passive) high pass RC filter with 0.7Hz as cut off frequency. To remove the noise, there is (active) low pass op-amp filter with 2.3Hz as cut off frequency and gain of nearly 100.

So in this module, there is, in net effect, a band pass filter that allows 0.7Hz to 2.3Hz and amplifies the signal by 100. But this provides neither sufficient filtering nor sufficient amplification. To further filter and amplify the signal, the signal from first module is fed to an exact duplicate module containing passive high pass filter and active low pass filter.

After this, amplification is nearly 10,000 and signal has frequency component only between 0.7Hz and 2.3Hz. One can now see a pulse with little effort but only with amplitude of around 0.5V. So the signal is further fed to an op-amp with gain of 10.

This output can be fed to a red LED (voltage controlled by resistor) or analog pin of a microcontroller or DSO to observe the pulse clearly.

Respiration rate measurement:

Two flex sensors are bent and attached to two straps one of which goes around the chest and other around the stomach. As the expansion takes place, flex sensors straighten up a little and resistance decreases and vice versa.

A voltage divider is used to measure the flex sensor resistance. A known resistance of 68k is connected to VCC and the flex sensor. Other side of flex sensor is connected to ground in the circuit. Also, to remove the noise, a capacitor is put parallel to flex sensor. The voltage across flex sensor is fed to analog pin of Arduino.

Graphic User Interface

Serial Communication with Arduino:

This consists of two components: the Arduino code and the Java code.

The Arduino code simply reads the input from analog pins and writes it over Serial using Serial library of Arduino. The code used follows a protocol to tag the data to a particular sensor for computer to be able to distinguish the values according to sensor.

The Java code is a little complex. It uses the RxTxComm.jar for Serial connection. Official documentation and example code for this can be found easily on internet. The library provides easy access to ports, identification of ports by name, connection to ports, etc.

Processing of raw data to graph data at back end:

For the pulse circuit, the voltage is simply plotted against time. The digital output from DAC (say D) has to be scaled down to 5V using

$$V_O = \frac{D}{1024} \times 5.0$$

For skin conductivity and respiration rate, voltage divider is being used where the voltage across the resistance to be measured is given by

$$V_O = \frac{R_{unknown}}{R_{unknown} + R_{known}} \times V_{CC}$$

From above equation we can find the unknown resistance. Also, voltage has to be scaled down to 5V as it is done is pulse measurement.

Every data point is parallel to a corresponding time-stamp with precision of milliseconds in the program. When plotted, the time-stamps are used to get X co-ordinate of the point rather than directly dividing the length into equal parts. This gives the freedom to send more data for a particular value and less for another without making any explicit changes in the program.

Plotting of graph and other control features:

Suppose that resistance unknown resistance (say R_x) lies in the range R_1 to R_2 ($R_1 < R_2$) and this has to be zoomed to fit into say n pixels. Simple calculations show that the height *h* pixels above the base level is given by

$$h = (\frac{R_x - R_1}{R_2 - R_1}) \times n$$

So point is plotted with height *h* when resistance is R_x. This feature is used for both body resistance and respiration rate.

To get the pulse, we note the time of the graph at which it crosses a particular value (say t_1) and the also the time when it crosses that same value next time (say t_2). Pulse can be given by (if t_1 and t_2 are in milliseconds)

$$Pulse = \frac{60,000}{t_2 - t_1}$$

That value at which time is recorded is called the ‘trigger value’.

About the problems and modifications we had:

Electronics and Hardware

Skin conductivity measurement:

- We earlier thought of measuring resistance only about two hands but later realized that forehead is most common region of high sweating. So we decided to put up same components on forehead as well.
- We thought of making E-meter for this purpose using INA128 as amplifier for the signal from wheatstone bridge but later found out that change in resistance is comparable (almost 25 to 50 percent) to actual resistance and hence the voltage output from the amplifier cannot be approximated to be linear. And the region where reasonable change could be observed, depended on a lot of adjustments to bridge resistances. But since the change was comparable, we also found out that a simple voltage divider is enough to bring changes out to our notice. Also, the ADC of arduino is 10 bit which, as we found out, is sufficient to observe the changes.
- Though the final product doesn't give exact skin resistance but it is accurate up to a factor of 1 to 3 kilohms and thus provides reasonable opportunity to observe any change in it which is of order of 10 kilohms.

Pulse measurement:

- We tried a lot of circuits with different filters and amplifications since we were not sure of the exact form of input. We never expected the need to amplify pulse signal by 100,000 to get the required pulse. Also, there was a need of lot of filtering before signal can be amplified.
- An important point to note is that high amplification should always follow high pass filtering since DC voltages should be eliminated to avoid saturation in amplifier. One should remember that filters are non-ideal and there might be need of multiple similar filters before signal can be amplified.
- We had a lot of searching for TCRT1000 since we heard of it on all the similar projects on internet. We found out, later however, that putting up TCRT 5000 or even standard IR sensor and IR LED works.
- We thought of putting up LED and sensor on wrist but it didn't seem to work because of reasons unknown to us. But it generally works when put up reasonably tight for a little time (say around 20 seconds). We found this satisfactory and accepted as final.
- In a facebook discussion, we concluded that observation point out to the fact that convenience of pulse being found on the thickness of finger.

Respiration rate measurement:

- Our plan initially started with an abstract thought of measuring, using some way, the rate and depth of breathing of examinee. And finally, we found out that the best way is to measure the relaxation and contraction of chest and stomach as person breathes.

- Any shift of stress during the test from lungs to stomach or vice versa, or complete relaxation, or highly abrupt breathing can, in theory, be observed and attributed to a particular mental state.
- The theory sounds beautiful but the problems arose when we were searching for a way to measure stretch of the elastic. We found an unclear hint on internet that something called ‘strain gauge’ is used to measure the stretch only to find out later that it is used for metals and was useless for our purposes.
- Second thing that attracted us was strain gauge and we easily thought of a solution to modify it for our purposes as mentioned above. This idea was acceptable with reasonable testing of resistance change with bending. We found out that the resistance change was again enough to measure it using a voltage divider so we finalized the circuit too. (It was a momentary decision reasoned by the satisfaction from simple skin conductivity circuit.)
- On actual testing on a examinee, we observed a lot of noise in the output and we decided to put up a capacitor, in parallel to flex sensor, as a quick fix. The thing finally seemed to work fine.

Blood pressure measurement:

- We tried to find an appropriate sensor which had a range that was sufficient for our purposes.
- We also had to take care that sensor shouldn’t be too costly. Most of sensors we found were either not for the range we needed or their range was too big which made them costly. We finally found a sensor MPS20N0040D-D which has a range of 0 to 40kPa (0 to 300mmHg) and costed Rs.390/-.
- The sensor took more than three days to arrive when ordered online, it wasn’t available in Mangaldeep or Lamington, hence leaving us with no other option.
- When we tested the pressure sensor, the readings seemed to vary drastically from what we expected, hence making us conclude the circuit diagram given in datasheet is indeed faulty.
- We tried to deduce the actual diagram from observations but it didn’t turn out to be as simple as expected. Even voltage division across the resistances was not as we expected. Some point were showing unnecessary voltages even when theoretically they shouldn’t.
- So we decided to use only the diaphragm resistance of the sensor and make an external circuit for rest of the part. We did make a beta circuit from the observations, but all the circuit does, is registering changes in the pressure given to it, which doesn’t satisfy the needs of a blood pressure measurement device. We needed to make the signal smooth (filtered) enough to make any vibrations in pressure visible.
- Not only that, the sensor lacked the precision that we might need for the same. Long story in short, we needed to add additional (more than we expected) circuitry to make it useful for our purposes.
- The air pump motor we ordered was said to work in the range of 9-12V, upon giving more than 7V, the motor stopped working, we disassembled the motor to check what’s

wrong, all in vain. When we put it back together, the motor started working again, only to stop working after reaching 9V this time. We disassembled it again and figured a plastic rotor part in motor heating up more than usual and partially melting damaging its edges and it worked, never again after that.

- We needed a new pump altogether. We also needed to make sure that the pump is strong enough to generate a pressure up to 300 mmHg in the system.
- The time was hitting us hard. A lot of time was taken up in components delivery and then unsuccessful testing. We concluded, on discussion with seniors, that it is an ITSP in itself and we focused, from then on, on the final product preparation and testing.

Graphic User Interface

Serial Communication with Arduino:

- Initially we thought of using Python to code up serial, GUI and graph plotting but later choose copy up the basic framework of a very similar code we wrote up in our Winter Project 2015 which was coded up in Java.
- The Serial communication in Java requires RXTXcomml.jar and rxtxSerial.dll (from http://jlog.org/rxtx-win.html) to be put up in appropritate folder to work. We used NetBeans IDE.
- When we were able to send and receive data using Arduino and out Java program, we saw that output being received was not what was expected. On observing the output, we concluded that there was some fault in synchronization. It seemed as if the Java program is processing data at a slower rate than the Arduino is sending it. So we put up some delay (1 to 3 ms) between sending two data points in the Arduino code. It worked. We later attributed each data point to a particular sensor code. To cover up the time loss in delay, we increased the baud rate and it still worked fine. We finalized on it.
- Sensor code is made up of a alphabet and a number. For example ‘c1’ means skin conductivity sensor numbered 1, ‘r2’ means respiration rate sensor numbered 2. This code is just followed (without any space) by the 10 bit output of ADC and then by a newline character. For example “c1102\n” is understood as conductivity sensor numbered one sent a value of 102. This corresponds to scaled voltage as mentioned earlier.
- When the code was finally deployed to form the final JAR executable file, we found out that we couldn’t find the ports. Inside NetBeans, everything seemed to work fine. The problem, as we found out later was that NetBeans needs 64-bit rxtxSerial.dll file while JAR file being executed needed 32-bit version of same file. So, the simplest solution was to put up the 32-bit rxtxSerial.dll in the ‘dist’ folder of the NetBeans project. And it worked!
- To make the Java program faster, without getting delayed in receiving data, we used multi-threading using Thread class in Java to create a new thread for data as soon as minimal processing was done.

Processing of raw data to graph data at back end:

- We just took care that there are minimum routines for data-processing and calculations to make it as fast as we could.

Plotting of graph and other control features:

- There was a need to simultaneously plot six graphs (including the blood pressure one). What the program does is that it plots at rate of 4 frames per second which seem okay. For every frame, program cleans up screen, plots reference lines, gets updated coordinate list of a sensor, plots it and similarly continues getting updated list and plotting for rest of them.

About further scope and improvements:

Electronics and Hardware

Pulse measurement:

Pulse measurement can be experimented with, to find a foolproof method to get pulse for every examinee without any exception.

Respiration rate measurement:

The signal received from the device can be improved by proper filtering and amplification. Since the resistance change (say 2k) seems to be much less that actual value of flex sensor resistance (65k), one can put up bridge circuit and add an amplifier to get much better output.

Graphic User Interface

Serial Communication with Arduino:

Code might be improved to increase the data sent per second.

Plotting of graph and other control features:

Better methods can be used to get non-fluctuating pulse on display. GUI can be automated to get zoom into the right region. Also, program can be automated to show pop-ups in situation of high fluctuations which will also need the program to store the baseline values.

Other modifications

Blood pressure measurement

A blood pressure measurement device which supports continuous monitoring can be made and installed to get better update.

Expressions

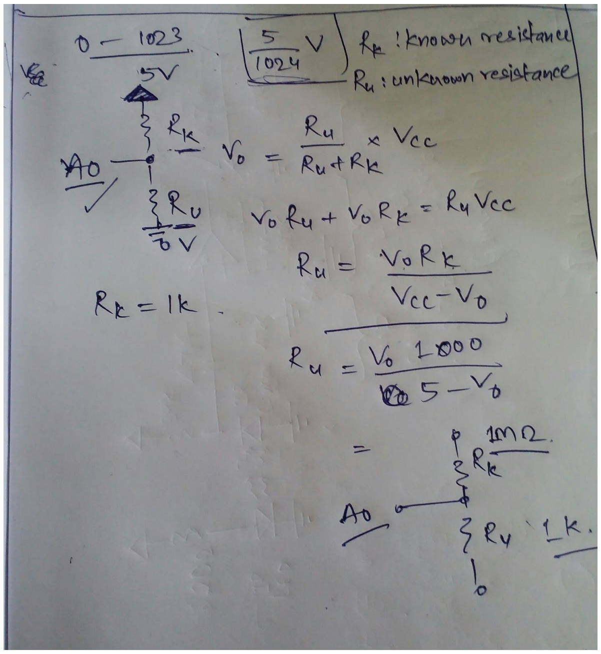
Image processing can be used to further extend the whole idea to monitor the eye-dilation, face-lines, abrupt movement of eyes, etc. These features have been associated to lying and can definitely help in polygraph test.

Links:

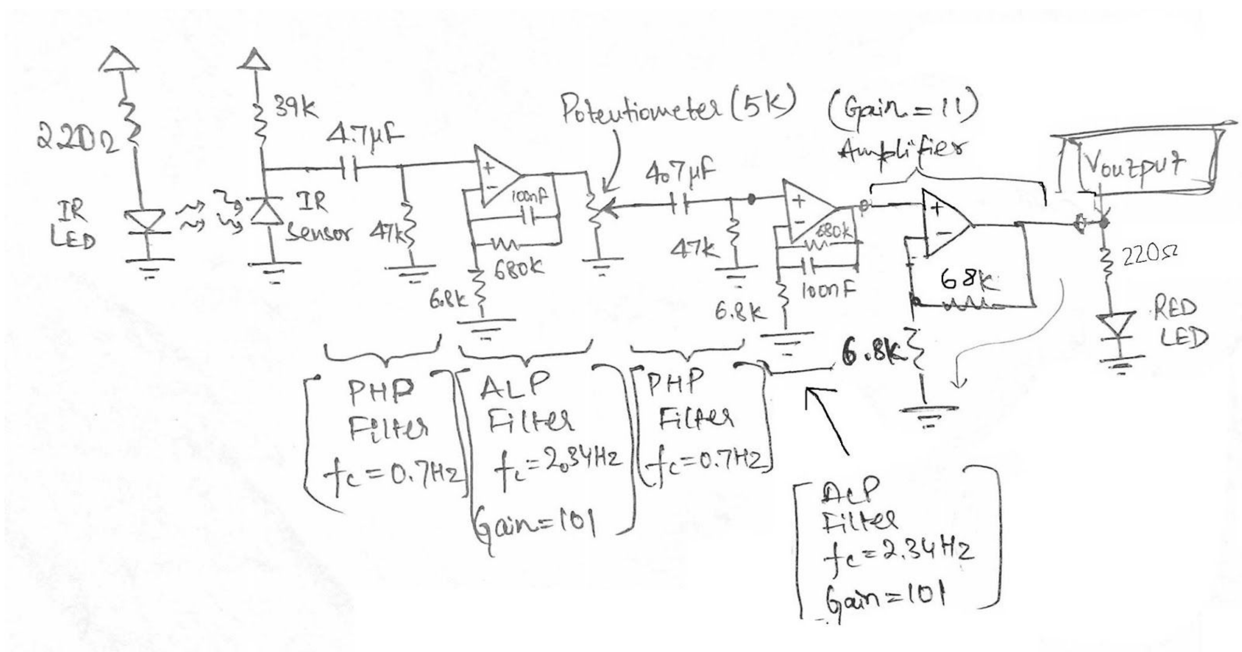
1. Electrodermal Activity: skin conductivity (https://en.wikipedia.org/wiki/Electrodermal_activity)
2. Electrodermal Activity: skin conductivity measurement meter (https://en.wikipedia.org/wiki/E-meter)
3. Instrumentation Amplifier (https://en.wikipedia.org/wiki/Instrumentation_amplifier)
4. Thin Film Resistors have low tolerance (https://en.wikipedia.org/wiki/Resistor#Thick_and_thin_film)
5. Digital Potentiometer (https://en.wikipedia.org/wiki/Digital_potentiometer)
6. Guide to amplifiers (http://www.ti.com/lit/an/slyt213/slyt213.pdf)
7. About pulse measurement and TCRT1000 sensor (https://hrsensor.wordpress.com/2008/04/28/sensor-report-reflective-optical-sensor-tcrt1000/) (http://embedded-lab.com/blog/introducing-easy-pulse-a-diy-photoplethysmographic-sensor-for-measuring-heart-rate/) (http://www.vishay.com/docs/83752/tcrt1000.pdf)
8. How sphygnomanmeter works (http://www.practicalclinicalskills.com/sphygmomanometer.aspx)
9. Blood pressure measurement using Arduino - very interesting code - (http://www.instructables.com/id/Automated-Blood-Pressure-Cuff-Arduino-Project/?ALLSTEPS)
10. Active Low Pass Filters: (http://www.electronics-tutorials.ws/filter/filter_5.html)
11. Great website for basic electronics: (http://www.electronics-tutorials.ws/)
12. Explained circuit for photoplethysmography: (http://embedded-lab.com/blog/introducing-easy-pulse-a-diy-photoplethysmographic-sensor-for-measuring-heart-rate/)
13. Interfacing arduino with python: (http://playground.arduino.cc/Interfacing/Python)
14. Measuring pressure Using Sphygnomanometer: (https://www.youtube.com/watch?v=Gmic13mvsgo)
15. Change in Blood Pressure with stress: (http://hyper.ahajournals.org/content/35/4/887.full)
16. Pressure Sensor: (http://e-radionica.com/productdata/Pressure_Sensor.pdf)
17. Why deployment of RxTx from NetBeans (or some other IDE) might not work and how I solved it? (http://stackoverflow.com/questions/17319254/java-deploying-rxtx)

Circuit Diagrams:

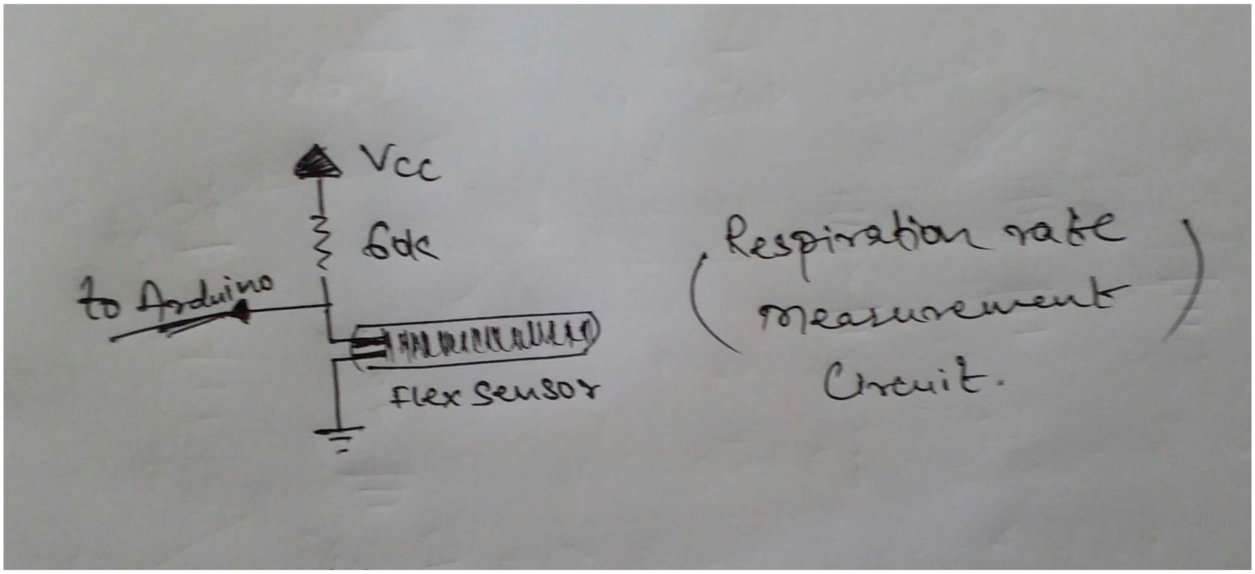
Skin Conductivity:



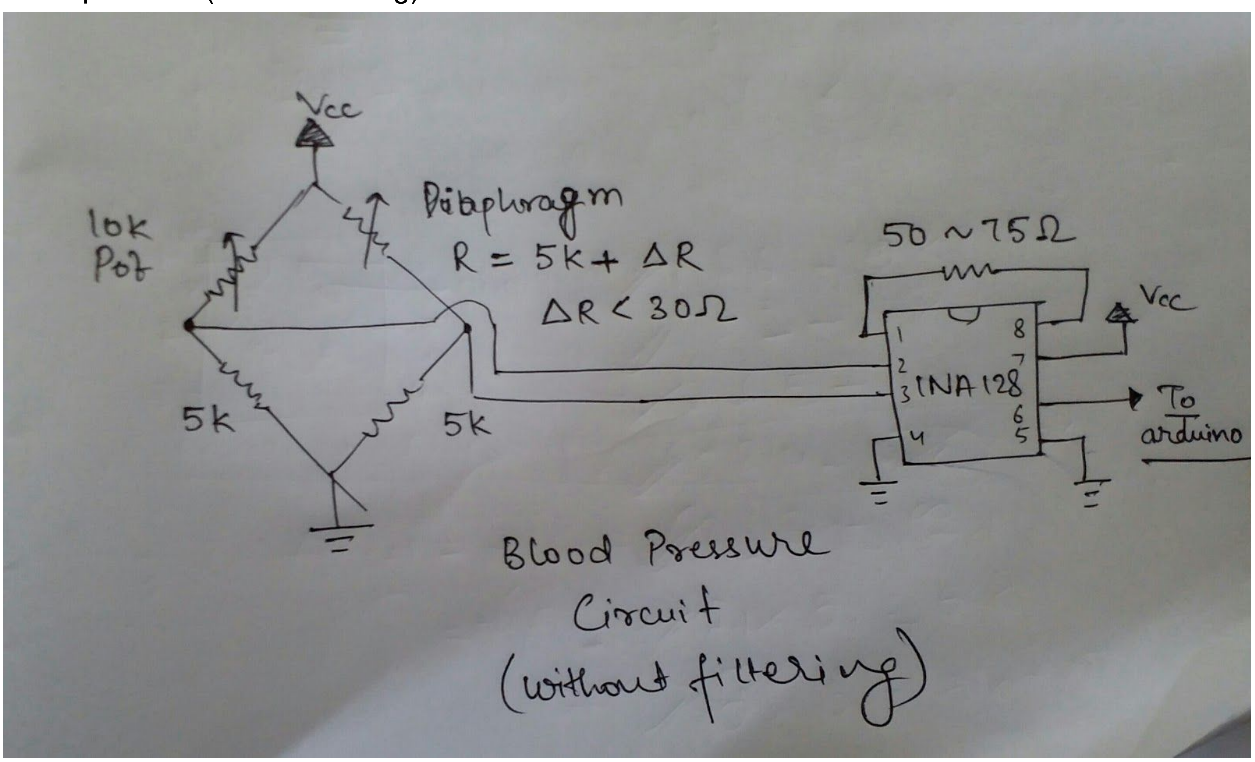
Pulse measurement:



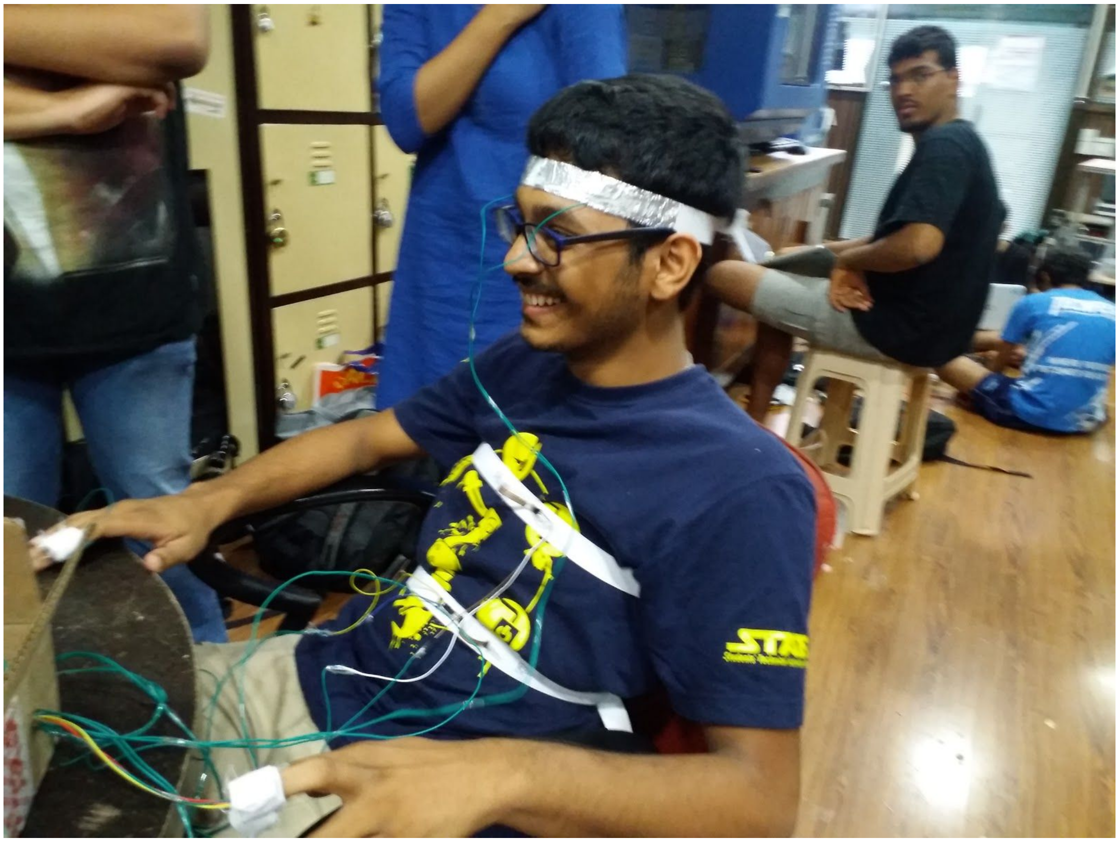
Respiration Rate:

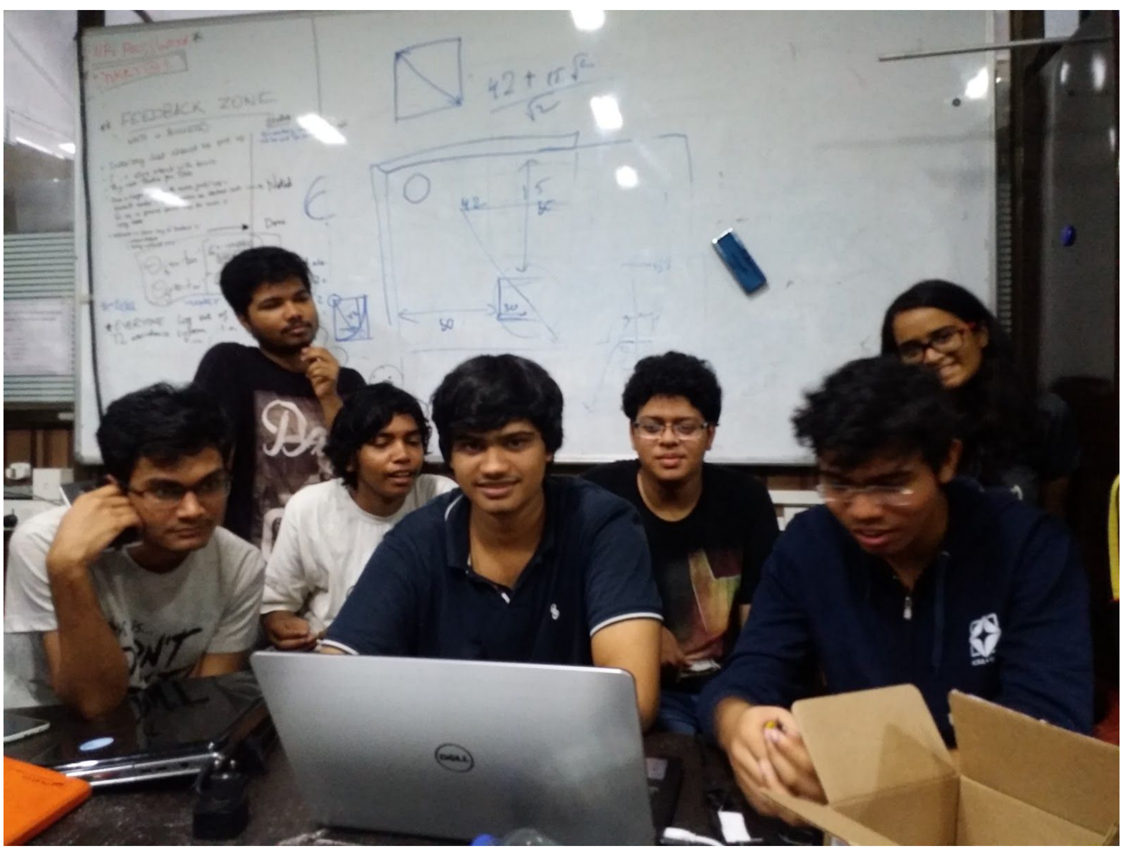


Blood pressure (without filtering):



Final Day Fun:




FEEDBACK ZONE


YOU ARE NOT ALLOWED TO TAKE ANY EQUIPMENT OUT OF THE LAB


extended