*Sudan University of Science and Technology*

*Faculty of engineering*

*Biomedical engineering*

# Design of Patient Monitor Using Android Application

**Prepared By:**

1. **AmnaAbdelmoneim Bashir**
2. **RayanOmer Ibrahim Alataya**
3. **Mohammed Ahmed Mohammed**

**Supervised By:**

**Dr.musaabAlkhair**

**(October,2017)**

# الآيـــــــــــة

{يَرْفَعْ اللَّهُ الَّذِينَ آمَنُوا مِنْكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ وَاللَّهُ بِمَا تَعْمَلُونَ خَبِيرٌ}

صدق الله العظيم
سورة المجادلة الآية {11}

## *Dedication*

*Every challenge work needs self-efforts as well as guidance of elders especially those who were very close to our heart.*

*Our humble effort I dedicate to our sweet and loving*

## *Families*

*Whose affection, love, encouragement and prays of day and night make able to get such success and honor,*

*Along with all hard working and respected*

## *Teachers*

## *Acknowledgement*

*We would like to express our sincere appreciation our principal supervisor, Dr. MusaabAlkhair, for his constant guidance and encouragement, without which this work would not have been possible. For his unwavering support, we are truly grateful. We are also grateful to all the lecturers in our department for their support towards the successful completion this project.*

*We also would like to express our heartfelt gratitude's to Eng. BayadirFath-elrahman and the electronics department in Sudan University of science and technology for supporting us during the entire data collection period, really grateful to them.*

*We would also like to thanks our families, colleagues at university and outside for supporting and encouragement.*

# TABLEOF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# ABSTRACT

In the last decade the healthcare monitoring systems have drawn considerable attentions of the researchers and students. The prime goal was to develop a reliable patient monitoring system so that the healthcare professionals can monitor their patients, who are either hospitalized or executing their normal daily life activities, the patient monitors are expensive, difficult to move from one place to another, not available at all healthcare centers and in rural areas. In this work we present a mobile phone device (android APP) based wireless healthcare monitoring system about physiological conditions (ECG, blood pressure, oxygen saturation, heart beats per minute and body temperature of a patient. Our proposed system is designed to measure and monitor important physiological data of a patient in order to accurately describe the status of her/his health, fitness and basic vital signs in easy procedures. In addition the proposed system is able to send alarming message about the patient's critical health data by text messages or by email reports to the doctors. By using the information contained in the text or e-mail message the healthcare professional can provide necessary medical advising. The system mainly consists of mobile phone, the data acquisition unit, microcontroller (i.e., Arduino UNO), and software (i.e., LabVIEW). The patient's temperature, heart beat rate, blood pressure, oxygen saturation, and ECG data are monitored, displayed, and stored by our system to be easy for recalling by user and reference doctor. To ensure reliability and accuracy the proposed system has been field tested. The test results show that our system is able to measure the patient's physiological data with reliable, acceptance values and inexpensive costs.

# المستخلص

في الاونه الاخيره أصبحت أنظمة مراقبة الرعايه الصحية محور إهتمام الطلاب والباحثين، وقد كان الهدف الاساسي هو تطوير وتصميم نظام موثوق به لمراقبة المرضي ، بحيث يمكن للمختصين في الرعايه الصحيه مراقبة مرضاهم حين تواجدهم بالمستشفي أو أثناء أداء مهامهم اليوميه خارج المستشفي.إضافة إلي ذلك نجد أن أجهزة المراقبه باهظة الثمن كما لا يمكن نقلها بسهوله من مكان الي آخر وايضا عدم تواجدها في المراكز الصحيه بالمناطق النائية .

في هذا المشروع نقوم بتصميم جهاز مراقبه صحيه في الهاتف المحمول عبر تطبيق اندرويد والذي يعتمد علي نقل البيانات بطريقه لاسلكيه ، وهذه البيانات يتم التحصل عليها باستخدام بعض المحسسات والاجهزه الالكترونية وذلك بمراقبة حالة المريض الصحيه وقياس بعض المعاملات المهمه جدا مثل درجة الحرارة ، ضغط الدم ، معدل ضربات القلب ، تركيز الاوكسجين وغيرها من المعاملات حتي يتمكن من وصف حاله المريض بكل دقه سهوله.

ومن ثم يمكن لهذا النظام إرسال رسالة نصيه أو تقرير متكامل عبر البريد الالكتروني للطبيب المختص وهذه الرساله تحتوي علي جميع القراءات المتحصل عليها والتي يستخدمها الطبيب لتوفير رعايه صحيه مثلي للافراد وقد تساعد في عمليات التشخيص والمعالجه لاحقا.

يتكون النظام من هاتف محمول و وحدة الحصول علي المعلومات التي تشمل العديد من المحسسات وجهاز اردوينو وغيرها من الاجهزه الاكترونيه.

درجة حرارة المريض ، ضغط الدم ، تركيز الاوكسجين ، معدل ضربات القلب وإشارة ضربات القلب يتم مراقبتها وعرضها وتخزينها بحيث تصبح سهلة المنال من قبل المستخدم أو الطبيب المختص ، ولضمان موثوقية هذا النظام قد تم إختباره ميدانيا وتحصلنا علي نتائج تظهر أن النظام قادر علي القياس بدقه ومراقبه حاله المريض الصحيه بكل سهوله ويسر.

# *Chapter One*

## Introduction

# INTRODUCTION

Patient monitor continuous measurement of patient parameters such as heart rate and rhythm, respiratory rate, blood pressure, blood-oxygen saturation, and many other parameters have become a common feature of the care of critically ill patients. When accurate and immediate decision-making is crucial for effective patient care, electronic monitors frequently are used to collect and display physiological data. Increasingly, such data are collected using non-invasive sensors from less seriously ill patients in a hospital's medical-surgical units, labor and delivery suites, nursing homes, or patients' own homes to detect unexpected life-threatening conditions or to record routine but required data efficiently.[1]

Modern bioinstrumentation, computers, and telecommunication technologies a modern PM should acquire, record, display, and transmit the physiological data from the patient body to a remote location at any time. For more efficient, timely, and emergency medical care the PMS must also be incorporated with an alarm system [1]. In order to alert the patient as well as the health care service providers the PM should not only monitor and analyze the critical patient's data but it should also send alarming messages in case the monitored data go outside their normal ranges.

Using the mobile phone healthcare system can be made available for people, who are living in remote areas without much access to other types of communications. Even a simple mobile phone can become a powerful healthcare tool now.

## 1.1 General Review

The patient monitoring systems is one of the major improvements because of its advanced technology to measure vital signs of patient (blood pressure, body temperature, heart rate per minute, ECG, and oxygen saturation). At the same time, the use of smartphones is quite generalized in the society, and it will increase meanwhile the prices of the technology is falling. Even if the smartphone has big computational capabilities, internet connection and several peripheral connection capabilities, it is not massively used in the personal medic field.

So we are here, just connecting the bio sensors( temperature sensor , heartbeat sensor, pulse rate sensor) so that simultaneously we can monitor the patient's condition and hence ruling out the use of the thermometer , pulse oximeter and other devices to check the condition of the patient. This project describes the design of a simple Arduino (UNO) based vital signs measures that could be transmitted to android mobile phone device.

The device alarms when the values of parameters exceed the provided threshold value. This threshold value is defined by the programmer at the time of programming the android.

The threshold value given for the project is as(80 to 120 mmhg)for blood pressure ,( 20 to 120 pulses per minute) for heart beat indication , ( 37°C) for temperature ,(80 pulse per minutes) for ECG and (100)for SPO2. This information i.e. the Heart Rate & the Body Temperature and saline level is then transmitted wirelessly to the doctor which in not in the vicinity of the patient.

## 1.2 Motivation

Traditionally, it was a custom to get these vital signs measured during a visit to the doctor, with advances in medicine and technology, this concept has adapted. There are many devices available in the market today that allow patients to monitor their own health on a regular basis from the comfort of their home. These devices are having a huge impact on health care costs as they are reducing the time and resources of medical physicians and facilities required by patients.

## 1.3Objectives

Design portable patient monitor, make it easily to use and available at inexpensive costs also ability to use it by any person to measure vital signs homely (that achieve the simplicity).

## 1.4Methodology

Design measure and monitor basic vital signs of body (ECG, blood pressure, heart rate, SPO2, body temperature), mechanism of programmed information by used Arduino (UNO) appendices(A), and android application to calculate, compare and display results of measured . Patient monitor system differ of our design in gain the data from PHYSIONET instead of pick up signals based on sensors from body.

Getting ideal ECG signal is almost difficult, the platform, power supply, GSM module and other components make much noise to signals, and so perfect way to avoid this is PHYSIONET. Ideal signal is saved in android application appendices (B). The problem in previous design is the negative values of ECG signal couldn’t determine by Arduino that make the percentage of error is

decreased, we fixed this error by edit the signals that obtained from PHYSIONET by MATLAB VEIW to be easy to insert in controller.

Available sensors like pulse rate sensor and temperature sensor LM35 are applied in platform to get values and measured from patient directly. Android APP represents signs known by ever one.

## 1.5Thesis structure

Chapter one is an introduction to the research, the related work is shown in chapter tow, while in chapter three the methodology used to conduct the project is explained, the results are mentioned, explained, and discussed in chapter four, In chapter five the conclusion and the recommendations are viewed in chapter six, chapter seven contains the references and appendices

!

# Chapter two

# Literature Review

# LITERATURE REVIEW

This project of Dhvani Parekh, deals with the signal conditioning and data acquisition of three vital signs: heart rate, blood pressure, and body temperature [2]. Heart rate is measured through an Electrocardiogram that is obtained by attaching skin surface electrodes on the patient's wrists and legs. Blood pressure combines the methodologies of Electrocardiography and Photoplethysmography to continuously monitor the systolic and diastolic blood pressure. Body temperature is measured inside the ear with a thermistor.

Deals with our design in vital signs, mechanism of programmed data. The results display in interface wave form, so make it difficult to determine by patients or user, this problem fixed by our design, the results displayed at numbers form to be easy to know by user.

Remote healthcare system for monitoring electrocardiographic and temperature data by Manjarres, has been presented in [3]. The system consists of three modules namely a hardware module, Bluetooth module and display module. The hardware module is used for data acquisition. The Bluetooth module is used for data transmission. Finally, the data are displayed by using the display module. The acquired clinical data are sent to a database server by using GPRS or Wi-Fi. The performances of the system have been tested on different patients and it has been found that the proposed system is very helpful for the physicians to read patient's vital signs, and help facilitate healthcare.

Differ in parameters used with our project, added other vital signs such as heart rate and oxygen saturation, also differ in hardware module and display and the way of transmitted data.

In this paper, Fernando Cornelio is presented for the remote monitoring of the body temperature and heart rate of a patient[4] by means of a wireless sensor network (WSN) and mobile augmented reality (MAR). The combination of a WSN and MAR provides a novel alternative to remotely measure body temperature and heart rate in real time during patient care. The system is composed of hardware such as Arduino microcontrollers (in the patient nodes), personal computers (for the nurse server), smartphones (for the mobile nurse monitor and the virtual patient file) and sensors (to measure body temperature and heart rate), a network layer using Wi-Fi technology and software such as LabVIEW. The results obtained from tests show that the system can perform effectively within a range of 20 m and requires ten minutes to stabilize the temperature sensor to detect hyperthermia, hypothermia or normal body temperature conditions. Additionally, the heart rate sensor can detect conditions of tachycardia and bradycardia.

The benefit of this paper used mobile phone to display the results of measured data neglected in our design the wireless sensors network (WSN) and personal computer.

The aim of this thesis [5] by Alex CorsBardolet, used sensors include breath rate sensor and a commercial thermistor used for human temperature measures. All this sensors will be connected through a device that handles the power and communication. After that, an Android application has been done to control this device and show the results of the measures. The values of the measures are sent to a remote server in order to store information. The results of a research of the actual state of art are presented.But didn’t mention other vital signs for body.

Amna Abdullah, presents a smartphone based wireless healthcare monitoring system (WHMS) [6], which can provide real time online information about medical status of a patient. In addition alarming and reminding messages about the patient health status can also be sent to patient mentors for necessary medical diagnosis and advising. The proposed system consists of sensors, a data acquisition unit, smartphone, and the LabVIEW program. The system is able to display, record, and send patient's physiological data. The patient is equipped with biomedical sensors, which transform the changes in the monitored physiological quantities into electronic data that are measured and recorded. The LabVIEW program assists monitoring and displaying the data. The patient's temperature, heart beat rate, muscles, blood pressure, blood glucose level, and ECG data can be monitored by present system. Their careful design of the hardware and software components of the system is able to fulfil any further requirement of the users. The results of this project are accurate enough to display.

We excerpt most data from this project (smart phone, microcontroller, parameters and LabVIEW program) except the sensors and ability to display in PC. The great benefits of this paper to collect most of data, tools, software programs, the way of display results and alarming. Also its result is more accurate and clear.

Aim of work by Tamil Nadu is to monitor the human body temperature, blood pressure (BP), Pulse Rate, GSR, Glucose, Body position, ECG. The human body temperature, BP, Pulse Rate and ECG are detected in the working [7] environment. This can be sensed by using respective sensors. The sensed information is send to the microcontroller through signal conditioning circuit. The sensor information will be transmitted from the patient unit to the main

controller unit with the help of ZigBee communication system which is connected with the microcontrollers. The main controller unit will send those sensed data as well as the location of that patient by the help of GPS Module to the observer/doctor. The results of project message is sent to a mobile phone using Global system mobile (GSM) Modem.

Measurement is almost not accurate because there are percentages of errors affected by sensors, circuits and missing some of data when transfer. Fix that by design our system and enhance the results of measured.

At paper of Maradugu Anil Kumar [8], critical cases are supposed to be monitored continuously SP02, Heart Rate as well as temperature. In the earlier methods, the doctors need to be present physically or in several cases SMS will be sent using GSM. In the earlier case the history of the patient cannot be displayed, only current data is displayed. In the current paper, we are using a novel idea for continuous monitoring patient's health conditions. The health care scheme is focus on the measurement and Monitoring various biological parameters of patient's body like heart rate, oxygen saturation level in blood and temperature using a web server and android application, where doctor can continuously monitor the patient's condition on his smart phone using an Android application. And also the patient history will be stored on the web server and doctor can access the information whenever needed from anywhere and need not physically present.

Agree with our project in most parameters differ in procedure, and way to transmit data. We used Bluetooth module to transmit data to mobile phone instead of using GSM.

This paper deals with design and developed for remote patient monitoring in healthcare field[9]. The primary function of this system is to constantly monitor patient's physiological parameters such as pulse rate, breathing rate, blood pressure rate and patient's body movement, and display the same information to the doctor. In hospitals, where patient's physiological parameters are needed to be constantly monitored, is usually done by a doctor or other paramedical staff for maintaining a record of it. It is a tedious method. In this proposed system transmitting module continuously reads patient's pulse rate or heart beat rate, breathing rate, patient body movement and blood pressure rate through a pulse sensor, airflow sensor, accelerometer and sphygmomanometer .The sensors are used sense the information and Microcontroller is used to receive the data from the sensor. Using a ZigBee transmitter the retrieval information be send to corresponding receiver locate at the computer. The ZigBee receiver receives the data from transmitter and finally displays record to doctors via web browser. Provide notification message about the emergency to the physicians through GSM module. The main aim is to reduce the cost for monitoring patient health using different types of sensor.

In this project the sensor nodes can wirelessly communicate with any smart phone through an Android application to continuously monitor and have complete access to the medical data of the patient [10]. Moreover it also aims to maintain an efficient electronic medical record of the person. Moreover, the consultant and the caretaker of the patient can have this important information remotely through an internet connection and provide with significant advice which encapsulates the term -smart first aid technology. In the proposed framework miniaturized sensors are worn on the body and non-intrusively

monitor a person's physiological state. The body vital signs (e.g.: heart rate, temperature etc.) are recorded through the sensor nodes and transmit to the smart phone via Bluetooth, where the data of vital signs is stored and will further transmitted to remote locations if needed.

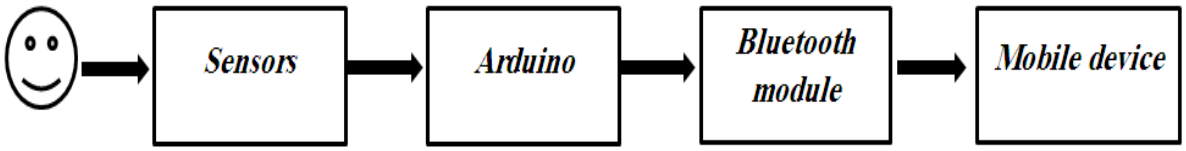
# Chapter Three

# Methodology

# METHODOLOGY

## 3.1 Over view

proposed design is measured, monitor the basic vital signs of body ECG, blood pressure, heart rate, SPO2 and body temperature, monitor the health and fitness, we found that most of biosensors are not available in our region like ECG sensor, blood pressure sensor and SPO2 sensor so we used PHISIONET to pick up signals and values of parameters mentioned above, on other hand the available sensors are pulse rate sensor and temperature sensor applied them in platform to pick up values. To implement this design the component have been available in circuit are pulse rate sensor, temperature sensor (LM35), amplifiers, filters, Arduino (UNO), Bluetooth module and mobile device (android app).




Block diagram (1) for system design

The block diagram above is represent all the components are used and the way of connection them. The first sensor used in the design is pulse rate sensor which is put in the patient's finger used to have a signal with special properties, the output of these circuits are connected to Arduino UNO in port A0. The second sensor is LM35 that used to measure temperature from patient's finger also connected with Arduino in port A1. The outputs of circuits are analog signal and Arduino mention in Appendices (A) is dealing with analog signals. The output signals are very small and have a noise so need amplifier and filter to remove noise. Signals of ECG, SPO2 and blood pressure are obtained from

PHISIONET, because the sensors of vital signs mentioned above are expensive and not available. The signals have negative peaks so, need to modulate because the Arduino never deals with negative values. PHISIONET is accurate web site contains real signals of patients, established by researchers, doctors, international hospitals and healthcare institutions.

## 3.2 Parameters

### 3.2.1Heart rate

Heart has a very important role to play in pumping your blood around the body, through a cycle of continuously and regularly contracting then relaxing. We can measure the rate of thiscontraction of the heart directly using pulse rate sensor placed on finger. The rate of the heart contractions can vary greatly, and is affected by many factors, including your state of excitement, exercise, disease and medications.[2]

The normal resulting heart rate is typically listed as between 60-80 beats per minute. In many cases there are easily explained reasons for resting heart rate to be outside of typical range. An abnormal heart rate is when your heart beats too fast, slow, or irregularly. This is also called an arrhythmia mentioned in table (3.1)

Within the heart is a complex system of valves, nodes, and chambers that control how and when the blood is pumped. If the functions of this vital system are disrupted, damaged, or compromised, it can change the pattern with which your heart beats. Arrhythmias can cause no symptoms, or you may feel discomfort, fluttering, pain, or pounding in your chest. Not all arrhythmias are life-threatening or cause health complications. To be on the safe side, though, any abnormal heart rhythm should be reported to your doctor. [2]

Table3.1: heart rate.

| Rate | Beat/min |
|---|---|
| Normal | **120-160** |
| Tachycardia | **>160** |
| Bradycardia | **<120** |

## 3.2.2 Body temperature

Temperature is a measure of the degree of heat intensity. The temperature of a body is an expression of its molecular excitation. The temperature difference between two points indicates a potential for heat to move from the warmer to the colder point. The human body's core temperature varies from day to day, and from time to time, but these fluctuations are small, usually no more than 1.0ºC. Humans are homoeothermic and body temperature is regulated at about 37ºC ±1ºC. The thermoregulatory center in the hypothalamus plays a very active role in keeping body temperature in the normal range. External and internal heat sources influence body temperature [11].

## 3.2.3 ECG

Including the shape of the P wave (atrial depolarization), the amplitude of the QRS complex fig.3.1(ventricular depolarization), and the shape of the T wave (ventricular recovery), are reproduced faithfully.

When the sampling rate is decreased to 100 measurements per second, however, the amplitude and shape of the QRS complex begin to be distorted. When only 50 observations per second are recorded, the QRS complex is grossly distorted, and the other features also begin to distort. At a recording rate of only 25

Measurements per second, gross signal distortion occurs, and even estimating heart rate by measuring intervals from R to R is problematic.


Fig.3.1: ECG signal.

The sensors are used to measure the ECG signal called electrodes, ECG electrode is a device attached to the skin on certain parts of a patient's body — generally the arms, legs, and chest — during an electrocardiogram procedure. It detects electrical impulse produced each time the heart beats. The number and placement of electrodes on the body can vary, but the function remains the same. The electricity that an electrode detects is transmitted via this wire to a machine, which translates the electricity into wavy lines recorded on a piece of paper. The ECG records, in a great detail, are used to diagnose a very broad range of heart conditions. An ECG electrode is usually composed of a small metal plate surrounded by an adhesive pad, which is coated with a conducting gel that transmits the electrical signal. These electrodes are available but have highly costs, so we used PHYSIONET to obtain the accurate, actual, precise, filtered signal without noises that come from power supply, platform and other components. The obtain signal is edited by software view (matlab) in fig 3.2.

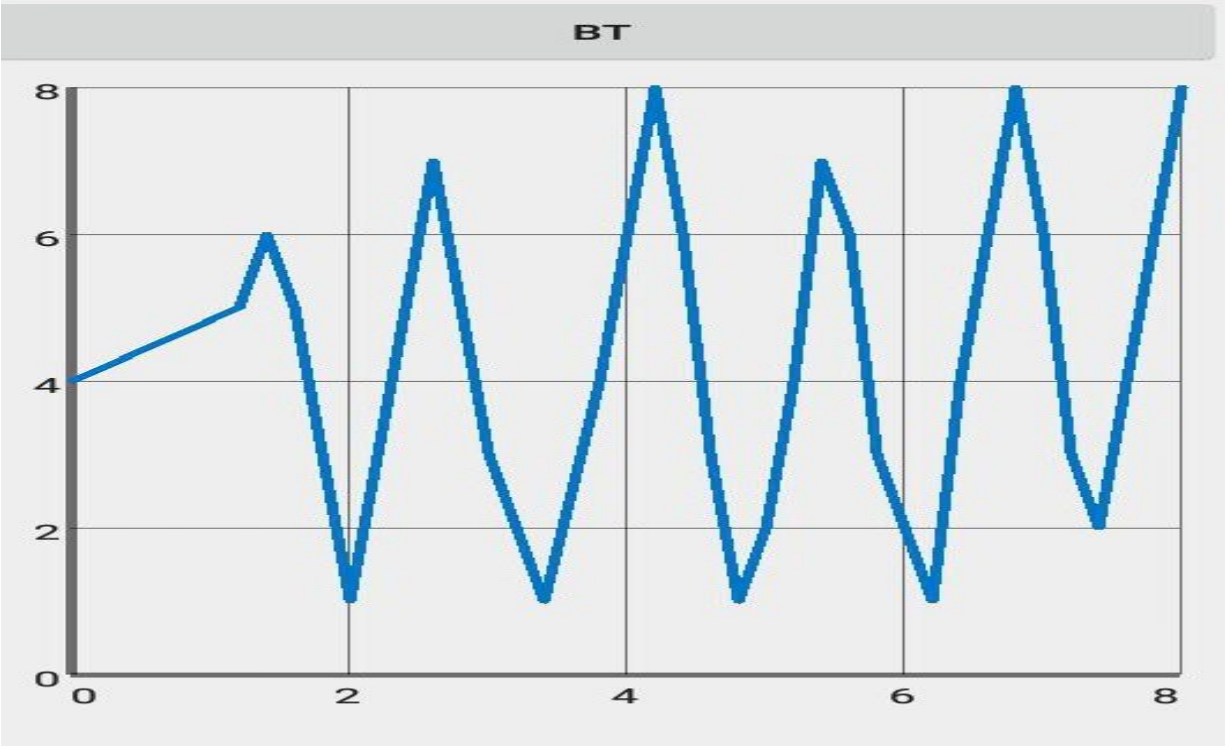



Fig.3.2: Edited signal.

## 3.2.4 SPO2

The heart pumps blood to the lungs where hemoglobin in the red blood cells is oxygenated. This oxygenated blood is then pumped to various parts of the body via the arteries. Arterial blood gas saturation (SpO2) is the percentage of hemoglobin in the arteries that is oxygenated. [13]

Person is getting enough oxygen. Arterial oxygen saturation of healthy people will be in the range 94%-99%. Anything below 90% indicates insouciant supply of oxygen. Low SpO2 can be due to problems in the lungs causing insouciant perfusion of oxygen to the blood or in the blood cells themselves as in the case of anemia. In newborns, low SpO2 is indicative of Septicemia (infection) congenital heart defects, anemia, respiratory distress syndrome etc.

"Is a reading that indicates the percentage of hemoglobin molecules in the arterial blood which are saturated with oxygen? The reading may be referred to as Sao2. Reading varies from 0 to 100% mention in table (3.2). Normal readings in a healthy adult, however, range from 94% to 100%. SPO2 sensor is not available so we also used PHYSIONET to pick up data; data saved in memory of Arduino to represent by APP.

Table3.2: saturation of SPO2%.

| Range | State | Priority |
|---|---|---|
| **<64** | Error level | 5 |
| **65-79** | High risk level | 4 |
| **80-91** | risk level | 3 |
| **92-94** | Deviant level | 2 |
| **95-100** | Normal level | 1 |
| **>100** | Error level | 5 |

## 3.2.5Blood pressure

Is the force exerted by blood against the walls of the arteries. Systolic pressure occurs when the heart contracts; diastolic pressure occurs when the heart expands. Blood pressure is measured in millimeters of mercury (mmHg) look fig.3.3. Blood pressure is affected by many factors: age, weight, time of day, activity level, climate, altitude and season. Certain activities can significantly alter one's blood pressure. Walking can raise systolic pressure by 12 mmHg and diastolic pressure by 5.5 mmHg. Sleeping can decrease systolic blood pressure by as much as 10 mmHg. Taking your blood pressure repeatedly

without waiting an interval of 5 minutes between readings, or without raising your arm to allow blood to flow back to the heart, can also affect it. Normal less than 120 and less than 80Prehypertension 120-139 or 80-89, Stage 1 Hypertension 140-159 or 90-99Stage 2 Hypertension greater than or equal to 160 or greater than or equal to 100 all this mention in table (3.3).

Table3.3: blood pressure measurement.

| Category | systolic | | Diastolic |
|---|---|---|---|
| Optimal | <120 | And | **<80** |
| Normal | 120-129 | And/or | **80-84** |
| High normal | 130-139 | And/or | **85-89** |
| Grade 1 hypertension | 140-159 | And/or | **90-99** |
| Grade 2 hypertension | 160-179 | And/or | **100-109** |
| Grade 3 hypertension | >=180 | And/or | **>=110** |
| Isolated systolic hypertension | >=140 | And | **<90** |

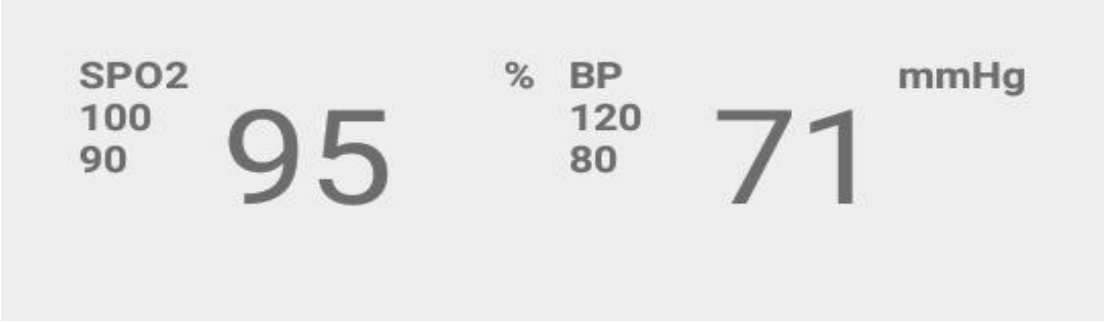



Fig.3.3: Blood pressure measurement.

# 3.3Component

## 3.3.1 Pulse rate sensor

Is a sensor used to measure the heart beats per minute in fig.3.1, the normal beats 80-120 bpm. The sensor is a plug-and-play for arduino.it essentially combines a simple optical heart rate sensor with amplification, noise cancellation and light emitting diode LED. Simply clip it to your finger in 5v Arduino.

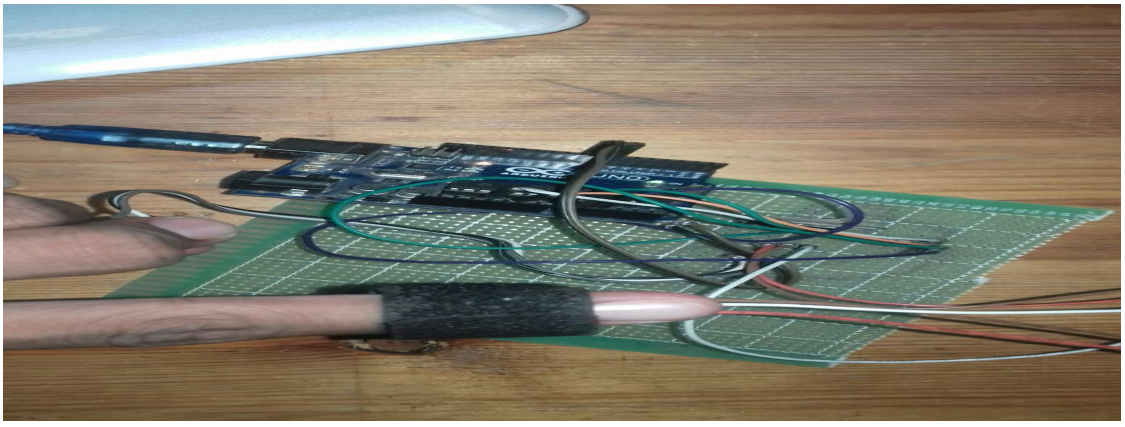


Fig.3.4: Pulse sensor.

### 3.3.2 The LM35 temperature Sensor

Body temperature can be measured by a lot of sensors such as lm35 fig.3.2, DS60 and thermostat. The LM35 can be applied easily in the same way as other integrated-circuit temperature sensors.TheLM35 is linear, analog, can measure temperature from -55 to +155, sensor does not require any external calibration to provide typical accuracies values. The LM35's low output impedance, linear output, precise and easy to use. So we used it to implementation the project.

Connect LM35 to Arduino Uno as shown in circuit diagram. The +5v for LM35 can be taken from vcc +5v out pin of Arduino. Also the ground pin of LM35 can be connected to GND pin of Arduino. Connect Vout to p1 pin of Arduino.

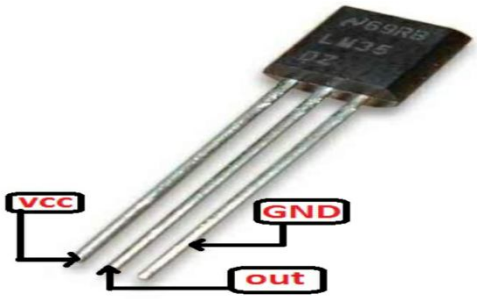



Fig.3.5: LM35 sensor.

### 3.3.3Arduino Uno

The Arduino Uno is microcontroller board based on the AT mega 328 look fig.3.6. It has 14 digital input/output pins, 6 analog inputs, a USB connection, a power jack and 16MHZ ceramicresonator. Simply connect to computer with a USB cable with AC-to-DC adapter or battery to get started. Programmed by code in appendices (A), it mentioned in chapter six.

Pin0 of Arduino UNO is connected by the output of the pulse rate sensor; pin1 is connected by output of IM35 temperature sensor. Arduinocomparable with microcontrollers is more accurate than it.

Signals from PHYSIONET are saved in the memory of Arduino to be easy to recall and represent in android APP during measurement. PHYSIONET is a free web access to large collections of recorded physiologic signals. They are available in digital forms in large amount of datasheet, so not difficult to convert them to graph such as ECG or actual values like SPO2 and blood pressure. All data picked up from sensors and obtained from PHYSIONETis saved in the memory of Arduino then transmitted these data through Wi-Fi module to be received by android APP that can represent the status of all measured data.


Fig.3.6: Arduino.

### 3.3.4 Bluetooth module HC-05

HC05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, in fig.3.7 designed for transparent wireless serial connection setup. Serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Blue core 04 External single chip Bluetooth systems with CMOS technology and with AFH (Adaptive Frequency Hopping Feature). It has the footprint as small as 12.7mmx27mm. Hope it will simplify your overall design/development cycle.

**Hardware features**

Typical 80dBm sensitivity, Up to +4dBm RF transmits power, low Power 1.8V Operation, 3.3 to 5 V I/O, PIO control and with integrated antenna and with edge connector.

**Software features**

Slave default Baud rate: 9600, Data bits:8, Stop bit:1,Parity:No parity, PIO9 and PIO8 can be connected to red and blue led separately, When master and slave are paired, red and blue led blinks 1time/2s in interval, while disconnected only blue led blinks 2times/s, Auto connect to the last device on power as default, Permit pairing device to connect as default, Auto pairing PINCODE:"1234" as default and Auto reconnect in 30 min when disconnected as a result of beyond the range of connection.

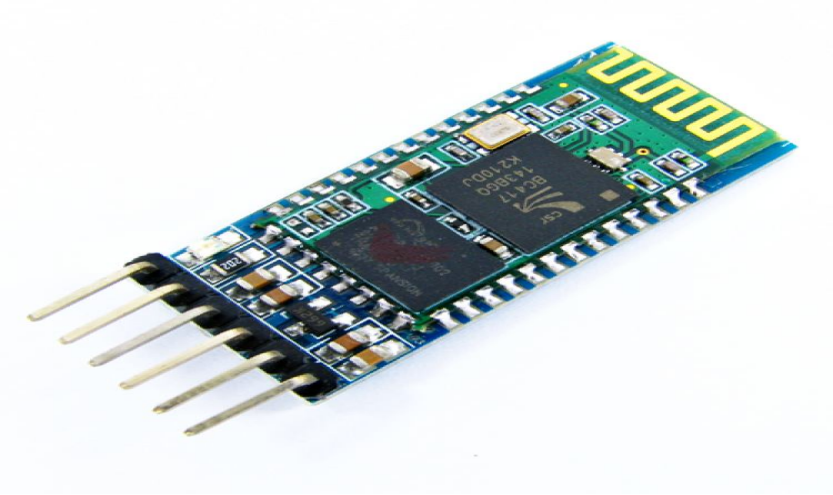


Fig.3.7: hc05 module.

## 3.4Physio Net

Is a research resource intended to stimulate current research and new investigations in the study of complex biomedical and physiologic signals [12]. It has three major components: Physio Bank is a large and growing archive of well characterized digital recordings of physiologic signals, time series, and related data for use by the biomedical research community. Physio Bank currently includes more than 50 collections of cardiopulmonary, neural, and other biomedical signals from healthy subjects and patients with a variety of conditions with major public health implications, including sudden cardiac death, congestive heart failure, epilepsy, gait disorders, sleep apnea, and aging. These collections include data from a wide range of studies, as developed and contributed by members of the research community. Physio Toolkit is a large and growing library of software for physiologic signal processing and analysis, detection of physiologically significant events using both classical techniques and novel methods based on statistical physics and nonlinear dynamics, interactive display and characterization of signals, creation of new databases, simulation of physiologic and other signals, quantitative evaluation and

comparison of analysis methods, and analysis of no equilibrium and no stationary processes. A unifying theme of many of the research projects that contribute software to Physio Toolkit is the extraction of “hidden” information from biomedical signals, information that may have diagnostic or prognostic value in medicine, or explanatory or predictive power in basic research. All Physio Toolkit software is available in source form under the GNU General Public License (GPL). Physio Networks is a virtual laboratory for development of data and software resources that will eventually become components of Physio Bank and Physio Toolkit. By providing large, secure workspaces with redundant backup to active researchers who can easily share them with colleagues anywhere, Physio Networks encourages investigators to create well-organized and documented, usable data and software repositories during the conduct of their research. When the research is complete and the major results have been published (or at any time the researcher wishes) the repository can be shared with a colleague, a group of colleagues, or the research community at large.

## 3.5 Android application

All data have been picked up from sensors and obtained from PHYSIONET which are programmed by Arduino Uno transfer from it through Bluetooth module are received by android Application.

APP fig 3.8 can represent all data in familiar and easy form to be readable by ever one. Smart mobile phone recently is most popular between all levels of ages. Easy to share the APP to arrive all people in all ages. The APP saved ideal values of measured vital signs range of 120-70 for normal pressure,

range of 90-100% for oxygen saturation, range of 60-100 for heart beat per minute and 37 degree centigrade for body temperature.

The project targeted rural areas and seniors most of them found difficulty to use and know measurements, so we used simplest form to represent data. Android APP could be used by several people or patients. It is saved privacy for users, has an internal folder which has many files related with users. If someone creates a new account, his/her file was also created automatically and saved all measured values with actual date and time for a period of time to be easy to recall and revision by a doctor. If you want to establish new account, first enter your details look fig.3.9 in APP and register it, also go to welcome page, rewrite in fig.3.10 the details to go to measurements page, then other measurement page obtained from sensors. You can captured the pages and send them to doctor.


Fig.3.8: application form.

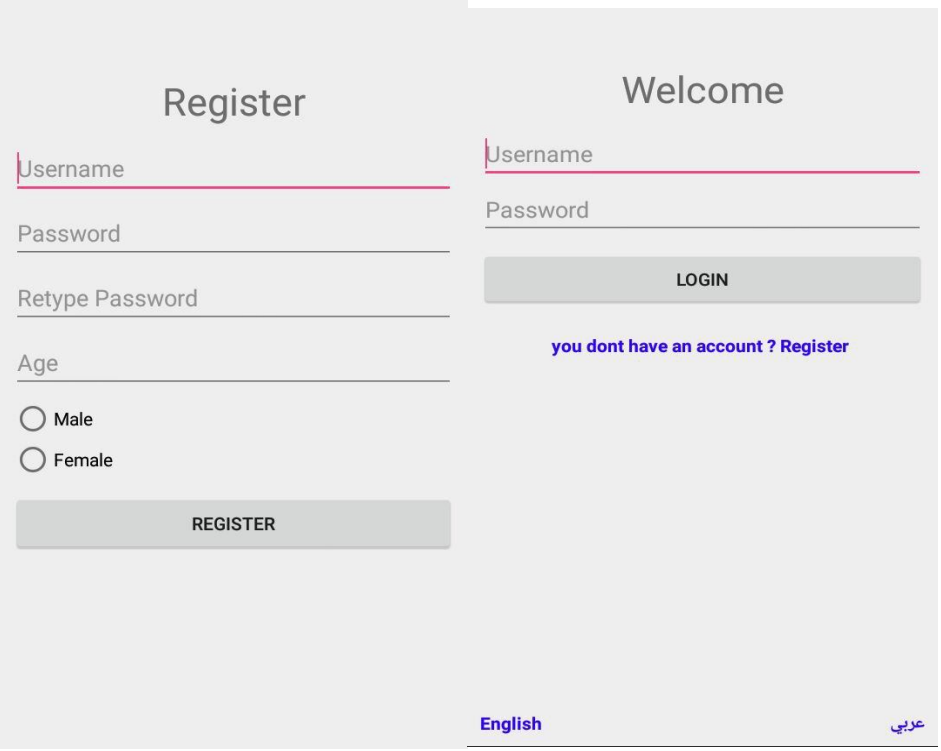
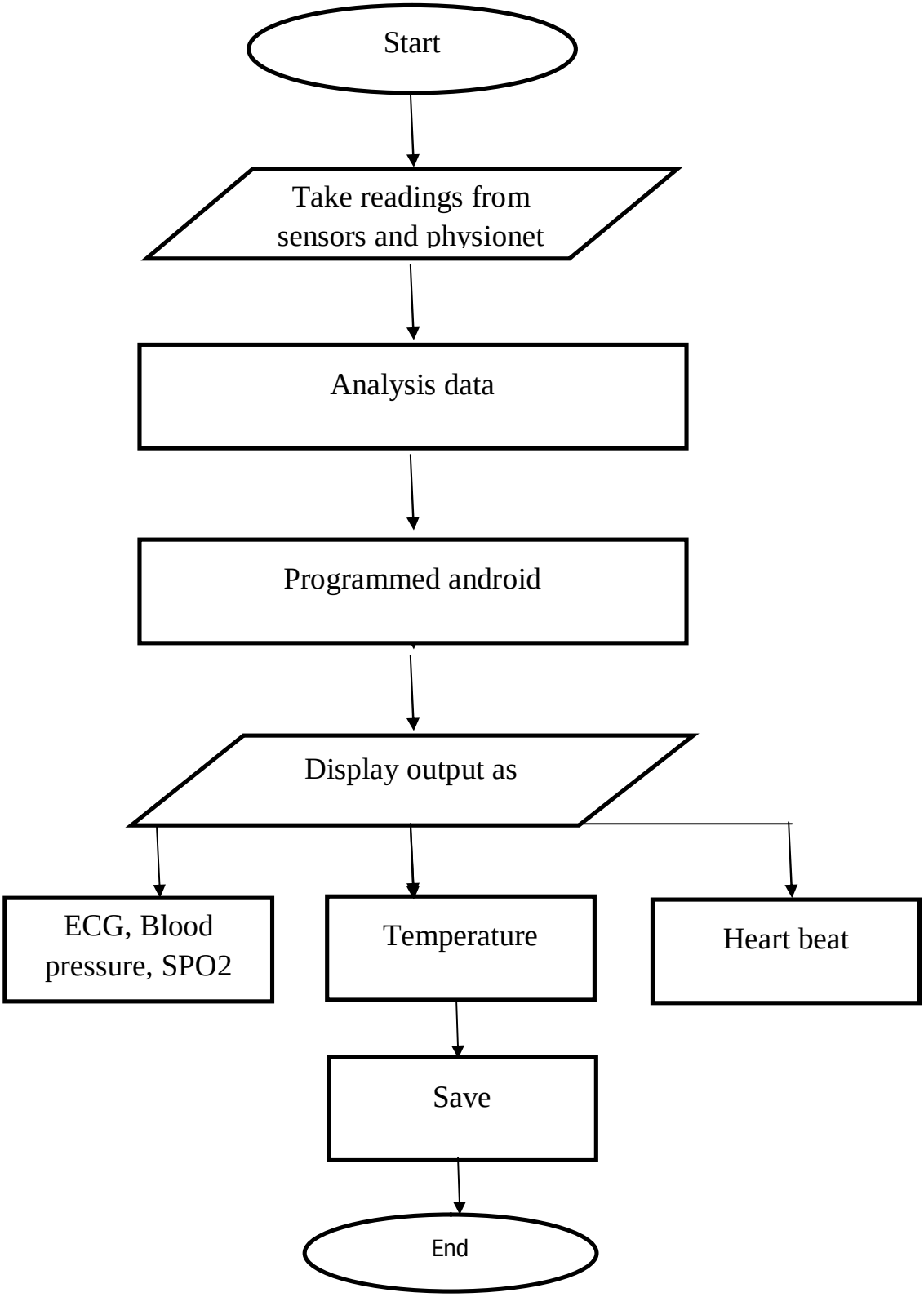



Fig.3.9Fig.3.10

Mobile device accepted data from Bluetooth module, then transformed data to language which could be known by android to represent result in similar way of normal patient monitor display, APP take small space in memory of mobile device and can simply open and send the APP source to other mobile phone device so, easily to gain smart patient monitor in your region in your home in your phone. Android APP designed by three complex codes first one for the face page, second for main body of representation results and the third code for measured data.

The program flowchart shown next representsthe steps of the program for the system. The program starts by receiving the readings from the sensors connected to the patient's body through wires and PHISIONET. The acquired data is then sent to the programming environment (i.e., LabVIEW Software).

The program analyzes and displays the data regarding the ECG, blood pressure, SPO2, heart rate and body temperature. Finally, the data are saved and also used to generate well-organized measurements and monitoring by the system with respect to the time.




**Flow chart system program**

## 3.6Circuit Diagram and Explanation

Connect the Pulse Sensor with the Arduino fig 3.11. The connections of the pulse sensor are very easy. Pulse sensor has three pins. Connect 5V and the ground pin of the pulse sensor to the 5V and the ground of the Arduino and the signal pin to the A0 of Arduino. You do not have to connect a resistor with because the Arduino has built in resistor at pin 13.

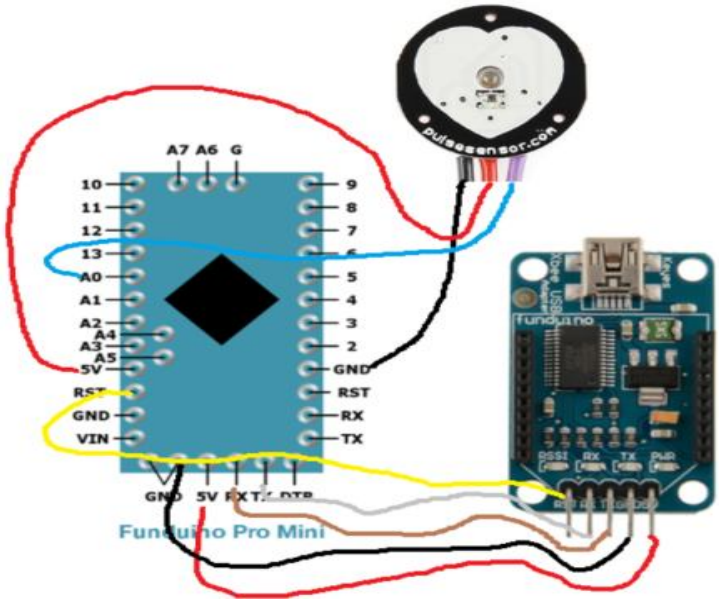



Fig.3.11Pulse sensor Connections

The red color is voltage, black is ground and blue is output of pulse sensor. Yellow wire is reset, black is ground, red is voltage, brown is receiving, gray is transferring between Arduino and board. Thisconnection installed in board and then wounded.

After that connect the LM35 sensor with Arduino. Connection also is very easy, we mentioned above that LM35 sensor has three pins, connected Vcc with +5v appear in wire with red color, GND of sensor with ground of

Arduinoin wire with blue color and output with pin A1 of Arduino in wire with green color look fig 3.12.

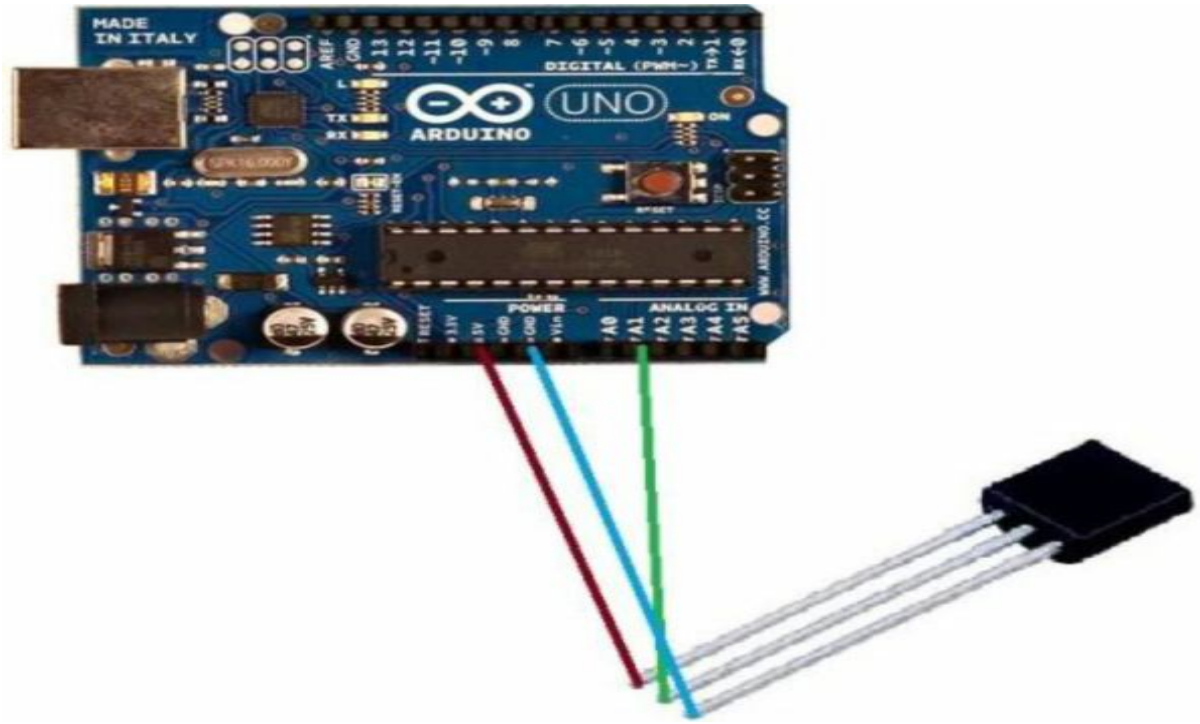



Fig.3.12:LM35 Connections.

Finally, Connect the Bluetooth module with the Arduino, HC-05 is a serial port module which makes it very easy to use. If you see the pin configuration of HC-05, there are total 6 but we only need 4 middle ones for our set-upconnect VCC with 5V of Arduino, please do not connect it with 5V as that can cook the module, connect GND with any GND of Arduino and connect Rx pin with RX of Arduino, connect TX pin with TX of Arduino. Green is RX, yellow is TX, black is ground, Red is vcc shown in fig 3.13.

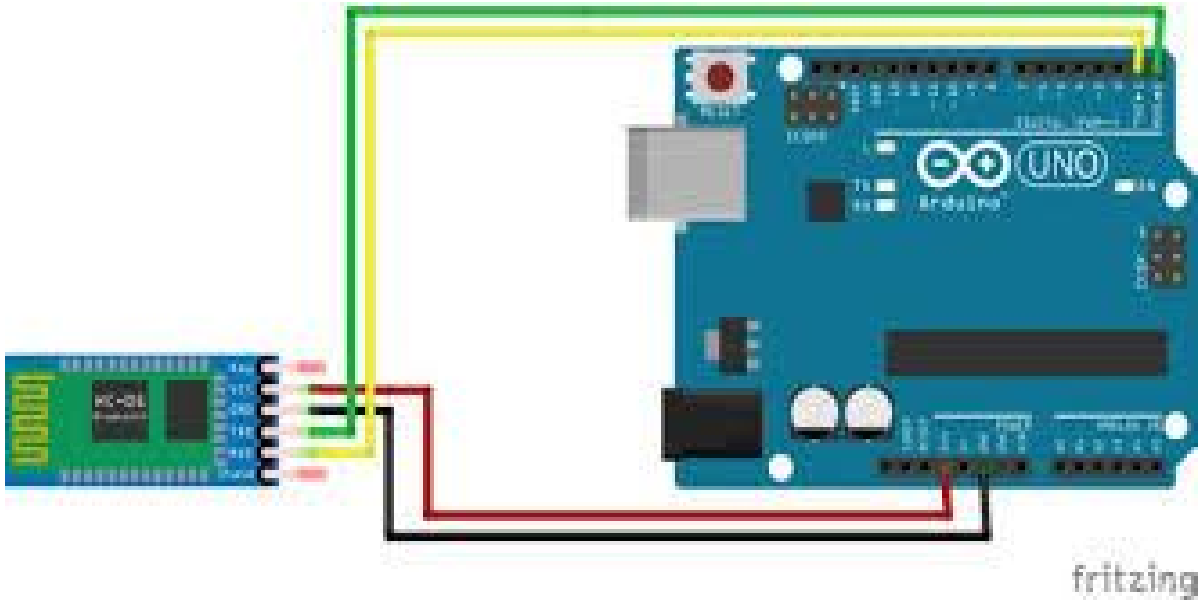



Fig.3.13: hc05 moduleconnection.

# Chapter Four

# Results

# 4.1RESULTS

Here developed an android application for receiving the medical parameters and displayed on android mobile with the help of Bluetooth Module and at a time uploaded on to the android web server. After opening the android app in mobile it shows the list of Bluetooth modules shows in Figure then connected the required Bluetooth module that is connected with the system hardware. Then took readings for temperature sensor and pulse rate sensor.Through various test procedures and techniques, many parts of this project were improved. Initially, basic features were tested to ensure that each component or block worked and then as testing progressed, modifications or adjustments were made to the circuits so they functioned well practically.

## 4.1.1Heart Rate

Circuit of heart rate was built in the board, the pulse rate sensor tested initially, the readings is acceptable in the range of heart beats per minute and readings in and testing was performed on each configuration, in relaxed and standing, observed in the next photo taken from application in mobile device.

## 4.1.2Body temperature

The LM35 was initially tested to ensure that it performed according to specifications.

A medical LM35 currently available in the market was used to compare the results with the temperature values. The initially readings not acceptable because out of range and was not performed its normal function so exchange LM35 sensor. Then tested five times and observed readings were differing completely from Previous and in range, shown in figure:

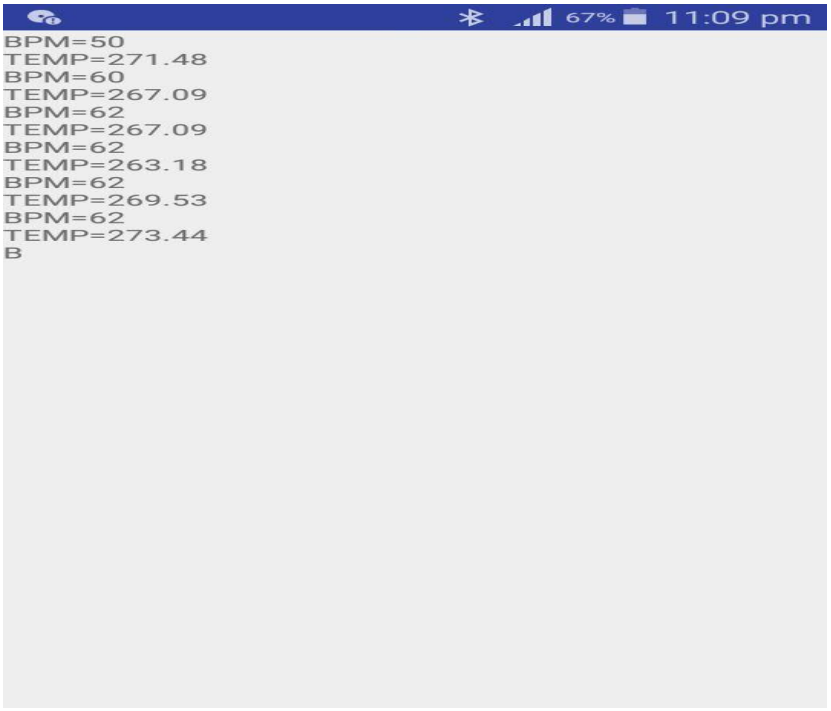



Fig.4.1

Acceptable readings for both heart rate and body temperature:

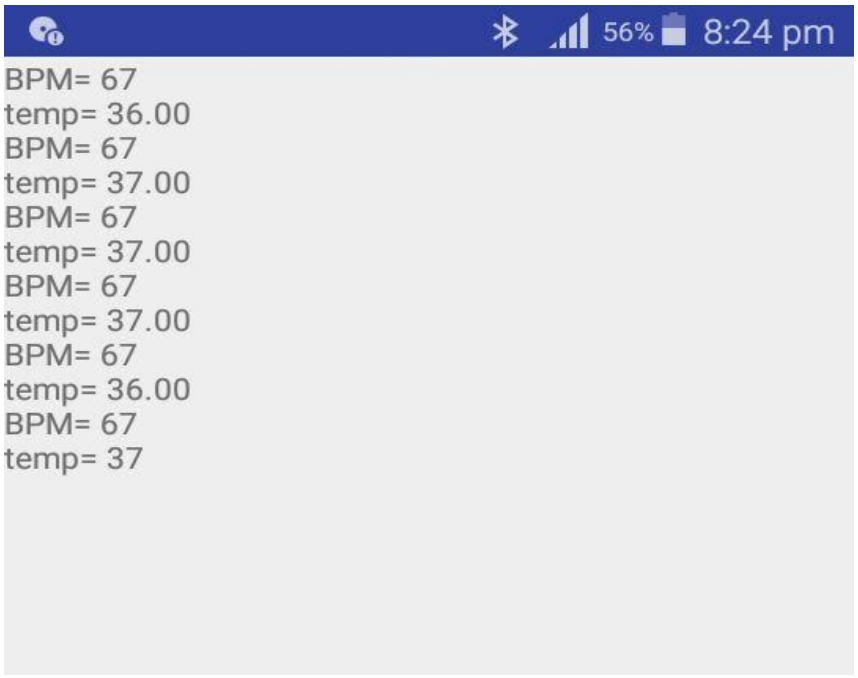



Fig.4.2

The signals of ECG, blood pressure oxygen saturation had obtained from PHYSIONET which saved in memory of Arduino to be displayed by the application in smart phone is:

- The signal of ECG is represented in image of patient monitor it is abnormal, that's refer tosavefile of patient that take information from it in PHYSINET.
- The saved data of blood pressure was 71 mmhg in normal range.
- The saved signal of SPO2 was 95% in normal range.

While the in the case of parameters within the normal range there is no alarm for patient that allow to understand, patient is in safe side.

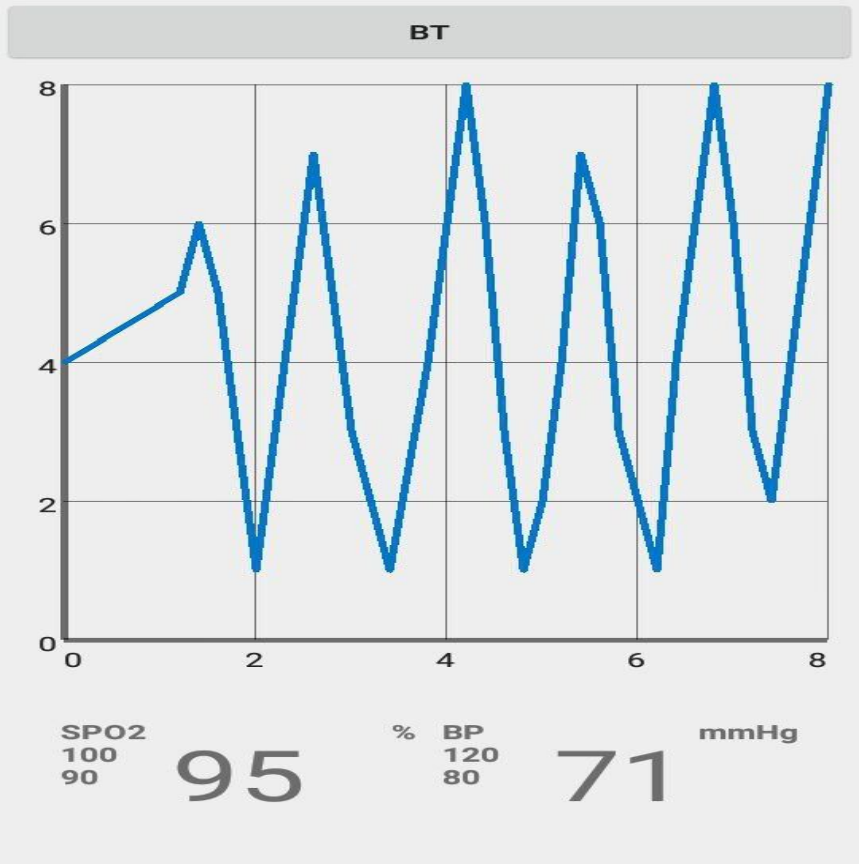



Fig.4.3

## 4.2Discussions

Results are simple, acceptale and accurate. The system is very power efficient. Only thesmartphone or the tablet is needed to be charged enough to do the test.

It is easy to use, fast, accurate, high efficiency, and safe (without any danger of electric shocks). In contrast to other conventionalpatient monitoring equipments the system has the ability to save data for future reference. Finally, thereliability and validity of the system have been ensured via field tests. The field tests show that it can produce medical data that are similar to those produced by the existing medicalequipment.

After completing all the procedures the collected data can be used to monitor (in real time) the state of a patient or to get sensitive information in order to be subsequently analyzed for medical diagnosis. The Android applications have been designed in order to easily see the patient's information.

# Chapter Five

# Conclusions & Recommendations

# 5.1CONCLUSION

Our system is simple. It is just few wires, sensors and Bluetooth module connected to a small kit Arduino Uno with a smartphone. The system is very power efficient. Only the smartphone or the tablet needs to be charged enough to do the test. It is easy to use, fast and safe (without any danger of electric shocks). In contrast to other conventional medical equipment the system has the ability to save data for future reference to be recall any time. Finally, the reliability and validity of our system have been ensured via field tests. The field tests show that our system can produce medical data that are similar to those produced by the existing medical equipment.

This project was build low cost, low power, reliable, and portable monitoring system that would accurately measure the vital signs. A reliable and continuous vital sign monitoring system targeted rural areas. The resulting system was also low in power and cost, and provided real time monitoring. It is also easy to use and provides accurate measurements. Given the scope of this project, the ECG, temperature, heart rate signal, body temperature SPO2 and blood pressure.

## 5.2Recommendation and future work

This project can be improved and expanded in numerous ways. First of all, the targetgroup for this product can be expanded to include people in everywhere and every ages. Currently, the signal conditioning circuits for these sensors are in analog form on breadboards.Also, the sensors used for measuring heart rate and temperature can be upgraded. All these sensors should be wirelessly connected to the phone and Arduino Uno, making it comfortable and non-invasive for the user to wear.

Some recommendations on future work would be to add vital signs monitor to this system, which is measuring the glucose concentration in blood. This can be achieved through glucose sensor, it can easily be extended to measure it to help diabetes patient. Adding this last sensing component would make this system a complete vital signs monitor.

In conclusion, with refinements to the design, the smart measuring ECG, blood pressure, heart rate, oxygen saturation and body temperature would make a great competitor against other products that currently exist in the market.

In future, we can develop a big data base of all the patients of any hospital and the these health parameters can be monitored continuosly, and also the information is uploaded to the hospital server. These servers keep the information of the patients in the data base, and doctors can have the access of patient’s history, when any further consulatcy happens with the doctor

# Chapter six
# References

# REFERENCES

[1] FRANCIS S. COLLINS, **Mobile Technology And Healthcare**, http://www.nlm.nih.gov/medlineplus/magazine/issues/winter11

[2] Dhvani Parekh, Prof. Jamal Deen, Project Coordinator: Prof. T. Doyle, **Designing Heart Rate, Blood Pressure and Body Temperature Sensors for Mobile On-Call System**, Department of Electrical and Computer Engineering, Faculty , Electrical and Biomedical Project Report, Submitted in Partial fulfillment of the requirements, for the degree of Bachelor of Engineering, McMaster University Hamilton, Ontario, Canada, April 2010 .

[3] Tell, J.P.; Manjarres, O; Quijano, M.; Blanco, **Remote Monitoring System of ECG and Human Body Temperature Signals**, 2013, IEEE Latin American Transaction, Vol. 11, No. 1, February, pp. 314-318

[4] Fernando Cornelio Jiménez González 1;2;*, Osslan Osiris Vergara Villegas 2, Dulce Esperanza Torres Ramírez 1, Vianey Guadalupe Cruz Sánchez 2 and Humberto Ochoa Domínguez, **Smart Multi-Level Tool for Remote Patient Monitoring Based on a Wireless Sensor Network and Mobile Augmented Reality** ,www.mdpi.com/journal/sensors,. 2014

[5] Alex CorsBardolet , Supervisor: Andrzej G lowacz**, A Remote Patient Monitoring System using Android Mobile Devices**, 30th June 2014

[6] Amna Abdullah, Asma Ismael, Aisha Rashid, Ali Abou-ElNour, and Mohammed Tarique, , **Real Time Wireless Health Monitoring Application Using Mobile Devices**, International Journal of Computer Networks & Communications (IJCNC) Vol.7, No.3, DOI: 10.5121/ijcnc.2015.7302 13Department of Electrical Engineering, Ajman University of Science and Technology,, P.O. Box 2202, Fujairah, United Arab Emirates, May 2015

[7] K. DEEPALAKSHMI, 2 Ms. A. SIVASANKARI,1, 2, Department of computer science, DKM College for women, Vellore, Tamil Nadu, India, **Patient Monitoring System Using Android APP** ISSN 2348-1196 (print) International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online) Vol. 3, Issue 3, pp: (299-308), Available at: www.researchpublish.com Page | 299 Research Publish Journals, , September 2015.

[8] **PATIENT MONITORING SYSTEM USING ANDROID TECHNOLOGY**, IJCSMC, Vol. 2, Issue. 5, pg.191 – 201, RESEARCH ARTICLE © 2013, IJCSMC All Rights Reserved 191, Biomedical Engineering & P.S.N.A College of Engineering and Technology, India, premas@psnacet.edu.in, May 2013.

[9] K.C. Kavitha, A.BazilaBanu, **Wireless Health Care Monitoring**,
Post Graduate Student, Department of Information Technology, Velammal College of Engineering and Technology, Madurai, India, Assistant Professor, Department of Information Technology, Velammal College of Engineering and Technology, Madurai, India

[10] Najeed Ahmed Khan , **Real Time Monitoring of Human Body Vital Signs using Bluetooth and WLAN**, Computer Science & IT Department NED University of Engineering & Technology ,Karachi, Pakistan.

[11] G. Speed, "Primary four."

[12] http://physionet.org/)

[13] M. Tavakoli, L. Turicchia, and R. Sarpeshkar, **An Ultra-Low-Power Pulse**

**Oximeter Implemented with an Energy-E_cientTransimpedanceAmpli-_er**, IEEE Transactions on Biomedical Circuits and Systems, Vol. 4, No. 1, pp. 27-38, Feb. 2010.

# Appendices

## Appendices (A)

```
#include <SoftwareSerial.h>
#define DEBUG true
SoftwareSerialbluetooth(7, 8);
#include <LiquidCrystal.h>
#include <stdlib.h>
LiquidCrystallcd(12, 11, 5, 4, 3, 2);
//Variables
float temp;
int hum;
String tempC;
int error;
intpulsePin = 0;                // Pulse Sensor purple wire connected to analog pin 0
intblinkPin = 13;               // pin to blink led at each beat
intfadePin = 5;
intfadeRate = 0;

// Volatile Variables, used in the interrupt service routine!
volatileint BPM;                // int that holds raw Analog in 0. updated every 2mS
volatileint Signal;             // holds the incoming raw data
```

```
volatileint IBI = 600;             // int that holds the time interval between beats! Must be seeded!

volatileboolean Pulse = false;     // "True" when heartbeat is detected. "False" when not a "live beat".

volatileboolean QS = false;        // becomes true when Arduino finds a beat.


// Regards Serial OutPut  -- Set This Up to your needs

staticbooleanserialVisual = true;   // Set to 'false' by Default.  Re-set to 'true' to see Arduino Serial Monitor ASCII Visual Pulse

volatileint rate[10];                    // array to hold last ten IBI values

volatile unsigned long sampleCounter = 0;          // used to determine pulse timing

volatile unsigned long lastBeatTime = 0;           // used to find IBI

volatileint P = 512;                     // used to find peak in pulse wave, seeded

volatileint T = 512;                     // used to find trough in pulse wave, seeded

volatileint thresh = 525;                // used to find instant moment of heart beat, seeded

volatileint amp = 100;                   // used to hold amplitude of pulse waveform, seeded

volatilebooleanfirstBeat = true;         // used to seed rate array so we startup with reasonable BPM

volatilebooleansecondBeat = false;       // used to seed rate array so we startup with reasonable BPM

void setup()

{
```

```
lcd.begin(16, 2);
lcd.print("heath monitor");
delay(100);
lcd.setCursor(0, 1);
lcd.print("Connecting...");
bluetooth.begin(9600);
Serial.begin(9600); //or use default 115200.
interruptSetup();
}
void loop() {
lcd.clear();
start: //label
error = 0;
lcd.setCursor(0, 0);
lcd.print("BPM = ");
lcd.print(BPM);
bluetooth.println("BPM="+String(BPM));
int x = analogRead(A1);
float y = (x / 1024.0) * 5000;
float temp = y / 10;
lcd.setCursor(0, 1);
lcd.print("TEMP= ");
lcd.print(temp);
```

```
bluetooth.println("TEMP="+String(temp));

delay (100);

  //Resend if transmission is not completed

if (error == 1) {

goto start; //go to label "start"

  }

delay(3000);

}

voidinterruptSetup() {

  TCCR2A = 0x02;     // DISABLE PWM ON DIGITAL PINS 3 AND 11, AND GO INTO CTC MODE

  TCCR2B = 0x06;     // DON'T FORCE COMPARE, 256 PRESCALER

  OCR2A = 0X7C;      // SET THE TOP OF THE COUNT TO 124 FOR 500Hz SAMPLE RATE

  TIMSK2 = 0x02;     // ENABLE INTERRUPT ON MATCH BETWEEN TIMER2 AND OCR2A

sei();            // MAKE SURE GLOBAL INTERRUPTS ARE ENABLED

}

ISR(TIMER2_COMPA_vect) {                  // triggered when Timer2 counts to 124

cli();                                    // disable interrupts while we do this

  Signal = analogRead(pulsePin);              // read the Pulse Sensor

sampleCounter += 2;                       // keep track of the time in mS
```

```
int N = sampleCounter - lastBeatTime;       // monitor the time since the last beat to avoid noise

//  find the peak and trough of the pulse wave

if (Signal < thresh && N > (IBI / 5) * 3) { // avoid dichrotic noise by waiting 3/5 of last IBI

if (Signal < T) {                    // T is the trough

     T = Signal;                       // keep track of lowest point in pulse wave

   }

  }

if (Signal > thresh && Signal > P) {      // thresh condition helps avoid noise

   P = Signal;                          // P is the peak

  }                                   // keep track of highest point in pulse wave

  //  NOW IT'S TIME TO LOOK FOR THE HEART BEAT

  // signal surges up in value every time there is a pulse

if (N > 250) {                              // avoid high frequency noise

if ( (Signal > thresh) && (Pulse == false) && (N > (IBI / 5) * 3) ) {

     Pulse = true;                          // set the Pulse flag when there is a pulse

digitalWrite(blinkPin, HIGH);               // turn on pin 13 LED

     IBI = sampleCounter - lastBeatTime;        // time between beats in mS

lastBeatTime = sampleCounter;               // keep track of time for next pulse

if (secondBeat) {                    // if this is the second beat

secondBeat = false;                  // clear secondBeat flag

for (int i = 0; i <= 9; i++) {       // seed the running total to get a realistic BPM at startup
```

```
rate[i] = IBI;
      }
    }
if (firstBeat) {                    // if it's the first time beat is found
firstBeat = false;                  // clear firstBeat flag
secondBeat = true;                    // set the second beat flag
sei();                              // enable interrupts again
return;                              // IBI value is unreliable so discard it
    }
wordrunningTotal = 0;                  // clear the runningTotal variable

for (int i = 0; i <= 8; i++) {          // shift data in the rate array
rate[i] = rate[i + 1];                // and drop the oldest IBI value
runningTotal += rate[i];              // add up the 9 oldest IBI values
    }
rate[9] = IBI;                        // add the latest IBI to the rate array
runningTotal += rate[9];                // add the latest IBI to runningTotal
runningTotal /= 10;                   // average the last 10 IBI values
    BPM = 60000 / runningTotal;             // how many beats can fit into a
minute? that's BPM!
    QS = true;                          // set Quantified Self flag
    // QS FLAG IS NOT CLEARED INSIDE THIS ISR
  }
```

```
  }
if (Signal < thresh && Pulse == true) {  // when the values are going down, the beat is over
digitalWrite(blinkPin, LOW);          // turn off pin 13 LED
   Pulse = false;                     // reset the Pulse flag so we can do it again
amp = P - T;                          // get amplitude of the pulse wave
thresh = amp / 2 + T;                 // set thresh at 50% of the amplitude
   P = thresh;                        // reset these for next time
   T = thresh;
  }
if (N > 2500) {                       // if 2.5 seconds go by without a beat
thresh = 512;                         // set thresh default
   P = 512;                           // set P default
   T = 512;                           // set T default
lastBeatTime = sampleCounter;         // bring the lastBeatTime up to date
firstBeat = true;                     // set these to avoid noise
secondBeat = false;                   // when we get the heartbeat back
  }
sei();
  // enable interrupts when youre done!
}// end
```

# Appendices (B)

**login code:**

```
packagecom.uni.bluetoothtest;
importandroid.content.Intent;
import android.support.v7.app.AppCompatActivity;
importandroid.os.Bundle;
importandroid.view.View;
importandroid.widget.Button;
importandroid.widget.TextView;
public class Login extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.login);
      Button login = (Button) findViewById(R.id.login) ;
TextView register = (TextView) findViewById(R.id.register) ;
login.setOnClickListener(new View.OnClickListener() {
         @Override
public void onClick(View v) {
startActivity(new Intent(Login.this , Main.class));
finish();
         }
```

```
        });
register.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View v) {
startActivity(new Intent(Login.this , Register.class));
finish();
            }
        });
    }
}
```

**control code:**

```
packagecom.uni.bluetoothtest;
importandroid.app.ProgressDialog;
importandroid.bluetooth.BluetoothAdapter;
importandroid.bluetooth.BluetoothDevice;
importandroid.bluetooth.BluetoothSocket;
importandroid.content.Intent;
importandroid.content.pm.ActivityInfo;
importandroid.os.AsyncTask;
import android.support.v7.app.AppCompatActivity;
importandroid.os.Bundle;
importandroid.view.MotionEvent;
importandroid.view.View;
```

```
importandroid.view.Window;

importandroid.widget.Button;

importandroid.widget.SeekBar;

importandroid.widget.Toast;

importjava.io.IOException;

importjava.util.UUID;

public class Control extends AppCompatActivity {

   Button baseLeft, baseRight ,elbowUp , elbowDown, kneeUp , kneeDown,
wristUp , wristDown,  gripClose , gripOpen,   btnDis;

   String address = null;

privateProgressDialog progress;

BluetoothAdaptermyBluetooth = null;

BluetoothSocketbtSocket = null;

privatebooleanisBtConnected = false;

static final UUID myUUID = UUID.fromString("00001101-0000-1000-8000-
00805F9B34FB");

   @Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

     //
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCA
PE);

setContentView(R.layout.control);

       //receive the address of the bluetooth device
```

```
        Intent newint = getIntent();
address = newint.getStringExtra("address");
//call the widgtes
baseLeft = (Button)findViewById(R.id.baseLeft);
baseRight = (Button)findViewById(R.id.baseRight);
elbowUp = (Button)findViewById(R.id.elbowUp);
elbowDown = (Button)findViewById(R.id.elbowDown);
kneeUp = (Button)findViewById(R.id.kneeUp);
kneeDown = (Button)findViewById(R.id.kneeDown);
wristUp = (Button)findViewById(R.id.wristUp);
wristDown = (Button)findViewById(R.id.wristDown);
gripClose = (Button)findViewById(R.id.gripClose);
gripOpen = (Button)findViewById(R.id.gripOpen);
btnDis = (Button)findViewById(R.id.btnDis);
newConnectBT().execute()  ;
baseLeft.setOnTouchListener(new View.OnTouchListener() {
            @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+basel*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
```

```
            }
return false;
            }
        });
baseRight.setOnTouchListener(new View.OnTouchListener() {
            @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+baser*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false;
            }
        });
elbowUp.setOnTouchListener(new View.OnTouchListener() {
            @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+elbwu*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
```

```
send("+1234*");
            }
return false;
        }
    });
elbowDown.setOnTouchListener(new View.OnTouchListener() {
        @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+elbwd*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false
        }
    });
kneeUp.setOnTouchListener(new View.OnTouchListener() {
        @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+kneeu*");
return true;
```

```
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false;
          }
       });
kneeDown.setOnTouchListener(new View.OnTouchListener() {
          @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+kneed*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false;
          }
       });
wristUp.setOnTouchListener(new View.OnTouchListener() {
          @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+wrstu*");
```

```
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false;
          }
       });
wristDown.setOnTouchListener(new View.OnTouchListener() {
          @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+wrstd*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false;
          }
       });
gripClose.setOnTouchListener(new View.OnTouchListener() {
          @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
```

```
send("+gribc*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false;
          }
       });
gripOpen.setOnTouchListener(new View.OnTouchListener() {
          @Override
publicbooleanonTouch(View view, MotionEventmotionEvent) {
if(motionEvent.getAction() == MotionEvent.ACTION_DOWN){
send("+gribo*");
return true;
}else if (motionEvent.getAction() == MotionEvent.ACTION_UP){
send("+1234*");
            }
return false;
          }
       });
btnDis.setOnClickListener(new View.OnClickListener() {
          @Override
public void onClick(View view) {
```

```
Disconnect();
            }
        });
    }
private class ConnectBT extends AsyncTask<Void, Void, Void>  // UI thread
    {
privatebooleanConnectSuccess = true; //if it's here, it's almost connected
        @Override
protected void onPreExecute()
        {
progress = ProgressDialog.show(Control.this, "Connecting...", "Please
wait!!!");  //show a progress dialog
        }
        //    +1234*
        @Override
protected Void doInBackground(Void... devices) //while the progress dialog is
shown, the connection is done in background
        {
try
            {
if (btSocket == null || !isBtConnected)
                {
myBluetooth = BluetoothAdapter.getDefaultAdapter();//get the mobile
bluetooth device
```

```
BluetoothDevicedispositivo =
myBluetooth.getRemoteDevice(address);//connects to the device's address and
checks if it's available

btSocket =
dispositivo.createInsecureRfcommSocketToServiceRecord(myUUID);//create a
RFCOMM (SPP) connection

BluetoothAdapter.getDefaultAdapter().cancelDiscovery();

btSocket.connect();//start connection

            }

        }

catch (IOException e)

        {

ConnectSuccess = false;//if the try failed, you can check the exception here

        }

return null;

    }

    @Override

protected void onPostExecute(Void result) //after the doInBackground, it
checks if everything went fine

    {

super.onPostExecute(result);

if (!ConnectSuccess)

        {

msg("Connection Failed. Is it a SPP Bluetooth? Try again.");
```

```
finish();
            }
else
            {
msg("Connected.");
isBtConnected = true;
            }
if (progress != null &&progress.isShowing()) {
progress.dismiss();
            }
        }
    }
    @Override
protected void onDestroy() {
progress.dismiss();
super.onDestroy();
    }
private void msg(String s)
    {
Toast.makeText(getApplicationContext(),s,Toast.LENGTH_LONG).show();
    }
private void Disconnect()
    {
```

```
if (btSocket!=null) //If the btSocket is busy
        {
try
            {
btSocket.close(); //close connection
            }
catch (IOException e)
{ msg("Error");}
        }
finish(); //return to the first layout
    }
private void send(String message)
    {
if (btSocket!=null)
        {
try
            {
btSocket.getOutputStream().write(message.getBytes());
            }
catch (IOException e)
            {
msg("Error");
            }
```

```
        }
    }
}
```

**main code:**

```
packagecom.uni.bluetoothtest;
importandroid.app.ProgressDialog;
importandroid.bluetooth.BluetoothAdapter;
importandroid.bluetooth.BluetoothDevice;
importandroid.bluetooth.BluetoothSocket;
importandroid.content.Intent;
importandroid.os.AsyncTask;
importandroid.os.ParcelUuid;
import android.support.v7.app.AppCompatActivity;
importandroid.os.Bundle;
import android.support.v7.widget.Toolbar;
importandroid.view.Menu;
importandroid.view.MenuItem;
importandroid.view.View;
importandroid.widget.AdapterView;
importandroid.widget.ArrayAdapter;
importandroid.widget.Button;
importandroid.widget.ListView;
importandroid.widget.TextView;
```

```
importandroid.widget.Toast;
importandroid.bluetooth.BluetoothDevice;
importjava.util.ArrayList;
importjava.util.ResourceBundle;
importjava.util.Set;
importjava.util.UUID;
public class Main extends AppCompatActivity {
privateBluetoothAdaptermyBluetooth = null;
private Set<BluetoothDevice>pairedDevices;
   Button baseLeft ,btnPaired;
ListViewdevicelist;
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
      Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar) ;
setSupportActionBar(toolbar);
devicelist = (ListView)findViewById(R.id.list);
myBluetooth = BluetoothAdapter.getDefaultAdapter();
if(myBluetooth == null)
      {
         //Show a mensag. thatthedevice has no bluetooth adapter
```

```
Toast.makeText(getApplicationContext(), "Bluetooth Device Not Available",
Toast.LENGTH_LONG).show();

        //finish apk

finish();

      }

else

      {

if (myBluetooth.isEnabled())

        { }

else

        {

          //Ask to the user turn the bluetooth on

          Intent turnBTon = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);

startActivityForResult(turnBTon,1);

        }

      }

   }

private void pairedDevicesList()

   {

pairedDevices = myBluetooth.getBondedDevices();

ArrayList list = new ArrayList();

if (pairedDevices.size()>0)
```

```
        {
for(BluetoothDevicebt : pairedDevices)
            {
list.add(bt.getName() + "\n" + bt.getAddress()); //Get the device's name and the address
            }
        }
else
        {
Toast.makeText(getApplicationContext(), "No Paired Bluetooth Devices Found.", Toast.LENGTH_LONG).show();
        }
finalArrayAdapter adapter = new ArrayAdapter(this,android.R.layout.simple_list_item_1, list);
devicelist.setAdapter(adapter);
devicelist.setOnItemClickListener(myListClickListener); //Method called when the device from the list is clicked
    }
privateAdapterView.OnItemClickListenermyListClickListener = new AdapterView.OnItemClickListener()
    {
public void onItemClick (AdapterViewav, View v, int arg2, long arg3)
        {
            // Get the device MAC address, the last 17 chars in the View
```

```
        String info = ((TextView) v).getText().toString();
        String address = info.substring(info.length() - 17);
        // Make an intent to start next activity.
        Intent i = new Intent(Main.this, Receive.class);
        //Change the activity.
i.putExtra("address", address); //this will be received at ledControl (class) Activity
startActivity(i);
      }
   };
   @Override
protectedbooleanonPrepareOptionsPanel(View view, Menu menu) {
getMenuInflater().inflate(R.menu.actions_menu,menu);
returnsuper.onPrepareOptionsPanel(view, menu);
   }
   @Override
publicbooleanonOptionsItemSelected(MenuItem item) {
switch (item.getItemId()) {
caseR.id.scan :
pairedDevicesList();
break;
      }
returnsuper.onOptionsItemSelected(item);
```

```
    }
}
```

**receive code:**

```
packagecom.uni.bluetoothtest;
importandroid.app.ProgressDialog;
importandroid.bluetooth.BluetoothAdapter;
importandroid.bluetooth.BluetoothDevice;
importandroid.bluetooth.BluetoothSocket;
importandroid.content.Intent;
importandroid.content.pm.ActivityInfo;
importandroid.os.AsyncTask;
importandroid.os.CountDownTimer;
importandroid.os.Message;
import android.support.v7.app.AppCompatActivity;
importandroid.os.Bundle;
importandroid.util.Log;
importandroid.widget.Button;
importandroid.widget.TextView;
importandroid.widget.Toast;
import com.jjoe64.graphview.GraphView;
import com.jjoe64.graphview.series.DataPoint;
import com.jjoe64.graphview.series.LineGraphSeries;
importjava.io.IOException;
```

```
importjava.io.InputStream;

importjava.io.OutputStream;

importjava.io.UnsupportedEncodingException;

importjava.util.UUID;

importjava.util.logging.Handler;

importjava.util.logging.LogRecord;

public class Receive extends AppCompatActivity {

   Button baseLeft, baseRight ,elbowUp , elbowDown, kneeUp , kneeDown, wristUp , wristDown,  gripClose , gripOpen,   btnDis;

   String address = null;

privateProgressDialog progress;

BluetoothAdaptermyBluetooth = null;

BluetoothSocketbtSocket = null;

privatebooleanisBtConnected = false;

static final UUID myUUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

byte[] mmBuffer ;

   Handler mHandler ;

InputStreammmInStream ;

TextViewtxt ;

TextViewhb ;

TextViewtemp ;

TextViewbrpm ;
```

```
    @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
      //
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCA
PE);
setContentView(R.layout.receive);
       Intent newint = getIntent();
address = newint.getStringExtra("address");
txt = (TextView) findViewById(R.id.txt)  ;
hb = (TextView) findViewById(R.id.heartbeat)  ;
temp = (TextView) findViewById(R.id.temp)  ;
brpm = (TextView) findViewById(R.id.brpm)  ;
GraphView graph = (GraphView) findViewById(R.id.graph);
LineGraphSeries<DataPoint> series = new LineGraphSeries<>(new
DataPoint[] {
newDataPoint(0, 1),
newDataPoint(1, 5),
newDataPoint(2, 3),
newDataPoint(3, 2),
newDataPoint(4, 6)
       });
graph.addSeries(series);
newConnectBT().execute() ;
```

```
CountDownTimer timer = new CountDownTimer(10000,500) {
int c = 0 ;
        @Override
public void onTick(long millisUntilFinished) {
brpm.setText(""+c);
            c = c++ ;
if (c > 120) {
                c = c-- ;
brpm.setText(""+c);
            }
        }
        @Override
public void onFinish() {
        }
    };
timer.start();
  }
private class ConnectBT extends AsyncTask<Void, Void, Void>  // UI thread
  {
privatebooleanConnectSuccess = true; //if it's here, it's almost connected
    @Override
protected void onPreExecute()
    {
```

```
        //   progress = ProgressDialog.show(Receive.this, "Connecting...", "Please wait!!!");  //show a progress dialog

        }

        //    +1234*

        @Override

protected Void doInBackground(Void... devices) //while the progress dialog is shown, the connection is done in background

        {

try

            {

if (btSocket == null || !isBtConnected)

                {

myBluetooth = BluetoothAdapter.getDefaultAdapter();//get the mobile bluetooth device

BluetoothDevicedispositivo = myBluetooth.getRemoteDevice(address);//connects to the device's address and checks if it's available

btSocket = dispositivo.createInsecureRfcommSocketToServiceRecord(myUUID);//create a RFCOMM (SPP) connection

BluetoothAdapter.getDefaultAdapter().cancelDiscovery();

btSocket.connect();//start connection

                }

            }

catch (IOException e)
```

```
        {
ConnectSuccess = false;//if the try failed, you can check the exception here
        }
return null;
    }
    @Override
protected void onPostExecute(Void result) //after the doInBackground, it checks if everything went fine
    {
super.onPostExecute(result);
if (!ConnectSuccess)
        {
msg("Connection Failed. Is it a SPP Bluetooth? Try again.");
finish();
        }
else
        {
msg("Connected.");
isBtConnected = true;
InputStreamtmpInput = null ;
try {
tmpInput = btSocket.getInputStream() ;
Log.d("socket" ,tmpInput.toString()) ;
```

```
}catch (IOException e) {
                // Log.d
            }
byte[] buffer = new byte[2048] ;
int bytes ;
try {
bytes = tmpInput.read(buffer) ;
Log.d("buufe" ,buffer.toString()) ;
                String msg = null;
try {
msg = new String (buffer , "utf-8");
txt.setText(msg);
if (msg.contains("T")){
temp.setText(msg);
}else if (msg.contains("B")){
hb.setText(msg);
                    }
                } catch (UnsupportedEncodingException e) {
e.printStackTrace();
                }
txt.setText(msg);
}catch (IOException e){
            }
```

```
            //    new ReadThread().read(btSocket);
            }
if (progress != null &&progress.isShowing()) {
              //  progress.dismiss();
            }
        }
    }
private void msg(String s)
    {
Toast.makeText(Receive.this,s,Toast.LENGTH_LONG).show();
    }
}
```

**register code:**

```
packagecom.uni.bluetoothtest;
importandroid.content.Intent;
import android.support.v7.app.AppCompatActivity;
importandroid.os.Bundle;
importandroid.util.Log;
importandroid.view.View;
importandroid.widget.Button;
importandroid.widget.TextView;
public class Register extends AppCompatActivity {
    @Override
```

```
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.register);
        Button register = (Button) findViewById(R.id.register) ;
register.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View v) {
startActivity(new Intent(Register.this , Login.class));
finish();
            }
        });
    }
}
```