A novel technique to monitor human body vital signs

G. S. Uthayakumar*, A. Sivasubramanian and A. Gratus Iruthaya Kisho

Department of Electronics and Communication Engineering, St. Joseph’s College of Engineering, Chennai-600119

*Correspondence Info:
G. S. Uthayakumar,
Department of Electronics and Communication Engineering,
St. Joseph’s College of Engineering, Chennai-600119
Email: guthayakumar@live.com

Abstract

Vital signs are measures of various physiological statistics, often taken by health professionals, in order to assess the most basic body functions. Vital signs are an essential part of a case presentation. The act of taking vital signs normally entails recording body temperature, pulse rate (or heart rate) and blood pressure, but may also include other measurements. Vital signs often vary by age. There are three vital signs which are standard in most medical settings: body temperature, pulse rate (or heart rate), blood pressure. The equipment needed is a thermometer, a sphygmomanometer, and a watch. Though a pulse can often be taken by hand, a stethoscope may be required for a patient with a very weak pulse. In this research paper, we have introduced the ANDROID operating system to measure the vital signs such as Heart rate, blood pressure and temperature.

Keywords: Operating system, ANDROID, Blood pressure, Heart rate, Temperature.

1. Introduction

From Greek and Indian mythology we came know that, the people in power were able to communicate over a distance by sending message. It is not by the wireless devices as today but they used birds and humans to send their message. In early 19th century, revolution of electronics leaded to the invention of telephone, radio, television, computer and so on. At present the telephone is a hand held device in various forms like mobile phones, PDA (Personal Digital Assistant) and so on. The microcontroller plays an important role in the field of electronic which has reduced the size, complexity and the density of electronic components. In our day to day life every electronic device has a microcontroller in it and changed the entire scenario. To say in simple, the world has become so small, that is on the palm due to electronics. But not electronics, it has conquered almost all the fields starting from agriculture to space research including medicine. The dictionary of medicine had appended a word called telemedicine some time back, yet to find a word for medication through hand held devices; it grows faster than mind waves. The health care industry, or medical industry, is a sector within the economic system that provides goods and services to treat patients with curative, preventive, rehabilitative, and palliative care. The modern health care sector is divided into many sub-sectors, and depends on interdisciplinary teams of trained professionals and paraprofessionals to meet health needs of individuals and populations. In present scenario, the delivery of medical advice along with diagnostics and healthcare services facing two major issues, one is transportation and other time consumption. Because of these issues, the service to the needy person is not in time. Also the optimal ratio of doctor and people is 1:600 but as per the latest data in India the ratio is 1:1700. It is indeed difficult to increase the number of doctor by six times immediately in a country like India to meet medical quality. This is where telemedicine technology comes in picture. Telemedicine through hand held device becomes more appropriate, personalized and affordable because of mobile connectivity. Telemedicine deals with monitoring of patients primary health parameter like blood pressure, temperature,
A Smartphone is a mobile phone built on a mobile operating system. There are many mobile operating systems like BADA, IOS, SYMBIAN, WINDOWS for mobile and ANDROID. The android operating system is one of the revolutionary developments of Google\(^1\). It is an open source code, primarily designed for a touch screen devices. The open source code and permissive licensing allows the software to be freely modified. Additionally, android has community of developers writing applications that extends the functionality of devices, written using Java programming language\(^2\). These smartphone are mostly using ARM processors. Since android OS comes with permissive license, the android smartphones are cost effective and having considerable market share in many countries. In India it has wide spread out almost all the villages of India. Considering the market Share in India and the open source code to develop Apps, we have taken the android smartphones and the android development kit as the base tool for our project to attain the ultimate goal of primary health care for all\(^6\).

1.1. History

In 1625, Santorio, who lived in Venice, published his methods for measuring body temperature with the spirit thermometer and for timing the pulse (heart) rate with a pendulum. The principles for both devices had been established by Galileo. Galileo worked out the uniform periodicity of the pendulum by timing the period of the swinging chandelier in the Cathedral of Pisa, using his own pulse rate as a timer. As a result of this early biomedical-engineering collaboration, however, were ignored. The first scientific report of the pulse rate did not appear until Sir John Floyer published “Pulse-Watch” in 1707. The first published course of fever for a patient was plotted by Ludwig Taube in 1852. With subsequent improvements in the clock and the thermometer, the temperature, pulse rate, and respiratory rate became the standard vital signs. In 1896, Scipione Riva-Rocci introduced the sphygmomanometer (blood-pressure cuff), which permitted the fourth vital sign, arterial blood pressure, to be measured. A Russian physician, Nikolai Korotkoff, applied Riva-Rocci's cuff with a stethoscope developed by the French physician Rene Laennec to allow the auscultator measurement of both systolic and diastolic arterial pressure. Harvey Cushing, a preeminent U.S. neurosurgeon of the early 1900s, predicted the need for and later insisted on routine arterial blood pressure monitoring in the operating room. Cushing answered his own questions by stating that vital-sign measurement should be made routine and its accuracy is very important\(^7\).

2. Smartphone and Android operating system

A Smartphone is a mobile built on a mobile operating system, with more advanced computing capability, connectivity than feature phone. The smartphones has a combined function of personal digital assistant, GPS navigation unit, portable media player and camera to form one multi-use device. Many smartphones also include high resolution touch screens and web browsers that display standard web pages as well as mobile-optimized sites. High speed data access is provided by Wi-Fi and mobile broadband. In recent development in mobile app markets and of mobile commerce has been a driver of mobile adoption\(^8\). There are many operating systems for smartphones like Android, iOS, Symbian, Bada and so on. Such operating systems can be installed on many different phone models, and typically each device can receive multiple OS software updates over its lifetime. A few other upcoming operating systems are Mozilla's Firefox OS, Canonical Ltd.'s Ubuntu Phone, and Tizen.

2.1 Android OS: Android is a Linux-based operating system designed primarily for touch screen mobile devices such as smartphones and tablet computers. Initially developed by Android, Inc., which was backed by Google and then bought it. Android is under ongoing development by Google and the Open Handset Alliance (OHA), and has seen a number of updates to its base operating system since its original release. These updates typically fix bugs and add new features. Android is open source and Google releases the code under the apache license. This open source code and permissive licensing allows the software to be freely modified and distributed by device manufacturers, wireless carriers and enthusiast developer. Android has a large community of developers writing applications (“apps”) that extend the functionality of devices, written primarily in a customized version of the Java programming language. Google play has more than one million applications in it. Android's open nature has further encouraged a large community of developers and enthusiasts to use the open source code as a foundation for community-driven projects, which add new features for advanced users or bring Android to devices which were officially released running Other Operating Systems\(^9\).

2.2 Eclipse IDE (Integrated Development Environment): In computer programming, Eclipse is a multi-language software development environment comprising a base workspace and an extensible system for customizing the environment. It is written mostly in Java. It can be used to develop applications in Java and, by means of, various plug-in ADA, ADK, C, C++, COBOL, FORTRAN, Haskell, JavaScript, Perl, PHP, Python, R, Ruby, Scala,
Clojure, Groovy, Scheme, and Erlang. The ADK plug-in is the only plug-in for the development of android application in Eclipse IDE.

2.3 Android SDK (Software Development Kit): Android software development is the process by which new applications are created for the Android operating system. Applications are usually developed in the Java programming language using the Android Software Development Kit. The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.5.8 or later, Windows XP or later. The officially supported integrated (IDE) is Eclipse using the Android Development Tools (ADT) Plug-in, though IntelliJ IDEA IDE (all editions) fully supports Android development out of the box, and Net Beans IDE also supports Android development via a plug-in. Additionally, developers may use any text editor to edit Java and XML files, then use command line tools (Java Development Kit and Apache are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

3. ARM Development kit and sensors

3.1 Temperature Sensor: The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±1/4°C at room temperature and ±3/4°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35’s low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55°C to +150°C temperature range, while the LM35C is rated for a −40°C to +110°C range (−10°C with improved accuracy). For every 1°C change in temperature there will be a variation of +10mV in its analog output. Operating voltage is from 4V to 20V. The output V_{out} gives an analog output which has to be converted into digital. The accuracy of LM35 is of ±0.5°C at 25°C. The pin out of temperature sensor is shown in the Fig.(4.8). LM35 Sensor is used for the measurement of body temperature. Initially the hardware part is calibrated to room temperature. When LM35 is touched by the patient whose temperature is to be measured there will be a variation in analog output of the sensor.

3.2 Blood Pressure: Blood pressure (BP), sometimes referred to as arterial blood pressure, is the pressure exerted by circulating blood upon the walls of blood vessels, and is one of the principal vital signs. When used without further specification, "blood pressure" usually refers to the arterial pressure of the systemic. A person’s blood pressure is usually expressed in terms of the systolic pressure over diastolic pressure and is measured in millimeters of mercury (mmHg). There are many methods to measure the pressure of human. Some of the methods are Auscultatory method, oscillometry method etc.

3.2.1 Normal Range of Blood Pressure: The blood pressure is different for different age groups. The range of pressure for different age group is given below.

| Stages       | Approximate age | Systolic | Diastolic |
|--------------|-----------------|----------|-----------|
| Infants      | 1 to 12 months  | 75–100   | 50–70     |
| Toddlers     | 1 to 4 years    | 80–110   | 50–80     |
| Preschoolers | 3 to 5 years    | 80–110   | 50–80     |
| School age   | 6 to 13 years   | 85–120   | 50–80     |
| Adolescents  | 13 to 18 years  | 95–140   | 60–90     |
3.2.2 Auscultatory Method: The auscultatory method (from the Latin word for "listening") uses a stethoscope and a sphygmomanometer as in Fig.(4.9). This comprises an inflatable (Riva-Rocci) cuff placed around the upper arm at roughly the same vertical height as the heart, attached to a mercury or aneroid manometer. The mercury manometer, considered the gold standard, measures the height of a column of mercury, giving an absolute result without need for calibration and, consequently, not subject to the errors and drift of calibration which affect other methods. The use of mercury manometers is often required in clinical trials and for the clinical measurement of hypertension in high-risk patients, such as pregnant women.

A cuff of appropriate size is fitted smoothly and snugly, then inflated manually by repeatedly squeezing a rubber bulb until the artery is completely occluded. Listening with the stethoscope to the brachial artery at the elbow, the examiner slowly releases the pressure in the cuff. When blood just starts to flow in the artery, the turbulent flow creates a "whooshing" or pounding (first Korotkoff sound). The pressure at which this sound is first heard is the systolic blood pressure. The cuff pressure is further released until no sound can be heard (fifth Korotkoff sound), at the diastolic arterial pressure.

4. Temperature Measurement: The temperature sensor LM35 is used in this project. LM35 series temperature sensors belong to precision integrated-circuit temperature sensors. The output of the sensor is linearly proportional to the Celsius (Centigrade) temperature. The LM35 is rated to operate over a range of -55°C to 150°C. It has an accuracy of 0.5°C at 25°C. For every 1°C change in temperature there will be a variation of +10mV in its analog output.

Fig.1. Block diagram of temperature measurement.

The analog output from temperature sensor is connected to ADC input of friendly ARM kit. There are ten ADC Channels in this development kit used for various purposes like touch screen and so. Here in our project we are using Channel 0 of the ADC in 12-bit mode to measure temperature as shown in block diagram Fig.1. The working of ADC in ARM processor goes on the following sequence. The mode selection to select 10bit/12bit conversion. The channel selection input given to select desired channel, here it is channel 0. Following channel selection analog MUX connects the selected channel conversion unit. Then the start of conversion (SOC) is given to ADC. After receiving the end of conversion (EOC) from ADC, the digital output data is readout from the ADC. All the above said operations are in I2C protocol. To make user friendly flexible development, the Friendly ARM kit gives a function “Hardwarecontroller.readadc( )” Which takes care of all the actives of ADC pertaining to reading a digital data of given analog input channel and returns value of the analog input. The value returned by the function is a raw digital data of the analog input signal. This value must be normalized. Here in this project the 12 bit ADC mode is used. This means that, the maximum number of combination is $2^{12}=4096$. Therefore the step size is maximum input voltage which is 3.23V in this case divided by number of combinations.

Step size = $\frac{3.23}{4096}=0.0007887$

Therefore the digital value of the given analog input is the value return by the function multiplied by step size. For example

The input analog voltage at channel 0 = 1.615V

The value return by the function = 2048

The digital value = $2048\times0.0007887 = 1.615$

In this temperature sensor LM35 for every 1°C change of temperature, there is a change of 10mV in its output. This means for 33°C of temperature, the output will be 330mV (0.33V). If this voltage is given as input to channel 0 of the ADC, the I2C function will return a value ~419.

The value return by the function = 419

The digital value = $419\times0.0007887=0.3304$

The digital value is 0.3304 but the actual temperature is 33°C. So the digital value must normalize by multiplying with 100, which is the step ratio. Therefore the temperature = $0.3304\times100=33.04(33°C)$ The data in the form of temperature thus got is displayed in Friendly ARM display. The same can be displayed in any android mobile with the Apps.
The program listing for Temperature Sensing and Display
package com.friendlyarm.temp_app;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.EditText;
import com.friendlyarm.AndroidSDK.HardwareControler;
public class Temp_app_activity extends Activity implements OnClickListener {
    private Button start;
    private int x, Temp;
    private char i;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_temp_app_activity);
        start = (Button)findViewById(R.id.button1);
        start.setOnClickListener(this);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.Temp_app_activity, menu);
        return true;
    }
    @Override
    public void onClick(View v) {
        if (v.getId()== R.id.button1) {
            i=0x00;
            HardwareControler.setLedState(0,1);
            x=HardwareControler.readADC();
            Temp=x*0.07887;
            HardwareControler.setLedState(0,0);
            EditText editText=EditText)findViewById(R.id.editText1);
            editText.setText(String.valueOf(Temp),
            TextView.BufferType.EDITABLE);
        }
    }
}

Fig.2. Photograph of temperature interface.
To do all these stepwise activities a program is developed using JAVA as programming language in Android platform with Eclipse IDE. The program is converted as an android App and loaded in Friendly ARM kit to capture the analog data and to display. The photograph of the android based system developed in this project is given in Fig.(5.2). The snapshot of the program out from an android mobile is given in Fig.3. The developed program, which is converted as android App is listed here.

4.1 Blood Pressure measurement: The second application development of this project is blood pressure measurement through smartphone. In order to measure the blood pressure of a human, a blood pressure unit is interfaced with Friendly ARM kit as in Fig. 5.4. The blood pressure measurement unit used here a wrist type unit. After wrapping the unit at the wrist, the unit is started with start switch. By this action the unit steps into housekeeping operations like display, sensor and motor initialing and control. Then the air is inflated into the cuff up to 200 millimeter of mercury (mmHg), which is controlled by internal controller of the unit. After that the air starts to deflate slowly in order to capture the maximum amplitude of pulse i.e. Mean Arterial Pressure (MAP) amplitude from which the systolic and diastolic pressure can be calculated. The principle of blood pressure measurement in this unit is Oscillometry method, which is described in chapter 4. The pressure sensor typically employs the piezo resistive principle to convert pressure to an electrical signal. The analog output from pressure sensor is connected to ADC (channel 0 input) of Friendly ARM kit. The channel 0 of the ADC is working in 12-bit mode. The working of ADC, I2C protocol and step size calculations are discussed in section 5.1. The same I2C protocol function “Hardwarecontroller.readadc( ) “ for ADC is used here too.

The blood pressure measurement unit used in this project is from Omron. This product is a complete stand alone instrument to measure blood pressure. This project needs a cuff with inflating and deflating arrangement and pressure sensor. Instead of designing and constructing the above requirements, they are borrowed  from the stand alone blood pressure measurement unit. From the pressure sensor an output is tapped out through signal conditioning circuit and the filter as shown Fig.5. Reset of the circuits and other setups like program are not used in this project. There are two reasons for this intermediate arrangement. The first reason is pertaining to the main task of this project. The main aim of the project is to interface a sensor with ARM processor and to develop android based application software to read the analog data from the sensor and to convert into digital data and then to display in a mobile display. The output of this project supposed to be compared with some standard results for its correctness. This is the second reason to have standard and stand alone blood pressure unit in this project. Since the both displays are taking analog data from same pressure sensor, they can be compared and this project design can be fine tuned if it is needed. By having the data in android platform, the data can be transported to any place, say a doctor, which enables remote and telemedicine. Whereas the standalone unit, can be used only for localized environment.
4.2 Working of Blood Pressure Measurement: The standalone unit and the ARM development kit using android app are measuring the pressure simultaneously with following procedure. Wrap the cuff at wrist of arm and conform that the hand is not stiff, Start the unit by pressing start/stop switch, after standalone unit initializing start the android App by touching start button on the development kit display, the standalone unit inflates the cuff and deflate cuff and measures the pressure and pulse rate and displays it on its own display, using the Analog out from the pressure sensor this project kit measures the pressure and displays it on the ARM kit display. Android app developed in this project reads the analog data from the pressure sensor and converts into digital data using the function “Hardwarecontroller.readadc( ) “ during the deflating time. The data received is compared with noise level, if the data is above the noise level the data is stored on a variable say ‘x’, if the data is below the noise level the data is ignored assuming as noise. The process is repeated until to get a valid data. The noise level fixed here is 0.004V (5counts) by trail. This noise level can be varied according to the requirement and situation.

After getting the valid data the multiple sampling is initiated. Every time the previous value is compared with the current value. If the current value is greater than previous value, the previous value is replaced with current value; otherwise the current value is ignored. By this process our sampling on the variable ‘x’ is tending towards the highest value of the pulse amplitude. Finally the variable ‘x’ will have the highest of the pulse amplitude; which is Mean Arterial Pressure (MAP). This highest amplitude MAP is converted into voltage MAP by multiplying with step size (ref. section 5.1). Further the voltage MAP is normalized by multiplying with normalizing factor, which is system depend in this case, it is 100. This normalized MAP is the final value which is used to calculate the pressure.

The systolic pressure=MAP*0.85 (mmHg).
The diastolic pressure=MAP*0.55(mmHg).

The calculated systolic and diastolic pressure further displayed on the display of the Friendly ARM kit. For an example a trial gives the highest pulse amplitude MAP as 1600. Then the calculation as follows

The voltage MAP = 1600*0.0007887 = 1.26192 V
The normalized MAP = 1.26192*100 = 126.192
The systolic pressure = 126.192*0.85 = 107.2632(~107mmHg).
The diastolic pressure = 126.192*0.55 = 69.4056(~69mmHg).

An important point to be noted is that, the frequency of the heart pulse is 1.2Hz. Whereas the clock frequency of the ARM processor used here is 1GHz. Therefore the sampling rate is very high of the order of 400000 times, which results a high accuracy on maximum value. The drawback is lot of redundant data on data capturing with the high sampling rate. This eliminated with previous and current value comparison. The data in the form of pressure thus got is displayed in Friendly ARM display as systolic and diastolic pressure. The same can be displayed in any android mobile with the Apps.

An easy and visual indication on the operation is given with a LED. The LED is connected to GPIO 0 of the ARM processor. The LED goes ON during sampling and puts down to OFF once the sampling and calculation gets over. This task is handled with the hardware function “Hardwarecontroller.ledstate(0,1)” to switch ON the LED. Here ‘0’ stands to say it is GPIO port 0 and ‘1’ stands to send a ‘1’ state at that GPIO port. The LED is switched off with the hardware function “Hardwarecontroller.ledstate(0,0)”
With this function GPIO port 0 will get a ‘0’ state so that LED is switched OFF. To do all these stepwise activities a program is developed using JAVA as programming language in Android platform with Eclipse IDE. The program is converted as an android App and loaded in Friendly ARM kit to capture the analog data and to display. The photograph of the setup is given in Fig.6 and the snapshot of the program out from an android mobile is given in Fig.7. The developed program, which is converted as android App is listed here.

4.3 Pulse Rate Measurement: The number of pulses per minute is measured by measuring time per pulse. This task is incorporated with blood pressure measurement. When the pulse amplitude is measured the function Hardwarecontroller.readadc( ) returns a zero as the value when the pulse is at the lower most position. The time between two subsequent pulses zero position is measured in terms of number of loop execution. This number of loop execution is multiplied by the time per loop execution, which results the time per pulse. From this time per pulse the number of pulse per minute is calculated and displayed at the Friendly ARM display.

The program listing for Blood pressure and Pulse rate Sensing and Display

```java
package com.friendlyarm.pressure_app;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.EditText;
import com.friendlyarm.AndroidSDK.HardwareControler;

public class Pressure_app extends Activity implements OnClickListener {
    private Button Start;
    private int x,y;
    int z[] = new int[3000];
    private int j,s;
    int b[] = new int[3000];
    int g[] = new int[3000];
    private double map,sp,dp,pu;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pressure_app);
        Start = (Button)findViewById(R.id.button1);
        Start.setOnClickListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    getMenuInflater().inflate(R.menu.pressure_app, menu);
    return true;

    int h,k,i=0;
    public boolean onCreateOptionsMenu(Menu menu) getMenuInflater().inflate(R.menu.pressure_app, menu);
    return true;
}

int h,k,i=0;
public void onClick(View v)
{

    if (v.getId()== R.id.button1)
    {
        HardwareController.setLedState(0,1);s=1; y=1;
        for(j=0;j<3000;j++)
        {
            for(i=0;i<130;i++)
            {
```
x=HardwareControler.readADC();
if (y==1 & & x<s)y=0;
if (y==0 & & x>s)s=x;

EditText editText2 = EditText) findViewById(R.id.editText2);
editText2.setText("Sensing Over", TextView.BufferType.EDITABLE);
HardwareControler.setLedState(0,0);

map=s*0.07887;
sp=map*0.85-6;
dp=map*0.55-6;
pu=((sp-dp)*2)*0.90);

EditText editText5 = (EditText) findViewById(R.id.editText5);
editText5.setText(String.valueOf(sp), TextView.BufferType.EDITABLE);

EditText editText1 = (EditText) findViewById(R.id.editText1);
editText1.setText(String.valueOf(dp), TextView.BufferType.EDITABLE);

EditText editText8 = (EditText) findViewById(R.id.editText8);
editText8.setText(String.valueOf(pu), TextView.BufferType.EDITABLE);

Fig. 6 Photograph of blood pressure interface.

Fig.7. Snapshoot of blood pressure display.
5. Conclusion

This research work were formulated and started with tasks like identifying a development kit, sensors for temperature and pressure, android IDE and the understanding of medical terminology, concept and methodology. All these tasks and environment, procedural and sequential working tasks were resolved one after another by careful study over the issues and with the help of technical supports rented from various quarters.

The identification of development kit was done through data collection from internet and electronic journals. There are numerous kits in the international scenario, few are not affordable and few other were not available within the stipulated time. Compromising and consolidating the various factors, the Friendly ARM Kit was purchased. The kit was installed with proper hardware environment and software needed and it was made to work for the requirement. The hardware organization and port organization were studied to develop suitable android apps. Out of available IDEs the eclipse, the free version, where down loaded and installed. The various menus like project creation, compilation and emulator were studied to develop android apps. The temperature sensor LM35 was selected since it is meeting the needs of the project. LM35 sensor was studied and interfaced with Friendly ARM kit. Regarding pressure sensor there were some difficulty for its availability and technical specification. The difficulties were resolved and Omron blood pressure measuring unit was selected and purchased. The analog output from the pressure sensor was tapped out and interfaced with Friendly ARM kit. Using eclipse IDE two android apps, namely temp_app for temperature sensor and the pressure_app for blood pressure and pulse rate sensing were developed and tested with the emulator. The correctness of the apps were conformed and installed in the Friendly ARM kit with its respective interface circuits. The temp_app with temperature interface circuit was tested for various temperatures. It found that it is working for accuracy of 0.1°C.

The pressure_app with its interface circuit was trailed for many times on various occasions and compared with standalone unit output. It is found that the system is working with a small difference of ±5mmHg. This difference may be because of the difference in approximation procedure between the standalone unit and this design. The approximation procedure of the standalone unit is unknown. Thus the project is found to be working for its requirement with higher satisfaction. These two apps can be used with any android based smartphones and Tab with proper interface circuits. This research work can be further extended with following up gradation. One more app and interface circuit can be developed for ECG. With more ADC channels all these studies can be made simultaneous. By having SIM card the service can be on the mobile network.

References
1. Aleksandra C. Zoric, SinisaS. Ilic, “PC Based Electrocardiography & Data Acquisition”, TELSIKS, IEEE, pp 619-622, September 28- 30 2005.
2. Tia Gao, Dan Greenspan, Matt Welsh, Radford R. Juang, and Alex Alm, “Real Time Patient Monitoring System Using Lab view”, International Journal of Scientific & Engineering Research, April-2012.
3. Sherin Sebastian, Neethu Rachel Jacob, "Remote Patient Monitoring System Using Android Technology”, IJDPS, September 2012. Prema Sundaram, International Journal of Computer Science and Mobile Computing Vol.2 Issue. 5, May-2013, pg. 191-201 © 2013, IJCSCMC All Rights Reserved 201.
4. C. Wen, M. Yen, K. Chang and R. Lee, “Real-time ECG telemonitoring system design with mobile phone platform”, Measurement, Volume 41, Issue 4, May 2008, Pages 463-470.
5. Wilkoff BL, Auricchio A, Brugada J, Cowie M, Ellenbogen KA, Gillis AM et al. “HRS/EHRA Expert Consensus on the Monitoring of Cardiovascular Implantable Electronic Devices (CIEDs): Description of Techniques, Indications, Personnel, Frequency Ethical Considerations”, Euro pace 2008;10:707–25.
6. Varma N, Epstein A, Schweikert R, Love C, Shah JA, Irimen " Evaluation of efficacy and safety of remote monitoring for ICD follow-up" the TRUST trial, Circulation 2008, Vol. 118, No. 22, 2316, Abstract 4078.
7. Kiely DK, “Resident characteristics associated with wandering in nursing home”, Int J Geriatric Psychiatry. 2000; 15(11):1013-1020.
8. K. Lorincz et al., “Sensor Networks for Emergency Response: Challenges and Opportunities,” IEEE Pervasive Computing, IEEE, Press, pp. 16-23, October-December 2004.
9. J. Hill et al., “System Architecture Directions for Networked Sensors,” in Proc. 9th Int’l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), ACM Press, pp.93-104, 2000.
10. K. Lorincz and M. Welsh, A Robust, “Decentralized Approach to RF Based Location Tracking, tech. report TR-19-04, Division of Eng. And Applied Sciences “, Harvard Univ., Cambridge, MA, 2004.