mDocInABox

Rattanaphan Boonbutra of Computer Science, Hood College.

Index Terms— Arduino UNO, Parallax BOF, BlueSMiRF, Amarino 2.0, Amarino plugin bundle, pressure sensor, the piezoelectric sensor, and the LM34 sensor, the heart rate receiver, the DHT22, and the ColorPAL sensor, Android (version 2.3.6), Android Development Kit.

Introduction

This project is very helpful for learning about the Arduino UNO board and Android app development. The Parallax BOE (Board of Education) is used with an Arduino UNO board. There are 3 components to this project. The first is hardware consisting of an Arduino UNO and a variety of sensors providing both analog and digital signals. Four of these sensors are analog such as the pressure sensor, the Piezoelectric sensor, and the LM34 sensor. Three sensors provide a digital signal such as the heart rate receiver, the DHT22, and the ColorPAL sensor. The second component is the connection between the Arduino UNO board and Android (version 2.3.6). Communication between the Arduino UNO board and Android OS is achieved using the Bluetooth "BlueSMiRF" Gold transceiver from Sparkfun. Amarino 2.0 and the Amarino plugin bundle were also used to transmit and receive the event and data via the Bluetooth. The last component is the Android app. The Android Development Kit was installed on top of Eclipse in order to develop the Android app.

This report will show the design of the application on Android 2.3.6 and also discuss the results, the problems and overall experience of building the project.

Table of Contents

1.0. Diagram and Connection

Figure 1 shows how the array of sensors, the BlueSMiRF Bluetooth, and the LCD2004 are connected to the Parallax BOE on the Arduino UNO board. There are seven sensors connected to the Parallax BOE. The sensors are connected as follows:[1]

1. Pressure sensor (analog signal) to A0
2. Piezoelectric sensor (analog signal) to A1
3. LM34 temperature sensor (analog signal) to A2
4. SpO2 Oxygen sensor (analog signal) to A3
5. ColorPAL sensor (digital signal) to D2
6. DHT22 temperature and humidity sensors (digital signals) to D3
7. Heart Rate receiver (digital signal) to D4
8. LCD2004; SDA pin to A4 and SCL pin to A5
9. BlueSMiRF; tx pin to D0 (rx) and rx pin to D1 (tx)

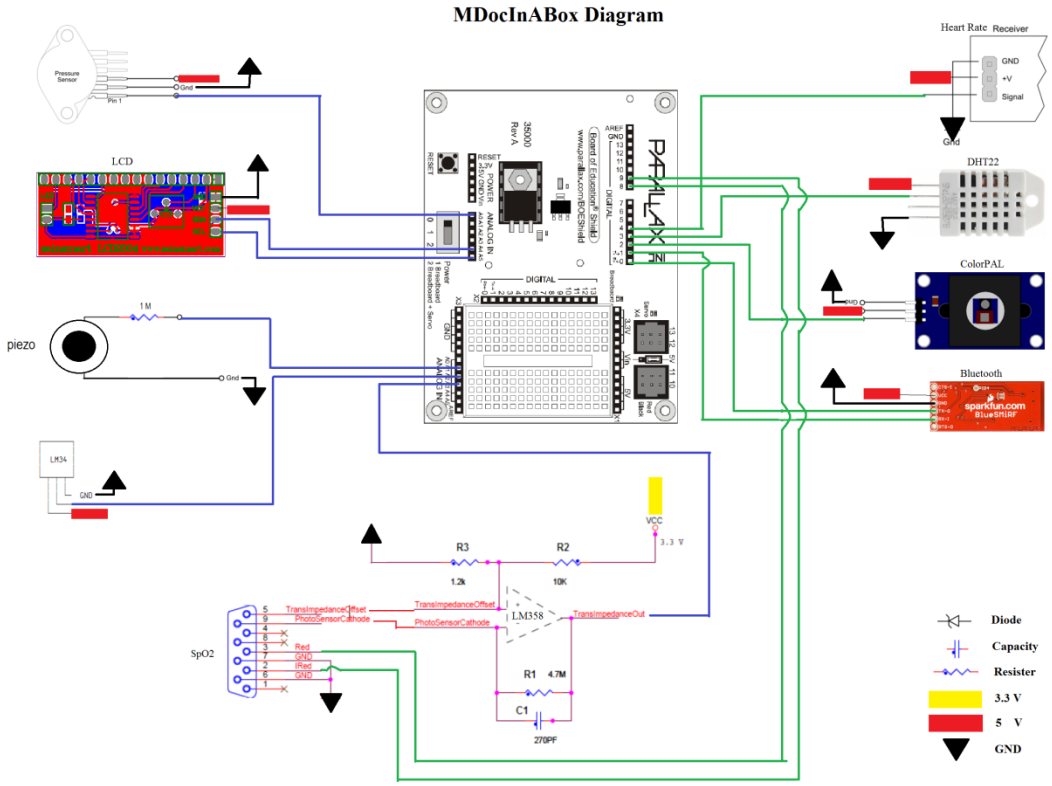

Figure 1

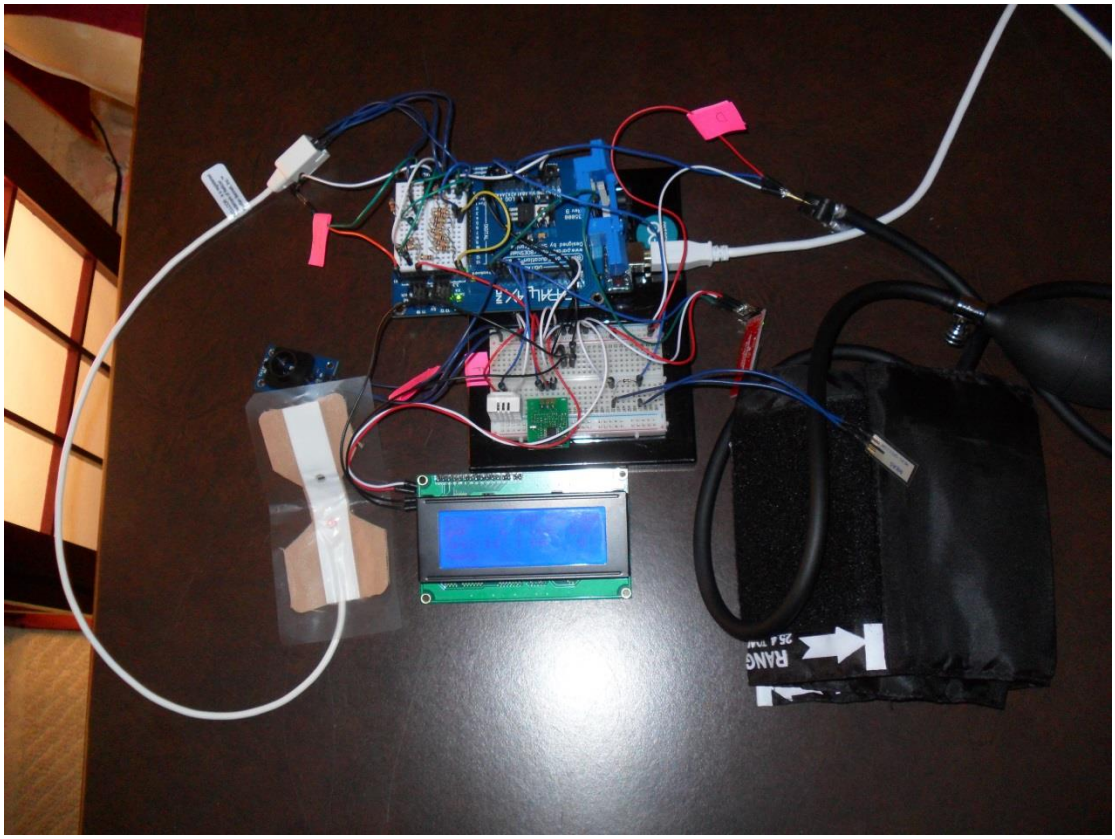

Figure 2: The board with all connections made per Figure 1.

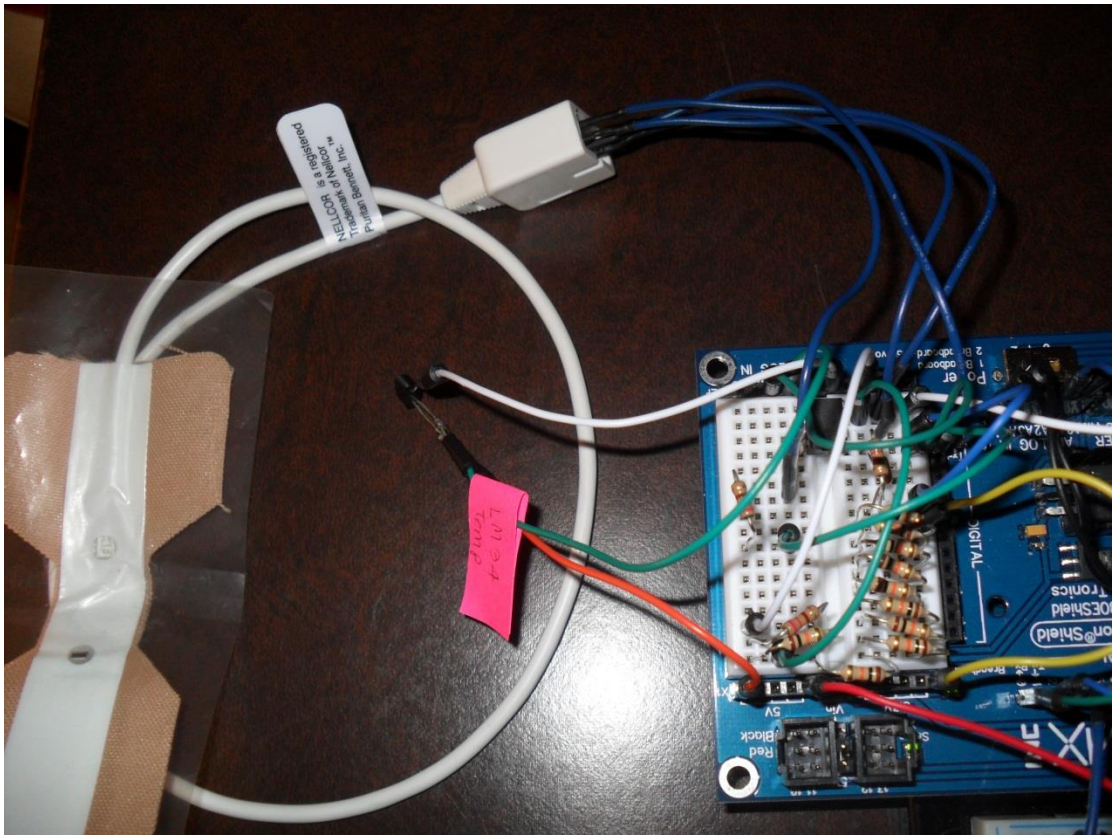

Figure 3: Shows the SpO2 finger sensor connected to the Parallax BOE.

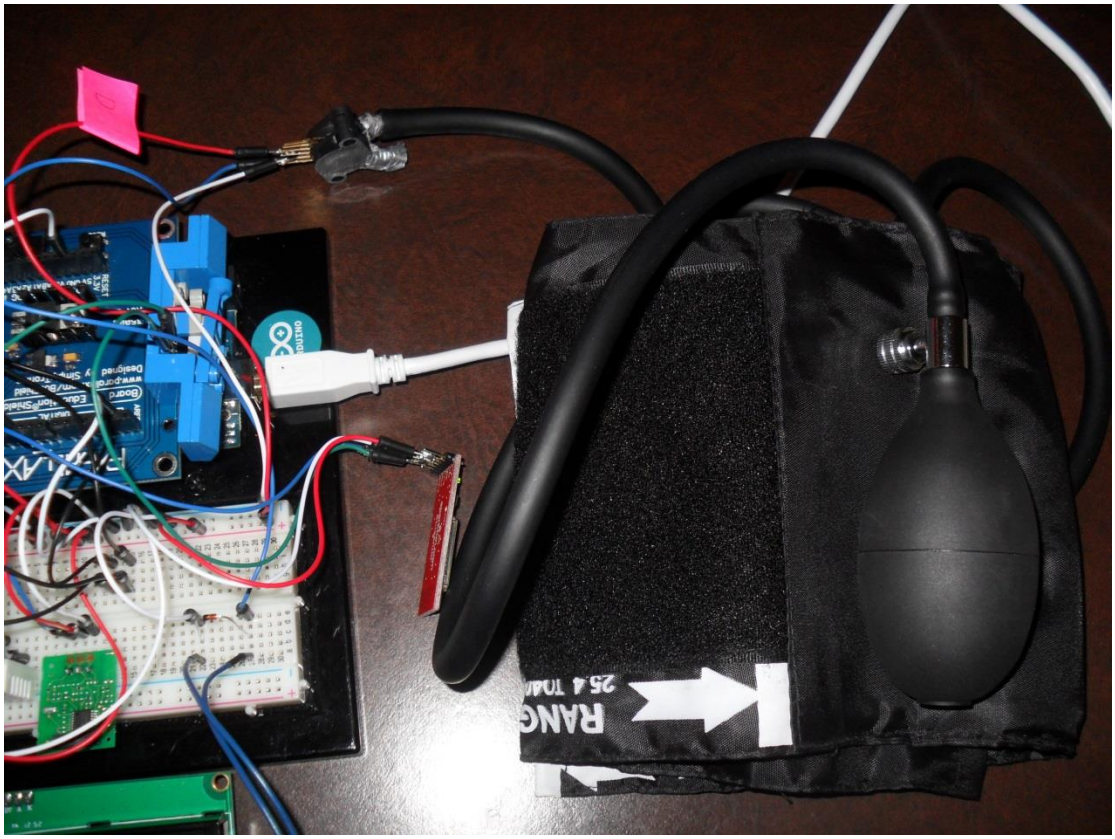

Figure 4: Shows the sphygmomanometer and cuff connected to the pressure sensor.

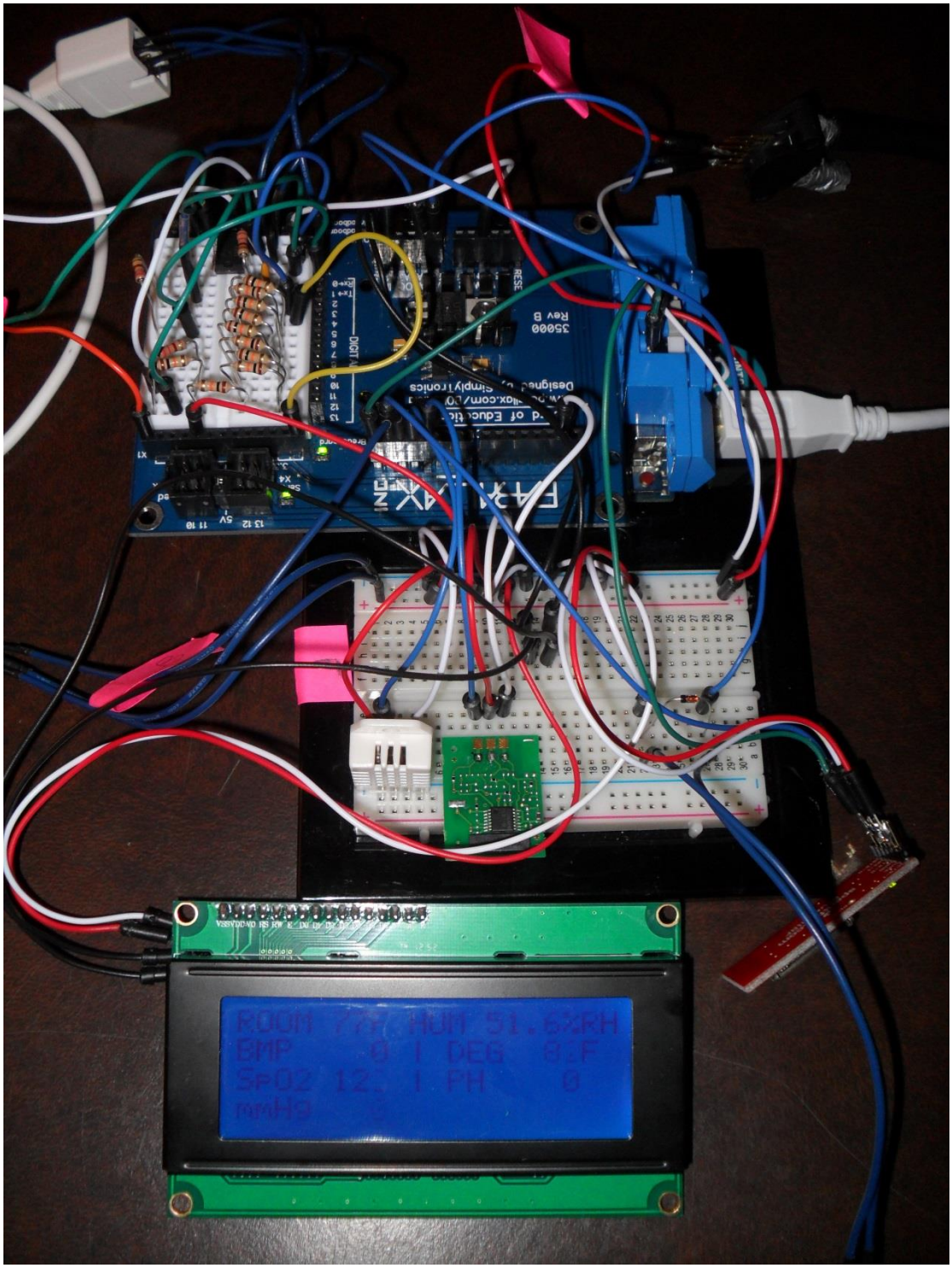

Figure 5: Shows LCD2004 connected to the Parallax.

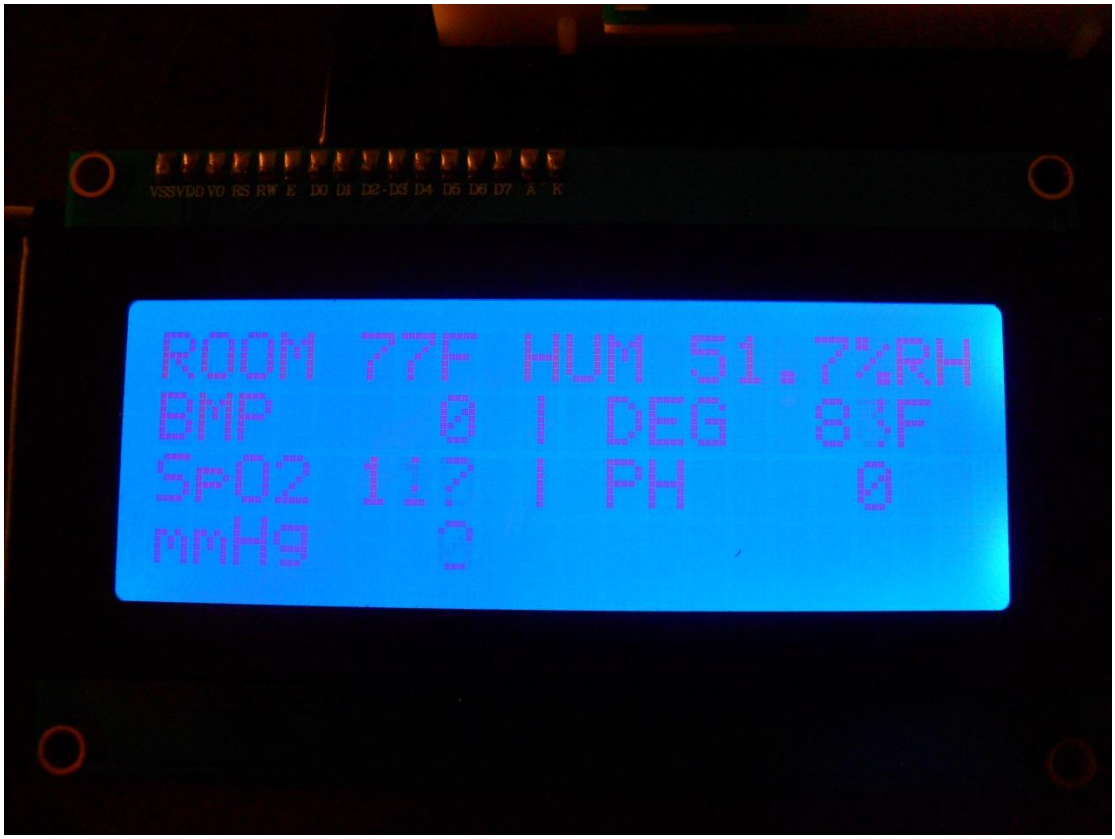

Figure 6: Shows the result of all sensors on the LCD2004

2.0. Mobile Application

The mDocInABox application was developed for the Android 2.3.6 OS. The app consists of the following screens:

1. Splash Screen
2. Home\menu screen
3. Heart Rate screen
4. Health Dashboard screen
5. pH Balance screen
6. Oxygen screen
7. Blood Pressure screen
8. Temperature screen

Moreover, the application consists of nine classes which are reviewed later in this document:

1. GraphView
2. HealthDashBoard
3. HeartRate
4. Home
5. Oxygen
6. pH
7. Pressure
8. Splashscreen
9. Temperature

The home screen serves to display a menu which links to the other screens. Every screen shows the room temperature and humidity at the top. The Heart Rate screen shows the output from the Heart rate receiver. There are two graphs on this screen. The graph below shows each heartbeat as a spike from 0 to 1. The BPM (beats per minute) graph shows the current beats per minute. The Health dashboard screen shows every value from the Arduino board such heart rate, temperature, blood pressure, oxygen and pH balance. This screen also has buttons at the bottom which can be tapped to save or synchronize the data with the health center database. The pH balance screen is very easy to read. It shows the value of the pH from the ColorPAL sensor. The Oxygen screen has a graph that shows the level of oxygenation from the SpO2 finger

sensor. The blood pressure screen has two graphs. The top graph shows the value from the Piezoelectric sensor which is used to detect bloodflow. The second graph monitors the pressure from the pressure sensor. The value of SYS and DIA is determined by analyzing both graphs. The splash screen shows when the mDocInABox app is started. The last screen is the Temperature screen. This shows the value from the LM34 sensor.

Figure 7

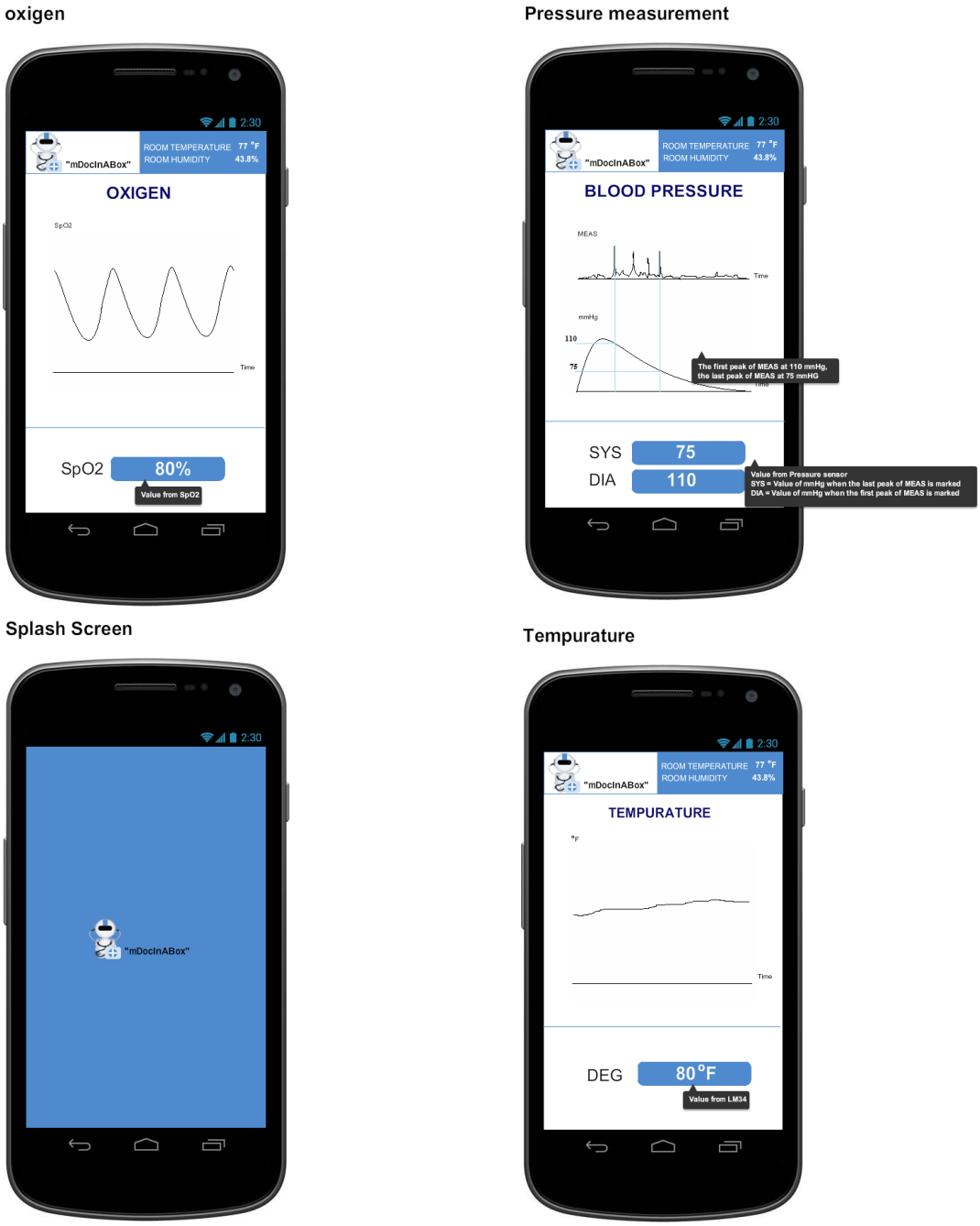

Figure 8

Note: Link to mobile design application for Android :
http://app.mockflow.com/view/859c0b2600acc2c0ce09fb0a16dc52c1

3.0. Results, Problems and Experience

In this section we discuss results and problems that were encountered. The data that has been compiled across several tables in this section show problems that were observed while operating the equipment. All of the results were captured from the Android application running on a Nexus One smartphone. Screenshots were captured using ddms.bat, a tool provided by the Android SDK.

In order to use this application find the mDocInAbox in the Android app drawer and tap it once. The splash screen can be observed on startup of the app. These steps are illustrated below.

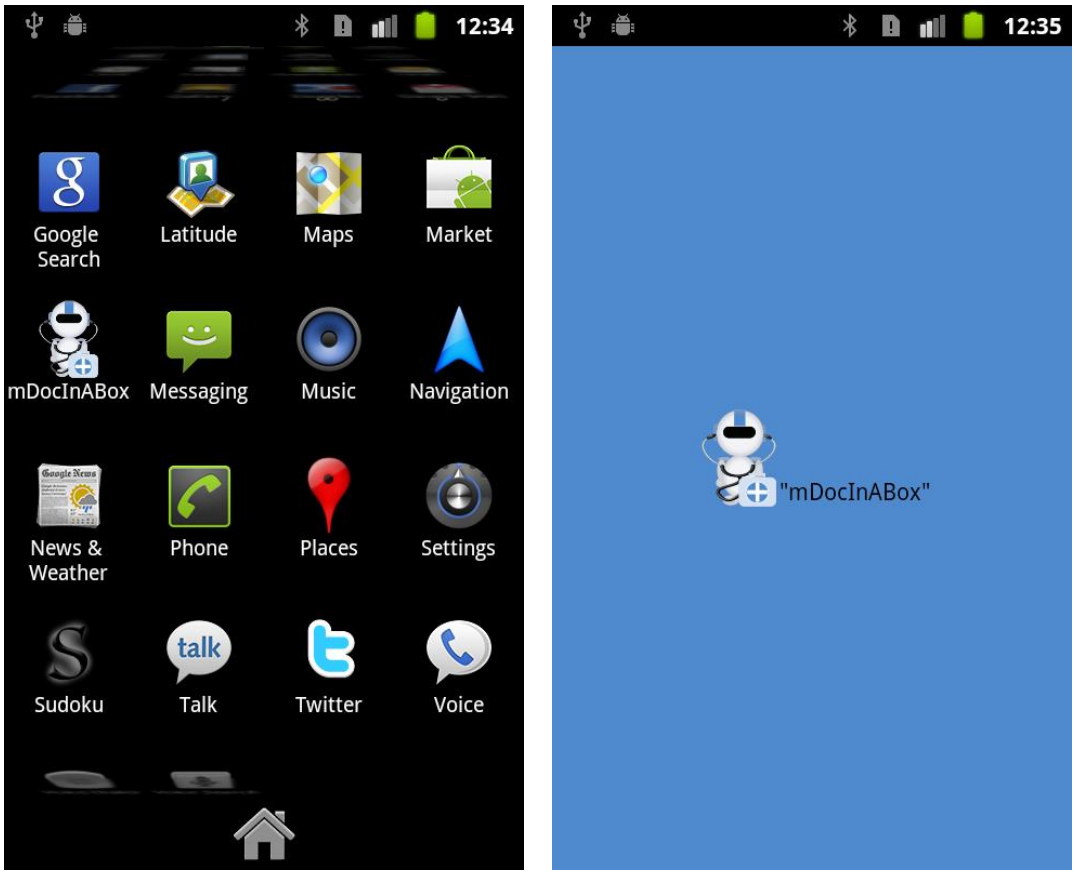

Figure 9: On the left, shows the icon of mDocInABox on the mobile home screen. On the right, shows the splash screen.[10]

3.0.1. Home screen capture result.

The home screen shows up after the splash screen disappears. The user is presented with a list of options that can be tapped to begin reading data from the appropriate sensor(s).

3.0.2. Health Dashboard screen capture result.

The Health Dashboard screen serves to summarize data from several sensors simultaneously and presents them in a clear and succinct interface.

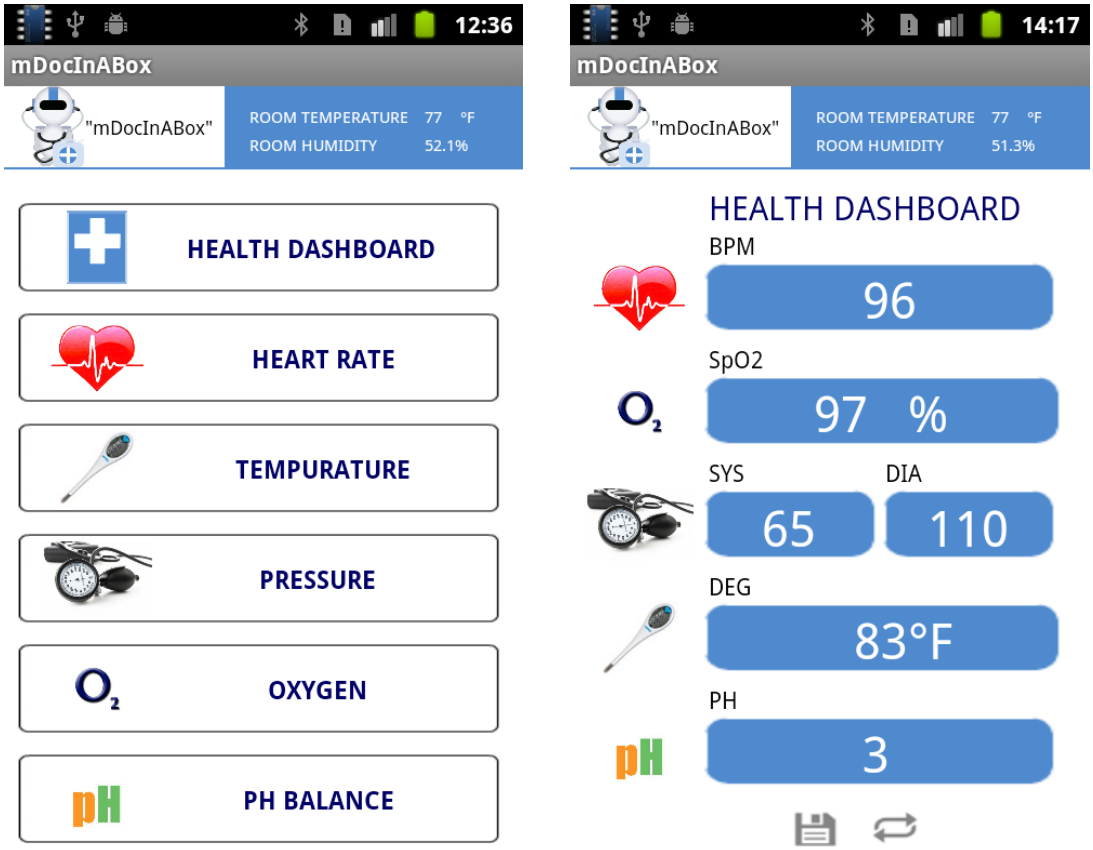

Figure 10: On the left, shows the capture of Home Screen. On the right shows the capture of Health Dashboard Screen.

3.0.3. Heart Rate screen capture result.

As shown in Figure 11, the value of BPM is not correct. As a result of not being able to detect every heartbeat (see the "Beat" graph), the patient's BPM cannot be accurately calculated.

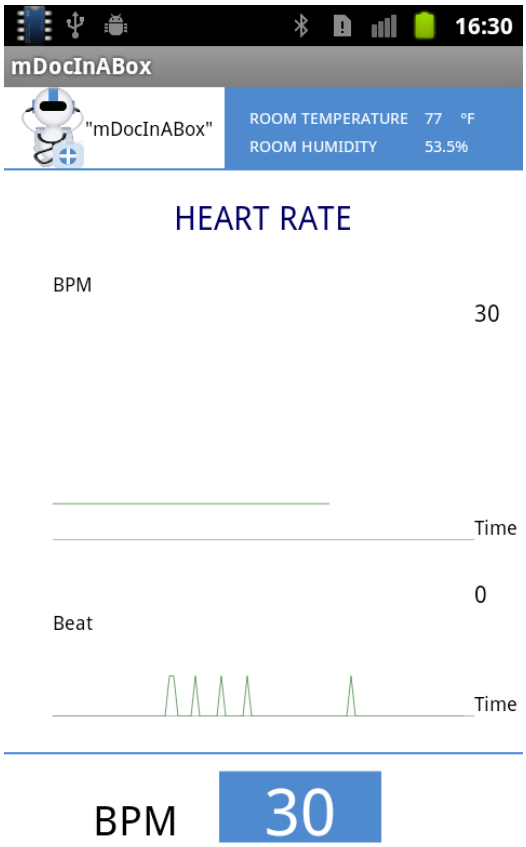

Figure 11 : Real-world Metrics

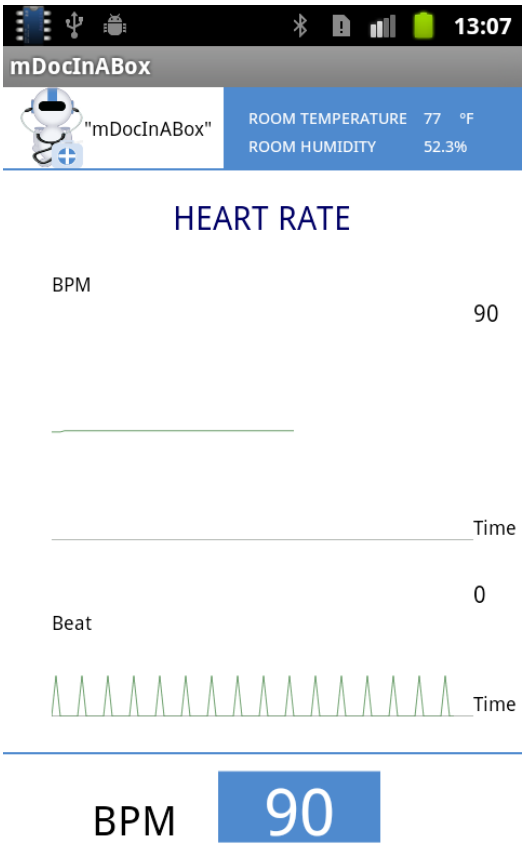

Figure 12 : Ideal, artificially generated metrics

$$BPM = 1 / (BCT-BPT) * 6{,}000$$

BPM = Beat Per Minute

BCT = Time[§] of last heartbeat

BPT = Time[§] of previous heartbeat

[§] Time in millisecond, so multiple by 6,000

Result Table 1

Result as displayed on	Data Transmit Method	BPM Accuracy
1. LCD screen	Multiple at Once	Accurate
2. LCD screen	Single	Accurate
3. Amarino 2.0	Multiple at Once	Inaccurate
4. Amarino 2.0	Single	Accurate
5. mDocInABox App	Multiple at Once	Inaccurate
6. mDocInABox App	Single	Accurate

It can be observed from Result Table 1 that when the result is output to the LCD screen (directly connected to the Arduino board) the BPM is reported accurately whether the software is sending multiple sensor values at once or only the values from the heart rate sensor. On the other hand, when Arduino software is configured to transmit the values of multiple sensors at once to the smartphone, the integrity of the data suffers.

3.0.4. Temperature screen capture result

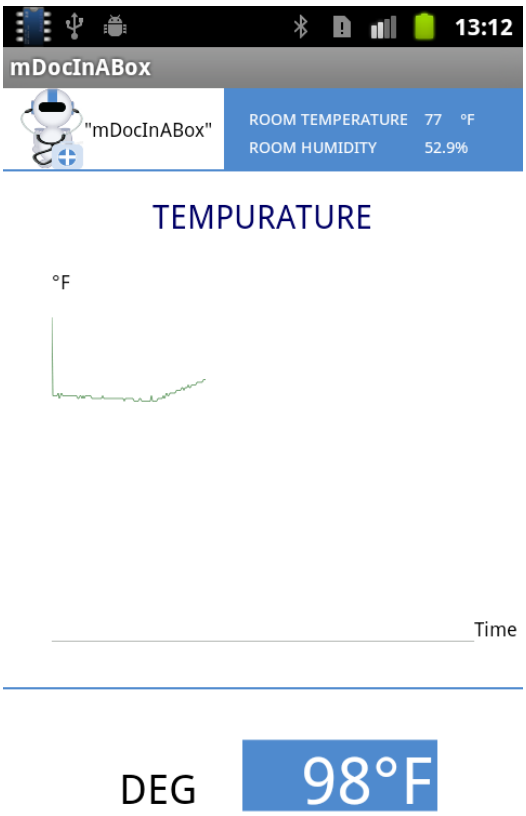

Figure 13

The value from the LM34 sensor can be used to measure temperature. The sensor may be used in temperatures ranging from +32 to +212 °F. The sensor is accurate to 1.0 °F at +77 °F.

DEG = Analog signal from LM34 – 77

3.0.5. Pressure screen capture result.

Figure 14 : Real-world metrics

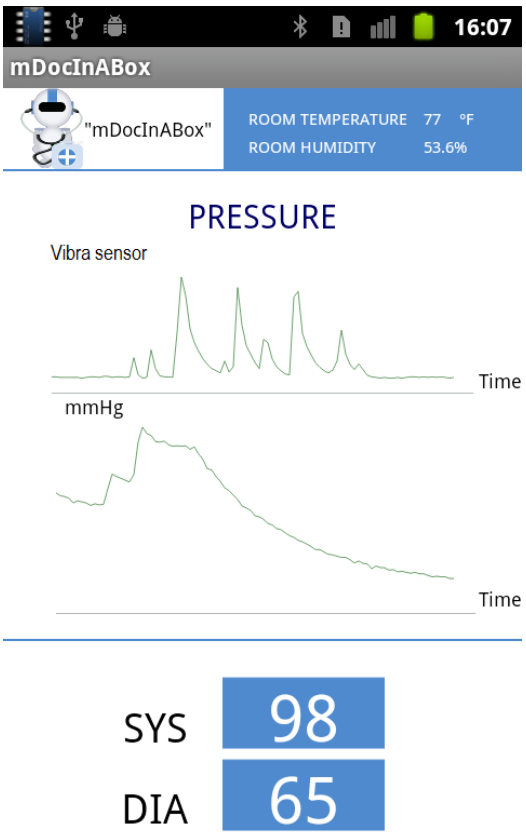

Figure 15 : Ideal, artificially generated metrics

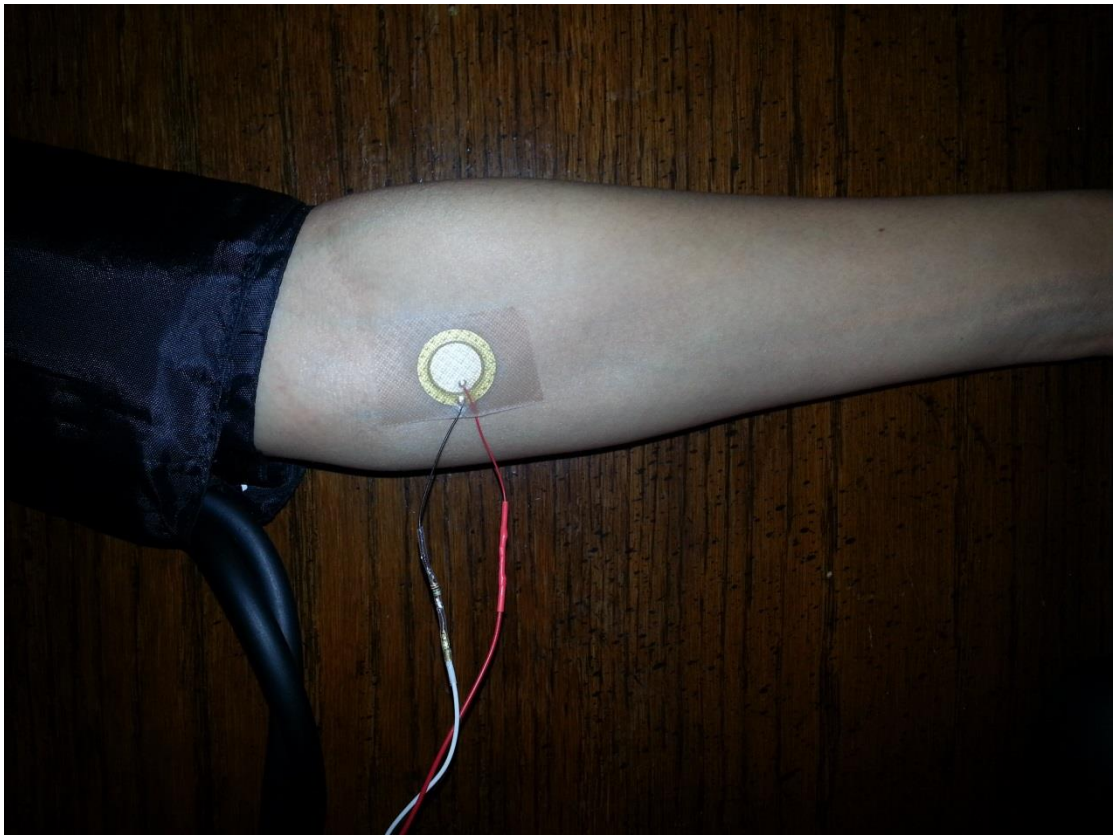

Figure 16

Blood pressure is measured by two sensors: the Piezoelectric sensor and the pressure sensor as shows in Figure 16. Figure 17 demonstrates how the DIA and SYS values can be determined from the analysis of data returned by these two sensors. The first step is to inflate the pressure cuff until the pressure is high enough and the Piezoelectric sensor gives a reading very close to zero. This indicates that the artery is closed. Second, the air pressure is decreased by deflating the cuff. The moment the Piezoelectric sensor detects a vibration (bloodflow) the pressure at this event is recorded. This event marks the value of SYS. The program will continue to monitor activity from the vibra sensor until the last peak. The value of DIA will be the value from pressure sensor at the last peak of Piezoelectric sensor.

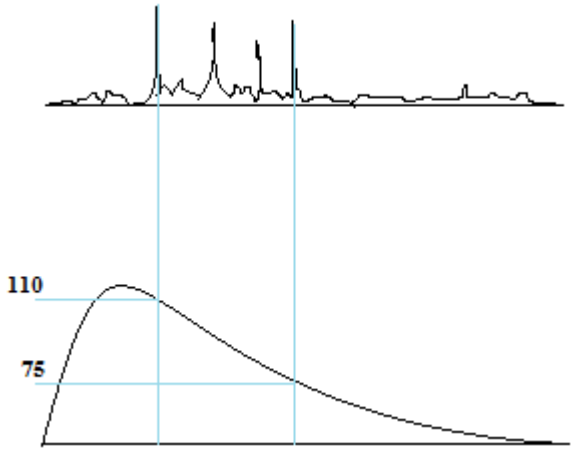

Figure 17 : SYS is 110 mmHg and DIA is 75 mmHg.

There is a problem as shown in Figure 14. The vibration graph (activity from the vibra sensor) shows that the vibra sensor is very sensitive. This leads to the introduction of artifacts in the vibration graph making detection of actual bloodflow very difficult. Due to this issue, values for SYS and DIA are incorrect. Ideally, the vibration and pressure graphs should look more like Figure 15. There are 4 peaks in this vibration graph. The first peak occurs when the pressure sensor reports 98 and the last peak occurs when the pressure sensor reports 65. This result agrees with that obtained by using the analog meter.

Results Table 2

Result as displayed on	Data Transmit Method	Sensor Accuracy
1. LCD screen	Multiple at Once	Accurate
2. LCD screen	Single	Accurate
3. Amarino 2.0	Multiple at Once	Inaccurate
4. Amarino 2.0	Single	Accurate
5. mDocInABox App	Multiple at Once	Inaccurate
6. mDocInABox App	Single	Accurate

Results Table 2 shows that when transmitting multiple sensor data to the smartphone, data integrity is lost.

Observations

Source	Result
1. Actual Sensor	The sensor is highly sensitive and values vary greatly.
2. Generated Piezoelectric sensor	The results are consistent with the analog method.

According to my observation the Piezoelectric sensor is not an appropriate sensor for this application.

3.0.6. Oxygen screen capture result.

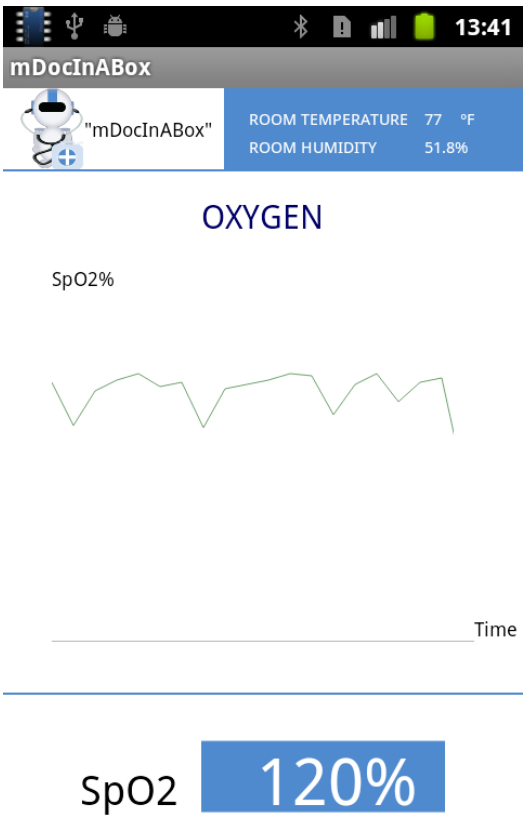

Figure 18 : Real-world metrics

There is a problem as shown in the Figure 18. The SpO2 graph demonstrates a loss of data. The expected result of is illustrated below.

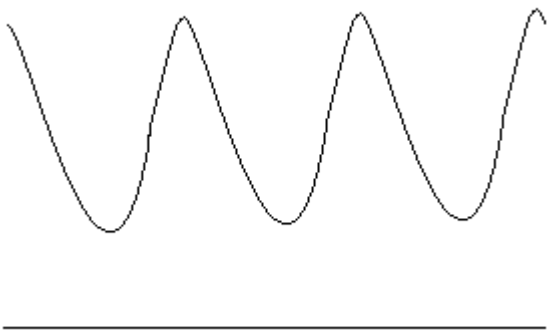

Figure 19: Ideal, artificially generated metrics

Results Table 3 : Sample results of SpO2 from Serial.println vs. Amarino 2.0 with multiple sensors

Serial.println	Amarino 2.0
1 98	2 100
2 100	5 107
3 102	6 109
4 104	11 115
5 107	13 120
6 109	18 110
7 112	19 108

Results Table 3 shows that data loss was experienced during transmission via Bluetooth when multiple sensors are polled simultaneously.

3.0.6. pH Balance capture result

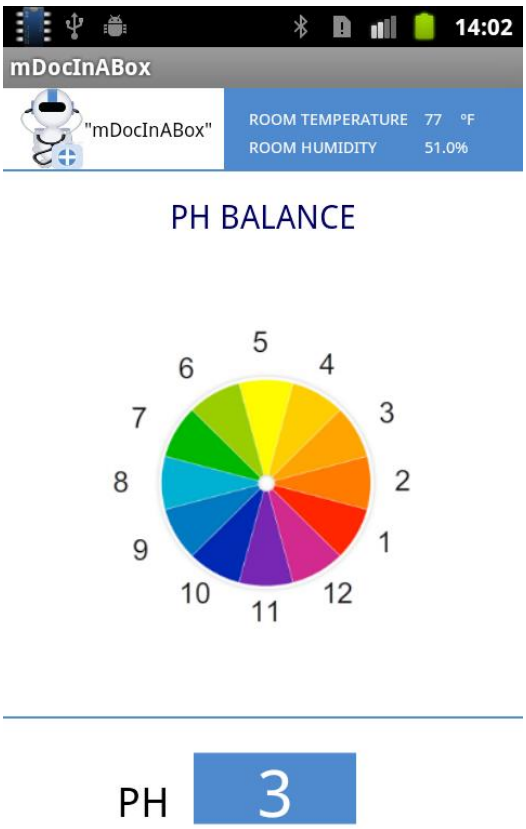

Figure 20

The ColorPAL sensor is used to indirectly measure the pH of substances by analyzing the color of litmus strips. The ColorPAL sensor returns data in the form of RGB values. In order to write software that would be able to accurately report pH upon visual inspection of a litmus strip, RGB thresholds were identified between individual pH colors. An example of this code appears below.

```
if(red>250 && red<260 && grn>80 && grn<90 && blu>100 && blu<120)
  {
   ph = 2;
  }
  else if(red>240 && red<260 && grn>90 && grn<100 && blu>120 && blu<130)
  {
    ph = 3;
  }
```

4.0 Improvement

1. The problem from the MEAS vibra sensor is solved by using piezoelectric sensor instead.

2. The code in mobile application was fixed. There is missing a break at switch case in pressure.java.

5.0 The result

1. The piezoelectric sensor can measure the bloodflow very well. Piezoelectric sensor is not too sensitive like MEAS vibra sensor. It is suitable to use to measure bloodflow.

2. Fetching data problem on the page pressure measurement is solved.

6.0. Summary

Based on observation and testing it appears that any screen of the mDocInABox Android app which is designed to display the output of multiple sensors demonstrates a problem. It can be deduced that these errors are introduced in the transmission of data between the Arduino board and the smartphone. Instead of creating a virtual serial port for each sensor, the data for several sensors was concatenated into a specially formatted string and transmitted to the app at a predetermined interval. This method was believed to provide the best synchronization between sensors but introduced the unwelcome artifacts which corrupted many sensor readings.

In order to improve the reliability and accuracy of this product a virtual serial port should be created per sensor to keep each data stream on its own port. The Arduino sketch would then have to be updated to provide better polling of the sensor data across all sensors.

7.0. Conclusion

Working on this project and developing the final product helped me to learn how to apply the Parallax BOE on Arduino UNO boards. I discovered the Parallax BOE to be very capable in satisfying the application of various medical sensors. On the other hand, sending the signal to Android via Bluetooth caused with some problems. According to my observations a more robust method of handling the output of several sensors at once needs to be designed before implementing the sensors otherwise the quality of the collected data will prove to be insufficient for medical purposes.

Acknowledgements

Thanks to Dr. George Dimitoglou, Professor of Computer Science at Hood College, MD.

Reference

[1] microMedic Kit Demos, [Online]. Available: http://learn.parallax.com/micromedic/kit-demos

[2] Getting Started w/ Arduino on Windows , [Online]. Available: http://arduino.cc/en/Guide/Windows

[3] How to connect my phone to Arduino in 10 steps , [Online]. Available: http://www.amarino-toolkit.net/index.php/getting-started.html

[4] Setting Up the ADT Bundle, [Online]. Available: http://developer.android.com/sdk/installing/bundle.html

[5] Building Your First App, [Online]. Available: http://developer.android.com/training/index.html

[6] Installing the Eclipse Plugin, [Online]. Available: http://developer.android.com/sdk/installing/installing-adt.html

[7] Chapter 6 Amarino Software Toolkit, [Online]. Available: http://www.amarino-toolkit.net/tl_files/thesis/amarino_thesis_kaufmann_2010.pdf

[8] K. Bonifaz, Design and Implementation of a Toolkit for the Rapid Prototyping of Mobile Ubiquitous Computing, August 2010, [Online]. Available: http://www.amarino-toolkit.net/tl_files/thesis/amarino_thesis_kaufmann_2010.pdf

[9] Technical Information - National Semiconductor LM358N Datasheet, [Online]. Available: http://www.futurlec.com/Linear/LM358N.shtml

[10] How to Capture Screenshots on the Google Nexus One, [Online]. Available: http://news.softpedia.com/news/How-to-Capture-Screenshots-on-the-Google-Nexus-One-136278.shtml