Final Report: Wireless Electro-cardiogram Monitor

Submitted as a requirement
for the course ICOM5047: Section 030
Design Course for B.S. in Computer Engineering Department
University of Puerto Rico at Mayagüez, PR

Students:

Bermudez Soto, Wilfredo

Ortiz Perez, Alexis M.

Vega Diaz, Rafael

Rivera Suarez, Melvin

Faculty Advisors:

José Fernando Vega, PhD
Associate Professor

Nayda G. Santiago Santiago, PhD
Associate Professor

Kejie Lu, PhD
Assistant Professor

Date: December 9, 2008

Contents

1. *Executive Summary*

In order to help our society and use technology to improve the quality of life, WARM team has taken on the task of designing a portable electrocardiogram monitor called the WARM Wireless ECG Monitor. Millions of people worldwide suffer from heart diseases and some even need to be monitored all the time. These patients would be at ease knowing that a device is taking care of monitoring their heart status, ready to send for help in case of a cardiac emergency. The main objective of this project is to design a system capable of monitoring heart without limiting the patient to a hospital or his home.

WARM Wireless ECG Monitor System was designed to be light weight, small and portable. To provide emergency communication, the WARM Wireless System has an Integrated Bluetooth that is capable of communicating with a cell phone. A cell phone application will provide the user interface for data display and sending alert messages.

The estimated cost of the system was $102,516.65 and the final cost was $96,399.20. This results in a minimal difference due to the estimated consultant hours and the actual hours needed.

WARM team's objective was to design a device that satisfies all potential customer needs. The development of a prototype tried to implement ways to address these needs. Due to different constrains, discussed later on that document, our prototype was not able to meet the requirements for all patient properly. The solutions to these problems are also part of this document.

2. Introduction

The number of deaths in United State caused by heart diseases was 452,000 in the United States in 2007. That represents one of every five deaths in the whole nation. With that problem in mind, WARM team along with Dr. Kejie Lou proposed to design a wireless monitor system that has the versatility to monitor the heart and in case of some anomaly, send an alert message to a family member or any other person that you have programmed.

The main problem with ECG monitoring is that the patient needs to be plugged to a machine that is stationary in a hospital or house, limiting the patient to stay in that area. Another important issue is the huge cost on hospitalization expenses.

The ECG signal is the most common way doctors use to analyze heart status. It is not an easy task that requires a great knowledge of cardiology and signal processing. With that in mind, WARM ECG monitor has the goal to help this task by analyzing the signal and provide doctors with the data they need to keep a patient healthy.

2.1. Document Description

This document presents a description of the work done designing the WARM Wireless ECG. It details the hardware and software design and specification criteria. Constrains, limitations and changes on the planed solution are also explained and justified. Market description, budget analysis, final results are detailed, and finally a section on future work explains the recommended next steps to follow in order to solve different constrains encountered during the design.

3. Design Criteria and Specifications

This section gives brief description of the design specification for detailed and technical details refer to the hardware design document or software design.

3.1. Electrodes

The main design considerations for the ECG electrodes were that they should be wearable, safe for human use and reliable for sensing the desired signals. We also had to consider the placement of the electrodes in order to make the device comfortable for the patient and to know what type of signal we would be getting.

3.2. Analog Circuitry

The electrical signal generated by the heart must be amplified to an appropriate level for sampling. This requires an amplifier with high gain. In order to bring the signal to a level close to 1 V, a gain ranging from 500 to 1000 is desired. Since the signal is measured differentially, the amplifier should also reject body signals common to both input terminals. The common mode rejection ratio (CMRR) of the amplifier should be greater than 80 dB to reject the undesired common signals.

After researching various amplifiers and circuit designs, we realized that an instrumentation amplifier could provide the desired gain levels with a high CMRR. The AD624 is a high precision, low noise, instrumentation amplifier with high gain accuracy, low gain temperature coefficient and high linearity used in high resolution data acquisition systems. It offers input noise levels below 4 nV/√Hz at 1 kHz and pin-programmable gains of 1, 100, 200, 500 & 1000 provided on the chip. This amplifier also has a CMRR above 80 dB at unity gain, which helps reject other common mode signals the electrodes will pick up.

To provide electrical isolation between the patient and the rest of the device, an opto coupler was used. The PS2506 opto coupler was selected due to its low cost and availability. To electrically insulate the AD624 output signal from the rest of the circuit, it is connected to the output of the AD624 and the output signal is then fed to the MCU's internal analog to digital converter (ADC). The internal circuit and the external terminal unit are electrically isolated from the measurement electrode of the electrocardiograph to prevent the occurrence of an electric shock accident. Also a 741-Opamp is used to offset the output DC voltage by connecting it to the reference node of the AD624. The 741 is connected as a buffer and provides low impedance at the reference terminal. This helps keep the CMRR at the desired level, since impedance at this terminal reduces the CMRR by 10 kΩ/R_{REF}. It can also provide some electric safety, because the auxiliary op amp will saturate when an abnormally high voltage appears between the patient and ground. The UA741 was selected because of its initial availability.

3.3. Microcontroller

To select a microcontroller (MCU) for our design, we have to consider functionality, performance, size, cost, and power consumption. Our design requires us to send the data over to a monitoring device. This means that we need to interface the MCU with a wireless communication module. We also decided to process the signal within the MCU, so performance is also an important issue. To make our device wearable and portable, the size of the MCU must be as small as possible, and its power consumption must be minimal.

With these criteria in mind, we selected the MSP430F149. First, the MSP430F149 was designed with low power consumption in mind and low cost, which is one of the defining criteria for any portable device. It can support clock rates up to 8MHz. It has 60 KB of flash ROM for code, and 2 KB of RAM for data. It also has two 16-bit timers, a 12-bit ADC with internal reference, sample and hold, auto scan feature, two DAC outputs, two UART (serial) ports, and 48 General Purpose IO (GPIO) pins. Note that multiple functions may be assigned to one pin, so not all features are available at a single time. The function and direction of each pin must be set in all programs. Unused pins should be set as GPIO outputs.

Texas Instruments microcontroller MSP430F149 is the main component of the design. It will be responsible of all the needed calculation of the ECG wave form and also provides the necessary synchronization of the system. Connections between analog circuit and microcontroller will be done via the pin connection of the ADC converter that is part of the microcontroller. This ADC has 12 bits and a conversion rate of 200 ksps (kilo sample per second), that fully satisfy our need sampling rate that is between 200 and 500 sample per second. To connect the Bluetooth module we are going to use the UART interface that is supported for both, the microcontroller and the Bluetooth module. Finally, the MSP430F149 is already available at the laboratory and has full software support.

3.4. *Peak Detection Algorithm*

The heart produces a very peculiar signal which represents the different phases the heart goes through to pump blood to the body. The signal mainly consists of five different peaks known as P, Q, R, S, T from which different parameters are calculated.

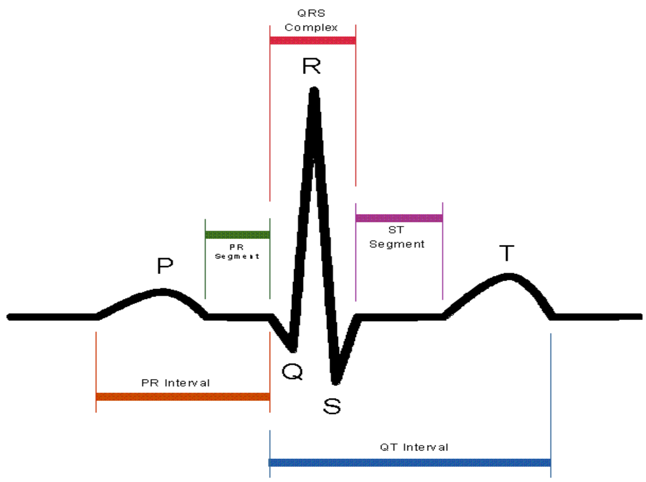


Figure 1 Ideal heart ECG wave.

Through the analysis of the wave and detection of the peaks one can calculate the following parameters:

- ✓ PR Interval - The time elapsed from the left bound of the P peak to the left bound of the Q peak.
- ✓ QRS Complex – The time elapsed from the left bound of the Q peak to the right bound of the S peak.
- ✓ QT Interval – The time elapsed from the left bound of the Q peak to the right bound of the T peak.
- ✓ QTc – QT Interval corrected.
- ✓ Vent Rate – The time elapsed between two R peaks.
- ✓ T Amplitude – The height of the T peak.

To detect the needed peaks we used the look2future algorithm. The algorithm works by setting a predefined step size, stepping through the signal and calculating the difference between the start of the step and the end. The purpose is to detect deflections[1] comparing the difference to a given threshold. By fine tuning the threshold and the step size, the algorithm can make an educated guess where in the sampled data might be a peak.

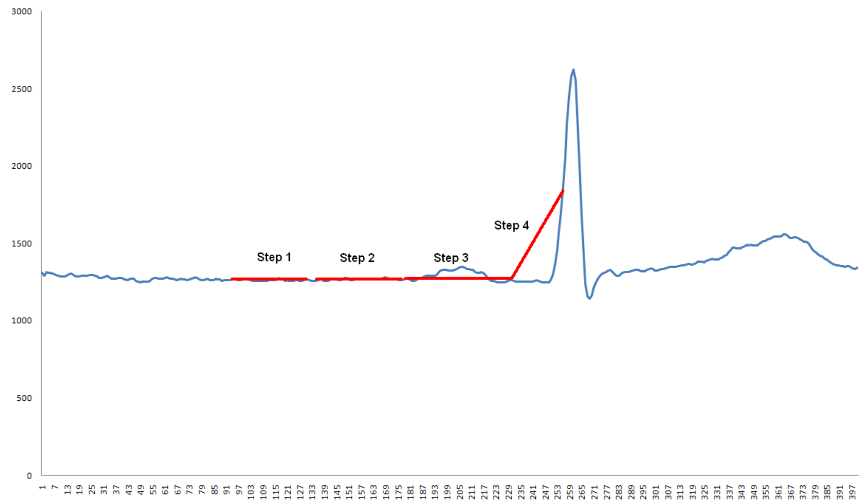


Figure 2 Look2future steps illustration

After detecting a deflection, the algorithm continues to find the maximum height of the peak by reducing the step size. When it detects that the steps have a negative difference it presumes that it has reached the maximum and then proceeds to find the peak boundaries. The boundaries are found in a similar manner iterating left and right and looking for the lower values.

In a first pass the algorithm is capable of detecting the R peak. Then the algorithm's parameters are adjusted for two more passes that detect the P and S peak. Code implementation for the MSP430Ff149 can be seen in APPENDIX B: Peak Detection Code.

[1] A turning aside or deviation from a straight line.

3.4.1. Algorithm Pitfalls

The algorithm seems very straight forward and simple, but still in the real world it can face some difficulties detecting the peaks. After the parameters are set the algorithm expects a certain signal form. If the signal contains too much noise it can detect false peaks. Given the nature of the ECG signal being monitored, its characteristics are constantly changing due to variables like the patient’s mood, blood pressure, among other possible factors. These factors require another layer of software to be added that adapts the detection parameters depending on the current signal changes.

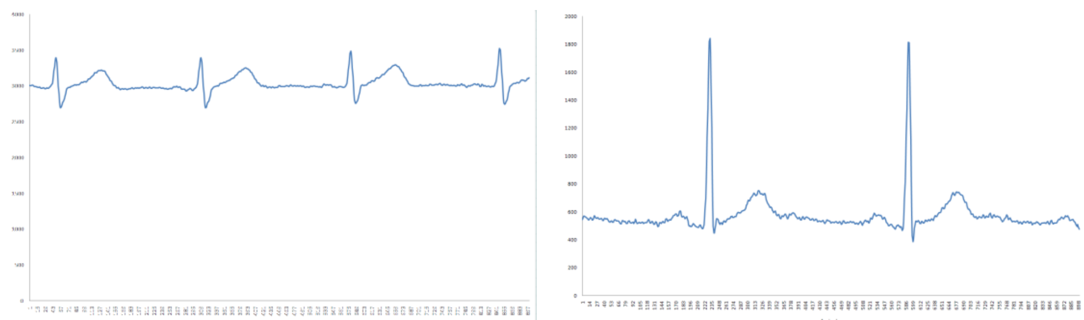


Figure 3 Two ECG Signals from different patients.

The need for this extra layer of software is also observed when analyzing different patient’s ECG signals. For instance one person might have a P wave while a different patient might not. ECG signals from patient to patient might defer so drastically that trying to set a universal step and threshold for the peak detection might be impossible.

3.4.2. Alternative Solutions

The MSP430F247 is a counterpart microcontroller to the MSP430F149. It has a faster processor running at 16MHz, twice as much memory, and consumes less power. The added memory could allow having the extra software needed to constantly adapt the peak detection parameters. Software performance is a most for such applications, optimizing the code could also improve the time needed for processing the signal. Optimizing the signal filtering circuit would reduce the need of software filtering reducing processing time from the micro-processor. Implementing an auto-zero circuit at the output of the monitoring circuit might also reduce unwanted signal offsets which might drastically change the peak detection parameters.

3.5. Wireless Communication Module

The wireless technology selection is a factor that completely changes the design of the monitoring device. Our main concerns are power consumption and complexity, but data transmission rate and operating range must also be taken into account. Since our product is designed to be wearable, the operating range can be 10m or less. The data transmission rate was dependant on the nature of the information to be transmitted to the monitoring device. Two technologies were suggested by Dr. Kejie Lu. Bluetooth and Zigbee were taken into consideration []. Bluetooth is an already established technology developed for short range communications as an alternative to cable connections. It is widely used in the cell phone industry and it has been making an entrance into the consumer device market. A Bluetooth device's range and power consumption are determined by its class. A device can reach up to 100 m @ 100 mW.

Zigbee is a similar technology intended for mesh networking. It has a greater signal range with less power consumption than Bluetooth, reaching 165 m @ 30 mW, but it has a lower data transmission rate. Its main use is for wireless mesh networks between multiple devices. The team decided to use Bluetooth technology to communicate with a mobile phone capable of executing a monitoring application instead of having to develop two separate devices with Zigbee technology.

For the wireless device subsystem two alternatives were taken into consideration. Wi-Fi is a well established protocol in today's market and in metropolitan areas are widely available. This kind of networks are relative easy to deploy for places that do not already have one like hospitals. Mobile phones are a more mature technology and more common than Wi-Fi. It has also a much greater signal range than Wi-Fi thanks to the great investment done by communication companies. Programmable mobile phones with internet access and Bluetooth enabled are easily found in today's market making it the choice alternative for our design.

Originally, we had selected KCWireFree® KC21 Bluetooth module, which met our design criteria. However, due to problems with its availability, we were forced to select another module which closely resembles the KC21. Bluegiga® WT12 was selected as our final interconnection Bluetooth module. This module is a class 2, Bluetooth® 2.0+EDR (Enhanced Data Rates) module. That means that it has low power consumption (2.5 mW), approximate range of 10 meters, and three times fast data rates compares to Bluetooth® 1.2. It is a highly integrated module with a build in antenna. The dimensions are 14mm x 25mm that make it a smaller chip in order to integrate to the system. WT12 is equipped with iWrap firmware, an easy to use firmware that provides a functional access to Bluetooth with ASCII commands send via serial UART (universal asynchronous receiver transmitter) interface. This module also provide 6 pins that could be programmed to do some of the iWrap commands when there occur a transition, this transition could be programmed to be low to high or high to low. This pins also can be configures as output of an occurring event like connection successful made.

3.6. Mobile Phone Application

Wireless devices can be programmed using several development environments directed to specific devices in the market. One of the technologies taken in consideration was the BREW [7] technology developed by Qualcomm. BREW is limited to CDMA devices which reduces greatly the market available as GSM is a more accepted standard worldwide. Also to develop BREW application Qualcomm requires the developer to acquire a license, a development kit, and to certify the application as TRUE BREW before releasing it to the market [8]. All these requirements can cost can mount up to $7000.00 making it an unattractive alternative.

Windows Mobile was also consider as an alternative for developing the application but the market for these devices is limited mainly to smart phones. While the flexibility granted by Windows Mobile was great for development the customer base for our product would have been reduced greatly if this platform was decided to be used.

J2ME is a Sun Microsystems developed technology, which allows developer to use JAVA as a development tool for Mobile devices. It uses a reduced version of the java virtual machine embedded into the device to run java programs called midlets in wireless phones. A great part of the phone industry worldwide has adopted this technology and a great amount of devices support it. Development tools for J2ME are freely available like Eclipse, MicroEmulator, and NetBeans. Given all the benefits of this technology the wireless application will be developed under it.

J2ME still presents some issues in the real world. Most of the cell phone carries in the US limit the capabilities of the phones restraining the virtual machines from certain parts of the cell phones. For our design we need access to the Bluetooth module, internal memory, and SMS messaging system. This can later be a concern for the product deployment in the market.

3.7. Communication Protocol

In order to establish a reliable communication between the cell phone application and the micro-controller application a communication protocol was defined. This protocol would run on top of the Bluetooth layer which is totally transparent to the protocol. A peer to peer model was chose in which both end points, the cell phone and the micro-controller, are seen as equal in terms of communication responsibilities and both are known as peers.

The applications needed to transmit simple commands along with some data representing the heart parameters being monitored. These parameters included Vent Rate in beats per minutes (BMP), PR Interval in milliseconds (ms), QRS duration in milliseconds (ms), QT duration in milliseconds (ms), QTc which is a corrected QT parameter, and the amplitude of the heart's T wave. All these parameters where sent using their ASCII representation with their respective command. Please refer to the Communication Protocol Appendix for syntax definition and a list of the commands.

3.7.1. Protocol Disadvantages

In order to send these commands the microcontroller relies in a function that sends the data through an UART port which is connected to the Bluetooth module. The application implementing the protocol has no knowledge if the Bluetooth connection is working or not, it assumes that every message is sent and arriving to the other peer. Another issue that was observed was that due to interference, the drop of the connection, garbage stored in the buffers, or some unwanted data known as echoes sent from the Bluetooth module might interrupt commands send from one of the peers.

3.7.2. Error Detection

To compensate some of these disadvantages the protocol provides the peers with a command for peer feedback. In case that a peer detects some anomaly in the communication like commands that make no sense, unanswered requests, or unfinished commands the peer can send a command to test the communication. The application sends this command to a peer with a string of ASCII characters and expects to receive that exact same string.

For code implementation, protocol syntax, and command definitions please refer to the Communication Appendix (APPENDIX A: Communication Protocol).

3.8. Cell Phone Device

3.8.1. Graphical Interface

The user interface was designed as a button driven GUI where all the features of the program can accessed through them.

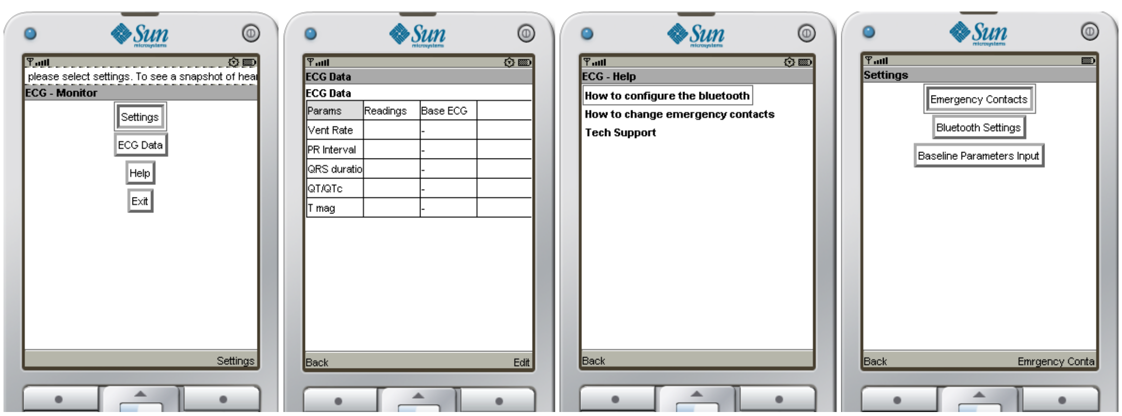


Figure 4– (from left to right) a) Main menu b) ECG Data Table c) Help Menu d) Settings Menu

In Figure 4 the various interfaces the user has access to are shown. In the main menu the user access to the ECG Data, Settings and Help. The ECG Data displays a table with relevant parameters of the patient's heart. In a same manner the in the help menu the user can access documentation with relevant information of the ECG Monitor system.

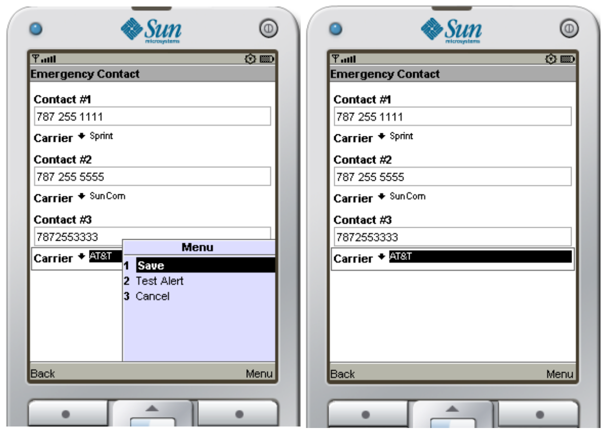


Figure 5 Emergency Contact Menu

In the settings menu the user can setup the emergency contacts and the Bluetooth settings. In we can see the menu where the user can setup up to three emergency contacts. The required data are the cell phone number and the carrier, after entering the data the user can proceed to save it, test the emergency contacts which will send test alert to all the contacts or cancel discarding any data entered.

3.8.2. Connecting to Bluetooth via Cell Phone

First of all, let's see a little overview of how the Bluetooth connection will be. The Bluetooth connection follows a client/server architecture (reference: http://en.wikipedia.org/wiki/Client-Server). The server is the device that becomes discoverable and offer services. On the other hand, the client is the device that connects to the server and asks to use a service offered by that server. In the diagram the server is offering a service named Send Data and the client is connected to the Server. The communication protocol used by them in the diagram is the RFCOMM protocol, a protocol for streaming data.

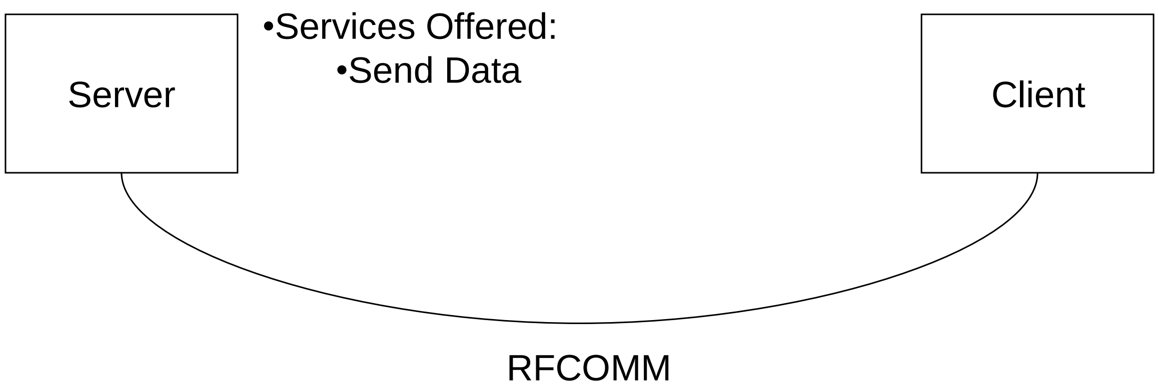


Figure 6 – Bluetooth communication model

In the case of the Wireless ECG the situation is similar. There are 2 main components: the Cell Phone and the Bluetooth module. The Cell Phone will be the client that will discover the server, in this case the Bluetooth module. When the connection is established the server will send the data to the client who will be waiting or the client will be able to request the information to the server.

At first we developed a cell phone application named BluetoothMidlet (MIDlet is the name of the applications for the cell phone) that tested the connecting capabilities and device and service searching capabilities of the Bluetooth integrated in the cell phone. This application provided full implementation of the Java methods for searching and connecting. Later the functionality of this application was integrated with the application that managed the GUIs, thus creating the main application for the cell phone.

When accessing the Bluetooth Settings menu from the settings the application will start a device inquiry and will present to the user a screen containing a list of the devices found so that the user will select the one he/she is interested to connect to After the user selects a device to connect the application will start a service search on the selected device and will present the user a screen containing a list of the services found on that device so that the user will select the service she/he is interested to connect to. Then, the application will show the user information about the connecting process.

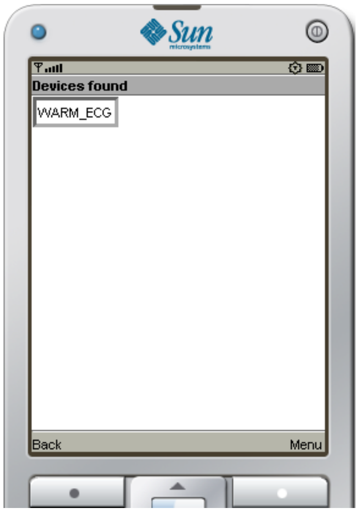


Figure 7 List of devices found. In this case there is only one.

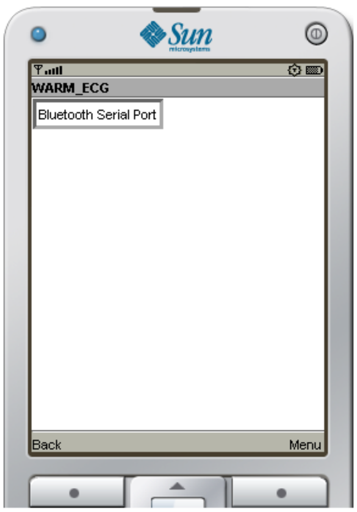


Figure 8 List of services found in a particular device. In this case there is only one.

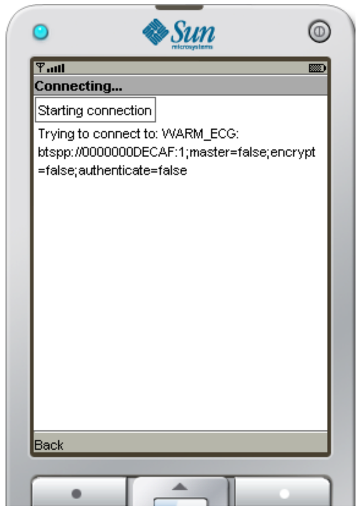


Figure 9 Information about the connecting process. It shows the friendly name and Bluetooth address of the device.

3.8.3. Emergency Alert & Settings Data

In case of emergency the system is configured to send and emergency alert in the form of a SMS text message that is sent through the internet. The messages sent to the emergency contacts in case of alert and in case of test alert are different. The exact content of the real emergency alert hasn't been yet decided due to the sensitive aspect of it.

In order to save the settings and pertinent data needed by the software the RMS database provided by J2ME was used. The first approach was to use xml files but these files only allowed to read data and not to write data to the xml file. Settings that are going to be saved in this database are SMS configuration for each carrier, emergency contacts number and carrier, the Bluetooth settings, and the baseline parameters boundaries that the doctor set in the application.

4. Methods and Approach to the Solution

This section presents the approach to the solution of WARM team on the developed of this project. Firs we present the different tasks and the final distribution among the team. Then the methodologies of where the system and part were tested. Finally a comparison of the proposed schedule and the final is made.

4.1. Activities and Distribution

To achieve the completion of the project WARM team is subdivided in two sub-groups the software and hardware group. The software group was composed of Alexis Ortiz and Wilfredo Bermudez. The hardware team was composed of Rafael Vega y Melvin Rivera. Table 1 summarizes the task distribution.

WARM ECG monitor system	**Resources**
Patient device	
Analog circuit	
Buy parts	Rafael Vega, Melvin Rivera
Assemble circuit	Rafael Vega, Melvin Rivera
Test circuit	Rafael Vega
Microcontroller	
Hardware	
Buy microcontroller and development kit	Rafael Vega
Interconnect with analog circuit	Melvin Rivera
Interconnect with Bluetooth	Melvin Rivera
Software	
Receive data from analog circuit and save it	Melvin Rivera, Wilfredo Bermudez
Bluetooth module initialization	Melvin Rivera
Baseline subroutine	Melvin Rivera, Alexis Ortiz, Wilfredo Bermúdez
Analyze received data	Melvin Rivera, Alexis Ortiz, Wilfredo Bermúdez
Send data to Bluetooth module	Wilfredo Bermudez, Melvin Rivera
Bluetooth module	
Buy Bluetooth module	Melvin Rivera
Create a PCB for the module	Melvin Rivera, Rafael Vega
Interconnect with TTl cable	Melvin Rivera
Test module	Melvin Rivera
Cell phone	
Software	
Phone programming environment	
Drivers installation	Wilfredo Bermudez
Phone interfacing with computer and test	Alexis Ortiz, Wilfredo Bermudez
Interfacing with Bluetooth	Alexis Ortiz, Wilfredo Bermudez
GUIs design	Wilfredo Bermudez
Communication protocols	Alexis Ortiz, Wilfredo Bermudez
Data request from microcontroller	Alexis Ortiz, Wilfredo Bermudez
Integrate System	Melvin Rivera, Alexis Ortiz, Wilfredo Bermúdez, Rafael Vega

Progress report and oral presentation II	Rafael Vega, Melvin Rivera, Alexis Ortiz, Wilfredo Bermúdez
Final report	Rafael Vega, Melvin Rivera, Wilfredo Bermúdez, Alexis Ortiz
Final presentation	Melvin Rivera, Wilfredo Bermudez
User manual	Rafael Vega, Alexis Ortiz

Table 1 Task distribution

4.2. Testing and Validation

The test and validation of the WARM ECG Monitoring System was made in a modular manner. All parts were tested as they were been developed. After testing each part the project was assembled and validated as a whole. While we were testing the system, we observed that it only worked for certain types of ECG signals. For that reason there are some design considerations on the future work part of this document.

4.3. Changes on the Schedule

There were different activities that need to be changed on the schedule. On this section we present the changes made and an explanation. You can also have a graphical representation on the Gant Chart Appendix.

- ✓ Analog circuit activities - the main change on schedule was as consequence of the analog circuit. This task was originally estimate for a duration of 13 days and it end with a duration of 45 days. This was due to the presence of noise on the signal. Because of this complication Rafael Vega was moved for other activities and we give more time to him to test and make the circuit.

- ✓ Bluetooth activities – the Bluetooth activities also suffer changes. In this case the reason was that the first Bluetooth module that we bought was lost on the post delivery process. For that reason we have to buy another one and the tasks related with the module was moved. The module was supposed to be in our hands for September 12 and we receive it on October 7. When we receive it there were a new task that have to be made, make a PCB in order to connect it with the microprocessor.

- ✓ Microcontroller activities

- Due to the backlog on the analog circuit, and Bluetooth module the activities of the microprocessor were also moved. In the case of the analog circuit, when we see that this task take too much time, we do the ADC connection testing with a sinusoidal signal at input. With this entire backlog all microprocessor task was moved and we end working weekends to mitigate the impact of those.

✓ Cell phone

- The only activity that suffers changes on the cell phone was the communication protocol implementation.

4.4. Contingency Method

To mitigate the multiple backlog and problems we make different changes on schedule. On the first part of the project Melvin Rivera was reassigned to help with the analog circuit while the Bluetooth module arrived. He was also assigned to work with the microcontroller in the initialization, data sampling, and UART communication process. Alexis Ortiz was assigned to deal with the Bluetooth in the cell phone. Due to the several delays extra hours were assigned to the team, and start worked on the weekends.

5. Market Overview

The following section describes other products and compares it with our product.

5.1. Potential Customers

Our potential customer base includes patients with the following risk factors:

✓ Diagnosed with or suffered from:

- Cardiovascular Disease
- Congenital Heart Defects

- Stroke
- Diabetes
- High Blood Pressure
- High Cholesterol Level
- Have a family history of heart disease
- Over the age of 65
- Overweight or obese
- Increased use of tobacco or alcohol
- Physical inactivity
- Had a heart transplant/pacemaker
- Waist size
- Triglycerides

5.2. Current Competition: Advantages and Disadvantages

Other existing products come from the LifeSync® Corporation and Alive Technologies [4]. LifeSync® has developed a wireless ECG transceiver which uses a special 12 lead electrode that is also produced by them. The patient transceiver uses Bluetooth to wirelessly transmit the data to a monitor transceiver which connects using wires to a standard hospital ECG machine. This helps with continuous monitoring of the patient, and makes it easier for nurses to set up a patient's ECG. However, since it still requires the monitor transceiver to be connected to a standard ECG machine, its main function is to replace the bedside ECG machine. It is not as portable as our unit, and the electrodes may make it uncomfortable to wear.

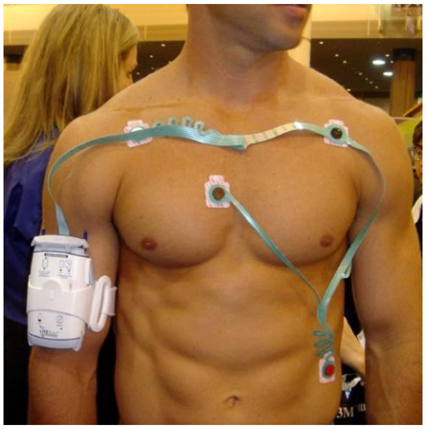


Figure 10 LifeSync Wireless ECG

The Alive Technologies Alive Heart Monitor is quite an advanced device that monitors the user's ECG and transmits it to the user's phone via Bluetooth [5]. The device connects with two electrodes to a patient's body. The main market of this product appears to be athletes since a software package called Mobile Sports Monitoring is also available for sale. With this software, the device can monitor the athlete's ECG, heart rate, speed, altitude, and tracks the person's location. However, the company is based in Australia and is still looking for distributers. Therefore, this product may not be easily available for purchase yet.

6. Results and Discussion

The WARM Wireless ECG design was made, but there are constrain that we left to a future work due to the time limitation. First the algorithm must be adaptable. It means that it will have the ability to adapt to the patient to satisfy all patient needs. On that moment it only works for waves that have elevate Q peaks compare with the P and T peak. The main problem on the implementation was do that the system runs on real time, that's because we need faster microprocessor that can execute all the analysis at the same time of the sample rate. That means that the process of analyze the wave have to be less than 694ms. Another constrain is that the system need a circuit that maintain the wave on a baseline. That is because when the patient moves the wave change it baseline and the amplitude parameters. Power consumption of the analog circuit is out of the scope of this project.

6.1. Social Impact

The social impact of the WARM ECG Monitor system cover from a better possible diagnosis, due to the patient lives his daily life and the constant monitoring. Also the diagnosed could be improved because the patient wasn't perturbed psychologically by the hospital environment, until saving economic resources by the reduction in hospitalization costs. The patient's family also suffer the benefit, that because with the constant monitor of their familiar, they could have more serenity. In case than an emergency occur the device can alert, and the patient could receive more rapidly the attention, incrementing their chances of survive.

6.2. Environmental Impact

This assessment was carried out by identifying the project's activities or actions that may cause environmental impacts and the environmental components and elements of each medium that may suffer such impacts. This project is categorized as low environmental impact since all electronic parts meets the requirements Restriction of Hazardous Substances (RoHS) directive. RoHS was originated in the European Union and restricts the use of specific hazardous materials found in electrical and electronic products. With this directive[2] the following banned substances are not present on these components:

- ✓ Cadmium
- ✓ Lead
- ✓ Mercury
- ✓ Hexavalent chromium
- ✓ PBB (Polybrominated Bi-Phenyl)
- ✓ PBDE (Polybrominated Diphenyl Ether)

6.2.1. Hardware Components Environmental Regulations

[2] http://europa.eu.int/eur-lex/pri/en/oj/dat/2006/l_037/l_03720060213en00190023.pdf

This section describes the hardware environmental regulation that the components meet. In table 2 environmental regulation and components are summarized.

Component	**Model/Info**	**Environmental Regulation**
Microcontroller	MSP430F149	Green (RoHS & no Sb/Br)
Bluetooth	Bluegiga wt12	RoHS
Instrumentation Amplifier	Analog devices *AD624ADZ*	RoHS
Differential Amplifier	National Semiconductor LM741	RoHS

Table 2. Hardware Components Environmental Regulations

Environmental Component	**Grade**
Physical	1
Social	2
Cultural	2
Economical	1
Public Health	2

Table 3: Environmental Susceptibility

Table 3 shows the level of impact the implementation of our device has on each aspect.

6.3. Legal Aspect

WARM Wireless ECG Monitoring System is a health care device. For that reason there are several legal aspects that are being taken in consideration. The specification of the device must be clear that this device have to be used with medical consultant. Also this device not pretends to substitute the regular ECG machines on delicate patients.

Another consideration is that the device not do a diagnostic, it only monitor that your parameter not violate the limits established by your doctor. In that context the patient is responsible of use properly the device and follows the instructions.

7. Final Cost Analysis

This section describes the total expenses of the project. It is analyzed and compared with the budget on the proposal document. This section is subdivided in the actual cost and a comparison of actual vs. budget.

7.1. Actual Cost

This section describes the cost of the entire project. It is divided in human resources, hardware parts cost and software components cost. The combined of this subdivision cost is what is called total project cost.

7.1.1. Human Resources Cost

Human resources are divided in fourth categories: Software Engineer, Hardware Engineer, Project Manager and the consultants. The payment per hour of these categories is 19.75, 19.17, 37.09 and 100 dollars respectively. The table 4 summarized the total cost of human resources. The total cost of human resources resulted in $38076.53.

Employees	Position	Dollar/hour	Worked Hours	Cost($)
Alexis Ortiz	Software Engineer	19.75	324	6399.00
Wilfredo Bermudez	Software Engineer	19.75	324	6399.00
Rafael Vega	Hardware Engineer	19.17	324	6211.08
Melvin Rivera	Hardware Engineer & Project Manager	37.09	324	12017.16
Sub-total				31026.24
Unemployment Insurance (1.40%)				434.36
Disability (1.1%)				341.29
S.S 5.3%				1644.39
Healthcare (7.4%)				2295.94
401K/403b (4.3%)				1334.13
Consultant		100	10	1000
Total				38076.35

Table 4 .Summarized the total human resources cost.

7.1.2. Hardware Component Cost

Hardware components are divided in analog circuit components (with a cost of 65.43), Bluetooth module (cost of 59.81) and equipment cost that include cell phone, electrodes and msp430F149 flash emulator kit (with a cost of 358.08). The total hardware component cost resulted in $483.32. The next subsections describe that in more details.

7.1.2.1. Analog Circuit

Table 5 describes the total cost of the analog circuit.

Analog circuit			
Part	Qty.	Unit Price ($)	Total Price ($)
AD624	2	25.02	50.04
PS 2506	2	.99	1.98
LM 741	1	1.39	1.39
R 3.5k	1	0.05	0.05
R 150	1	0.05	0.05
R 10k	1	0.05	0.05
R 4M	1	0.05	0.05
C o.1u	1	0.15	0.15
C 1u	1	0.16	0.16

Adjusted R(200k)	1	1.48	1.48
9 volts battery	4	2.50	10.00
Total			65.43

Table 5 Analog Circuit Costs

7.1.2.2. Equipment Cost

Table 6 summarizes the equipment cost.

Equipment	Cost ($)
Flash Emulator	138.24
Microprocessor	8.84
Cell phone	196
Electrodes (30)	15.00
Bluetooth module	59.81
Total	417.89

Table 6 Equipment

7.1.2.3. Software Components Cost

The fallowing programs were used on the development process:

- ✓ **Hyper terminal** – Uses to program the Bluetooth module iWrap firmware.
- ✓ **Code Composer** - Integrated Development Environment used for the development of the firmware in the MSP430 microcontroller.
- ✓ **Net beans 6**.1 IDE - Integrated Development Environment used for develop the cell phone application.

- ✓ BitPim - program that allows you to view and manipulate data on many CDMA phones from LG, Samsung, Sanyo and other manufacturers. Used to download the application to the cell phone.

Program	Price	Quantity	Cost
Hyper terminal	$0.00	1	$0.00
Code composer	$0.00	1	$0.00
Net beans	$0.00	1	$0.00
BitPin	$0.00	1	$0.00

Table 7 Software Cost

From Table 7 displays that there were no software cost due to the use of freeware software and trial versions of the software.

7.1.3. Total Cost

The project total cost was 38559.7 and with an overhead of 150 the total project cost result in 96399.20 dollars. Table 8summarized the total project cost.

Total cost	
analog circuit	$65.43
equipments and parts	$417.89
Employers	$38,076.35
Subtotal	$38559.7
overhead 150%	$96399.20

Table 8Total Project Cost

7.2. Actual Cost vs. Budget

In this section a comparison between total and estimated cost is made part by part.

7.2.1. Human Resources

Estimated budget for human resources was $40,551.71 and the final cost is $38076.35 with a difference of $2475.36 below the estimated. The difference between the estimated and actual cost is due to the following: the total hours worked was 324 that were 44 more than estimate (estimate was 280). Although that difference increase the total employee salary cost, the total consultant hours decreased by 70 (used 10 estimated 80), that give a total of 7000 dollars less than estimates. For that reason the final cost is 2475.36 less than estimated.

7.2.2. Analog Circuit

The estimated total cost of analog circuit was $17.14 and ended cost 65.43. The main changes were on the AD624. The estimated cost was $6.16 but it ended-up being $25.02 per circuit, it requires an additional one so the final cost ended being $50.04 with a difference of $43.88 above the estimated one. The PS2506 suffered changes with an estimation of $3.50 and a real cost of $1.98 for two units which is $1.52 below the estimated cost. Batteries were estimated to have a cost of $4.10, their real cost was $10.00.

7.2.3. Equipment and Parts

The total estimated cost for equipment was 437.81 and the final cost was $417.89 with a difference of $19.92 lower than the estimated. The cost for the electrodes was estimated in $45s for a box of 50 units. For reasons of availability we bought a box of 30 units at a price of $15.00, 30 dollar less than estimate. The Bluetooth module was estimated to cost $49.73 and the final cost was $59.81 with a difference of $10.08 above of the estimated cost.

7.2.4. Budget vs. Cost Summary

The total estimated budget for project was $102517 and final project cost was $96399.5. The difference is $6117.25 below the estimated. The table 9 summarized the budget vs. the cost.

Budget		Expense	
Category	**Cost**	**Category**	**Cost**
Total Employment Cost	$40,551.71	*Total Employment Cost*	38076.35
Total Hardware Cost	$454.95	*Total Hardware Cost*	$483.41
Project Cost	$41006.66	*Project Cost*	$38559.8
150% Overhead	$61510	*150% Overhead*	$57,839.6
Total Project Cost	***$102517***	***Total Project Cost***	***$96399.4***

Table 9 Budget vs. Expenses

8. Conclusions

Our team was able to develop a system that detects and analyzes an ECG signal, is able to communicate wirelessly with a cell phone application and sends messages to the emergency contacts. Our modular approach allowed us to successfully produce each of the systems parts. Software and hardware integration was also successful. However, the system still exhibits some constraints that we would like to address before developing a marketable design.

For the WARM Wireless ECG Monitor to be marketable as a portable medical device, we have to address the issues of power consumption and the ECG detection algorithm. The complexity of the algorithm in our prototype only provides detection for certain types of ECG signals. It must be modified so it can analyze all possible cases.

We developed a prototype at an expense of $483.41. Further hardware modifications are not expected to inflate the cost too much. This means that the development cost of a production version should be much lower. The size of the components is very discrete, guaranteeing that the WARM Wireless ECG Monitor will be small enough to be wearable. All the components are RoHS compliant, minimizing the environmental impact that the disposal of the WARM Wireless ECG Monitor might have.

Since WARM Wireless ECG monitor will allow a patient to monitor his heart without being constrained to a hospital or home, it will have a positive social impact. Also, the system is designed only to be operated under the supervision of a doctor and does not pretend to diagnose any heart disease. Even if used as intended, it could have a moderate legal impact if usage is not carefully supervised.

9. Future Work

After working with the project and dealing with road blockers during the implementation phase of the project we realized that some details of our design could be rethought for optimizing the performance of our project. These details cover almost every area of the project: the analog circuit, the Bluetooth module, the microcontroller and the microcontroller code for processing the ECG data.

9.1. Analog Circuit

The analog circuit used in our design was not optimized for use in a portable device. As part of the final product, we need a circuit with low power consumption to maximize battery life. The Circuit in Figure 11 is an alternative that uses low power CMOS amplifiers. It works with a single supply voltage that can range from 2.7 V to 5.5 V. Also, it can be assembled in minimal space, since the amplifiers it requires are available in MSOP and SOIC micro packages.

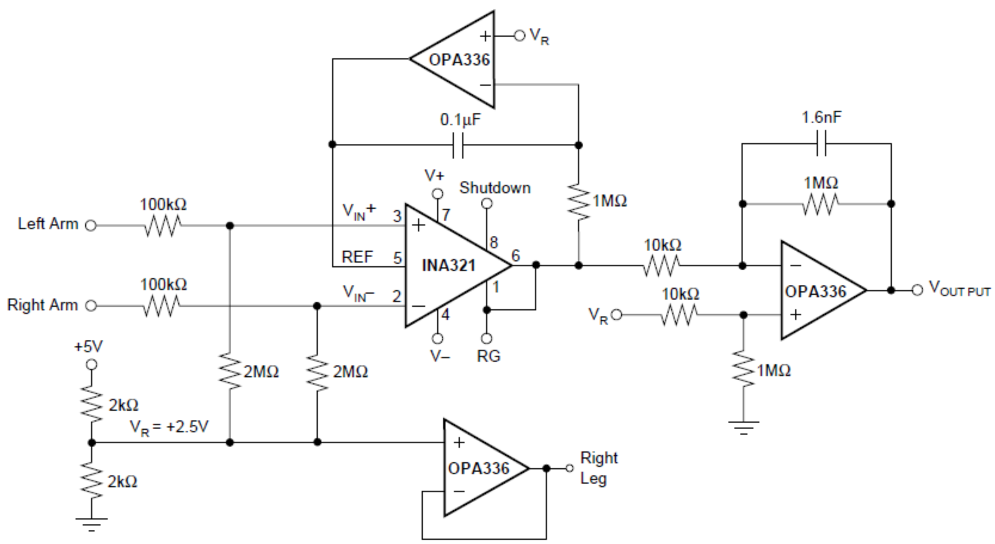


Figure 11 Analog Circuit Schematic

9.2. Bluetooth Module

We could also optimize the power consumption of the device using a class 1 Bluetooth module. Class 1 Bluetooth modules have less power consumption than class 2 Bluetooth devices (as the one we are currently using). Although the range of class 1 Bluetooth modules is smaller than class 2 Bluetooth modules this will not affect us since the device and the cell phone should be always near.

9.3. Microcontroller

We were evaluating the possibility of using a Digital Signal Processor (DSP) because of its capabilities with the signal and pipelines that might aid us in our work, but we discarded that option because the DSPs consume too much power for this type of project. The MSP430F247 is a counterpart microcontroller to the MSP430F149. It has a faster processor running at 16MHz, twice as much memory, and consumes less power. The added memory could allow having the extra software needed to constantly adapt the peak detection parameters.

9.4. Algorithm

Software performance is a must for such applications, optimizing the code could also improve the time needed for processing the signal. Optimizing the signal filtering circuit would reduce the need of software filtering reducing processing time from the micro-processor. Implementing an auto-zero circuit at the output of the monitoring circuit might also reduce unwanted signal offsets which might drastically change the peak detection parameters. Also, since the ECG waveform of a every people is different (in many factors as size appearance and polarization of almost all five waves) and also the ECG waveform is dependent on emotions and state of mood, we need an algorithm that will work with any of these different waveforms and variations. A solution for this is the use of an adaptive algorithm. Adaptive algorithms changes in behavior based on the information it receives. Our approach will be letting the algorithm sample the ECG waveform so that it knows the primary characteristics of it and then adapting to each possible change. This will take the processing to a new level and the detection of cardiac anomaly would be more precise and accurate.

Bibliographic References

[1] Bluegiga, "wt12 Data Sheet", Version 2.5, November 16, 2007

[2] Bluegiga, "iWrap 2.2.0 User Guide", version 3.3, November 21, 2007.

[3] Texas Instrument, "msp430F149x1xx User's Guide, 2006.

[4] Texas Instrument, "msp430x13x, msp430x14x, msp230x14x1 Mixed Signal Microcontroller", July 2000- Revised June 2004.

[5] Texas Instrument, "MSP430 Universal Synchronous Asynchronous Receive/Transmit Communication Interface", Application Report, April 1999.

[6] Analog Circuits, " Precision Intrumentation Amplifier AD624" , 1999

[7] NEC, " Photocoupler PS2506-1,-2,-4,PS2506L-1,-2,-4" , 4th Edition, 1997

[8] SGS Thomson Microelectronics, " UA741", October 1997

[9] Dionex "Using Smoothing Algorithm to Reduce Baseline noise in chromatography

[10] NetBeans "Java ME and JavaFX Mobile Learning Trail", http://www.netbeans.org/kb/trails/mobility.html

[11] Wireless Java and J2ME Tutorials, http://www.palowireless.com/java/tutorials.asp

[12] Micro Emulator http://www.microemu.org/, edited on march 3, 2008

[13] Scientific American, "Home is Where ECG is", June 2000,

http://www.sciam.com/article.cfm?id=home-is-where-the-ecg-is&page=1

[14] Chia-Hung Chen, Shi-Gun Pan, Peter Kinget, "ECG Measurement System",

http://www.cisl.columbia.edu/kinget_group/student_projects/ECG%20Report/E6001%20ECG%20final%20report.htm#1

[15] Qusay H. Mahmoud, Part 1 and 2: The Java APIs for Bluetooth Wireless Technology,

http://developers.sun.com/mobility/midp/articles/bluetooth2/, April 2003

[16] Enrique Ortiz, Using the Java APIs for Bluetooth, Part 2 - Putting the Core APIs to Work,

http://developers.sun.com/mobility/apis/articles/bluetoothcore/, February 2005

[17] Sun Microsystems, User'sGuide SunJava TM WirelessToolkitforCLDCVersion2.5.1 digital book, May 2007

[18] java API, http://java.sun.com/javame/reference/apis

Appendixes

APPENDIX A: Communication Protocol

Syntax:

```
   Command ::= letter'<'data'>'
   Data ::= letter | number | '' | ','data
number ::= [+-]?['0'-'9']+
letter ::= ['A'-'Z''a'-'z']+
```

Protocol:

a<data>

The monitor sends this command to the cell phone notifying it that there is an alert event. The data should be composed by the ECG parameters separated by a comma in the following order: VentRate,PRInterval,QRSduration,QT/QTc, T ampl

e<>

Request from the cell phone to the monitor to send the latest set of ECG parameters data.

h<data>

This command can be sent by either device to check if the connection is still working. Upon arrival of the command the device should acknowledge it by sending an h<data> command where data is the same as the one received.

i<data>

The data parameters calculated by the monitor from the patient's heart need to be compared with a range predefined by a doctor. To set this range of acceptable values the cell phone needs to use the i and j commands. The cell phone sends the i<data> command to the monitor to set the initial range of the heart parameters. The data should be composed by the ECG parameters separated by a comma in the following order: VentRate,PRInterval,QRSduration,QT/QTc,T ampl

j<data>

The data parameters calculated by the monitor from the patient's heart need to be compared with a range predefined by a doctor. To set this range of acceptable values the cell phone needs to use the i and j commands. The cell phone sends the j<data> command to the monitor to set the upper range of the heart parameters. The data should be composed by the ECG parameters separated by a comma in the following order: VentRate,PRInterval,QRSduration,QT/QTc,T ampl

p<data>

The monitor sends this command to the cell phone as a respond to an e<> command. The data should be composed by the ECG parameters separated by a comma in the following order: VentRate,PRInterval,QRSduration,QT/QTc,T ampl

z<>

This command is sent as a respond by either of the devices in case of error. For instance if a malformed command is received and not understood a z<> command can be sent.

Code:

<------------ MICRO -------------->

```
void commandParse(char* c){

     int i = 0;
     //while(c[i] != '>'){

       switch (c[i]){
       case 'e':
         if( c[++i] == '<' ){
           /*
           get ecg data and respond with it to the cellphone
           */
           send_to_UART("p<");
           parameter_Append();
           return;
         }

         else{
           send_to_UART("z<>\r"); //command not found
           return;
         }
         break;

       case 'h': //acknowledge back
         if( c[++i] == '<' ){
           UART0_putchar('h');
           UART0_putchar('<');

           i++;
           while( c[i] != '>' ){
             UART0_putchar(c[i]);
             i++;
           }
           UART0_putchar('>');
           UART0_putchar('\r');
           //i--;
           return;
         }
```

```
      else{
        send_to_UART("z<>\r"); //command not found
        return;
      }
      break;

    case 'i':
         if( c[++i] == '<' ){
                // save the lower end of the parameters
                saveBaseParameters( c, 0);
                return;
         }

         else{
                send_to_UART("z<>\r"); //command not found
                return;
         }
         break;

    case 'j':
         if( c[++i] == '<' ){
                // save the upper end of the parameters
                saveBaseParameters( c, 1);
                return;
         }

         else{
                send_to_UART("z<>\r"); //command not found
                return;
         }
         break;

    case 'z':
      if( c[++i] == '<' ){
        /*
        In case of error the command should be resent
        */
        return;
      }

      else{
        send_to_UART("z<>\r"); //command not found
        return;
      }
      break;

    default:
      send_to_UART("z<>\r"); //this command is useless in monitor
      break;

    }
  //  i++;
  //}
}
```

<------------ Cell Phone -------------->

```
void commandParse(String s){
   int i = s.indexOf('\r');//looks for character \r since all strings from UART end with this character
   i++;
   while(s.charAt(i) != '>'){

      switch (s.charAt(i)){
         case 'a':   //Searches for an alert command: a<>
            i++;
            if( s.charAt(i) == '<' ){
              receivedData = parseParameters(s.substring(i + 1)); //parses the parameters
               if(receivedData != null)
               {
                  emergency();  //send an emergency alert to the user and the emergency

                                      // contacts
               }
               else
               {
                  write(ERROR_MESSAGE);
               }
            }

            else{
              write(ERROR_MESSAGE);
            }
            i += (s.substring(i + 1)).length() - 1;
         break;

         case 'p':   //Searches for a data sent command p<>
            i++;
            if( s.charAt(i) == '<' ){
              receivedData = parseParameters(s.substring(i + 1)); //parses the parameters
               if(receivedData != null)
               {

                  //this block of text synchronizes the receiving thread with the main thread
                  synchronized(this)
                  {
                     tries = 0;
                     waitingData = false;
                  }
               }
               else
               {
                  write(ERROR_MESSAGE);
```

```
        }
    }

    else{
        write(ERROR_MESSAGE);
    }
    i += (s.substring(i + 1)).length() - 1;
break;

case 'h': //Searches for a aknowledge back command h<>
    i++;
    if( s.charAt(i) == '<' ){

        String p = s.substring(i + 1, s.length() - 1);
        if(waitingData)
        {
            if(p.equals(messageSent.substring(i + 1, s.length() - 1)))
            {
                alert("Connection Verified: Connected", 4000, AlertType.CONFIRMATION, null,

                          (Form) parent.getDisplay().getCurrent());

                //this block of text synchronizes the receiving thread with the main thread
                synchronized(this)
                {
                    tries = 0;
                    waitingData = false;
                }
            }
            else
            {
                write(ERROR_MESSAGE);
            }
        }
        else
        {

            write("h<" + p + ">");
        }

    }

    else{
      write(ERROR_MESSAGE);
    }
    i += (s.substring(i + 1)).length() - 1;
    break;



case 'z':   //Searches for an error command z<>
    i++;
    if( s.charAt(i) == '<' ){
```

```
                /*
                 In case of error the command should be resent
                */

                if(waitingData)
                {
                    if(tries < NUM_OF_TRIES)
                    {
                        write(messageSent);
                        tries++;
                    }
                    else
                    {
                        tries = 0;
                        alert("Oops! It seems there has been a problem.\n Try again.", 4000,

                                  AlertType.ERROR, null, (Form) parent.getDisplay().getCurrent());

                    }
                }
                else
                {

                    write(ERROR_MESSAGE);
                }
            }

            else{
              write(ERROR_MESSAGE);
            }
            break;

        default:
            i += (s.substring(i)).length() - 2;
          write(ERROR_MESSAGE);
        break;

      }
      i++;
    }
}
```

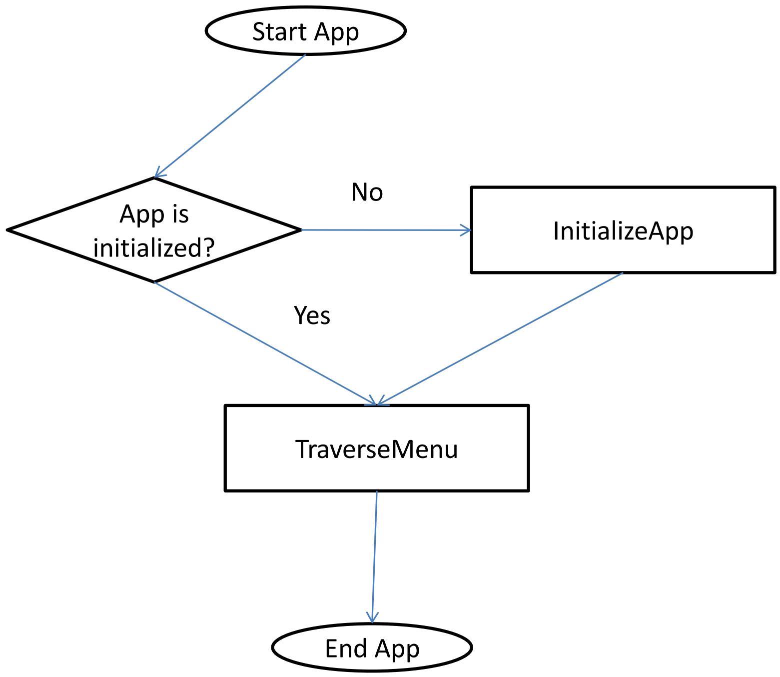
APPENDIX B: Peak Detection Code

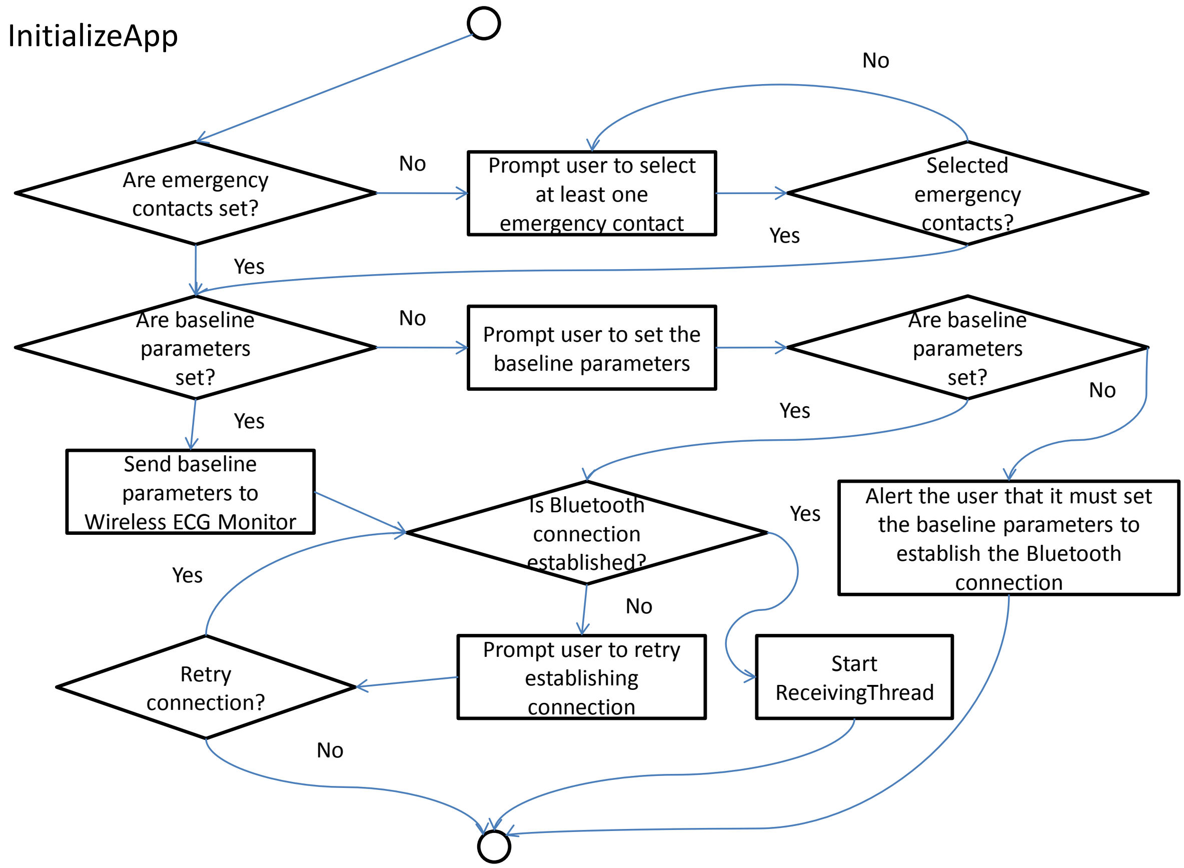
```
/*************************************************
* Method : higher_value                    *
* Return : Higher value of a dataset            *
* Param  : *data - array                  *
*        leftLimit - started point to analyze   *
*        rightLimit - end point to analyze     *
* Author: Misael Perez
* Modified by: Melvin Rivera, Alexis Ortiz, Wilfrdo Bermudez
*************************************************/
int higher_value( int *dataValue ,  int leftLimit, int rightLimit) {
 int counter = 0;
 int value = leftLimit;

 for( counter = leftLimit+1; counter < rightLimit; counter++ ){
  if( dataValue[value] < dataValue[counter] )
    value=counter;
 }
 return value;
}

/*************************************************
* Method : lower_value                    *
* Return : Lower value position of a dataset      *
* Param  : *data - array                  *
*        leftLimit - started point to analyze   *
*        rightLimit - end point to analyze     *

* Author: Misael Perez                    *
* Modified by: Melvin Rivera, Alexis Ortiz, Wilfrdo Bermudez
*************************************************/
int lower_value(  int *dataValue ,  int leftLimit, int rightLimit) {
 int counter = 0;

 for( counter = leftLimit+1; counter < rightLimit; counter++ ){
  if( dataValue[leftLimit] > dataValue[counter] )
    leftLimit=counter;
 }
 return leftLimit;
}

/*************************************************
* Method : right_bound_analysis             *
* Return : Position where a peak finish          *
* Param  : *data - array                  *
*        temp_future - last analysed point     *
```

```
* Author: Misael Perez                    *
* Modified by: Melvin Rivera, Alexis Ortiz, Wilfrdo Bermudez
*************************************************/
int right_bound_analysis(  int *dataValue,  int actual_point){

  while( dataValue[actual_point] > dataValue[ actual_point+6 ] ){
    actual_point = actual_point+1;
  }

  return lower_value( dataValue, actual_point, actual_point+6);
}

/*************************************************
* Method : left_bound_analysis               *
* Return : Position where a peak start          *
* Param  : *dataValue - array                *
*        position - first analysed point       *

* Author: Misael Perez
* Modified by: Melvin Rivera, Alexis Ortiz, Wilfrdo Bermudez
*************************************************/
int left_bound_analysis( int *dataValue,  int actual_point){
//  int actual_point = position;

  while( dataValue[ actual_point ] > dataValue[ actual_point-6 ] ){
    actual_point = actual_point-1;
  }

  return lower_value( dataValue, actual_point-6, actual_point);
}

/*************************************************
* Method : look2future                    *
* Return : void - Detect Peaks             *
* Param  : *data - data array              *
*        dataSize - amount of data         *

* Author: Misael Perez
*Modified by: Melvin Rivera, Alexis Ortiz, Wilfrdo Bermudez
*************************************************/
void look2future( int *dataValue ,int zero, int dataSize, int look2future, int recount, int* peakPosition,
int* peakLeftBound, int* peakRightBound) {
  int count=0;
  int actual_position = zero + look2future;
  if(actual_position >= DATA_SIZE2)
          return;
```

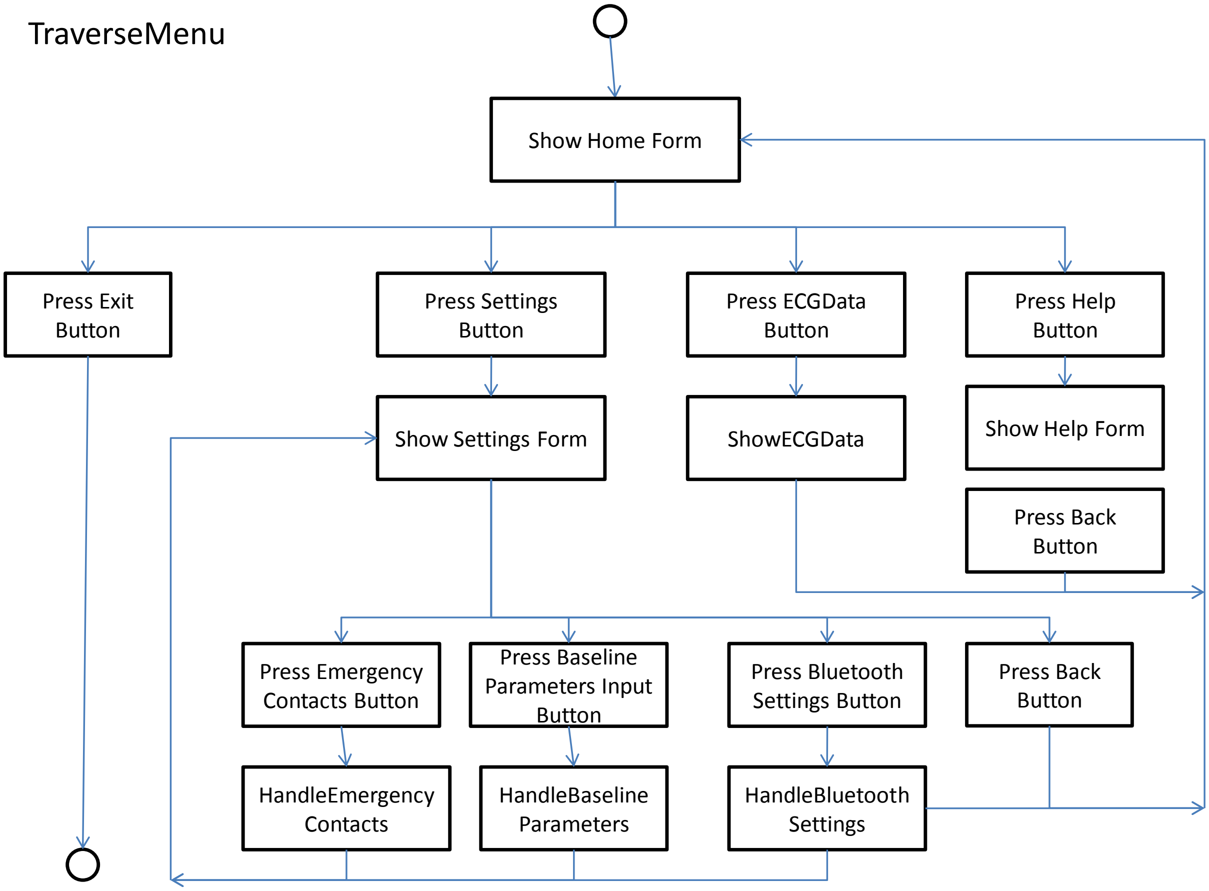
```
 do{
  if(delta_ave < abs( dataValue[ actual_position ] - dataValue[ actual_position-look2future ] )){
   //Tengo el pico en un intervalo
   if( dataValue[ actual_position-look2future] > dataValue[actual_position] ){


     peakPosition[count]=higher_value(dataValue, actual_position, actual_position+recount);
     peakLeftBound[count]=left_bound_analysis(dataValue, actual_position);
     peakRightBound[count]=right_bound_analysis(dataValue, actual_position);
     actual_position = peakRightBound[count]+look2future;
     return;
   }
   //Tengo que buscar el pico para ponerlo en un intervalo
   else{
    do{
     actual_position = actual_position + 1;
    }while(dataValue[actual_position] < dataValue[ actual_position + recount ] );

     peakPosition[count]=higher_value(dataValue, actual_position, actual_position+recount);
     peakLeftBound[count]=left_bound_analysis(dataValue, actual_position);
     peakRightBound[count]=right_bound_analysis(dataValue, actual_position);
     actual_position = peakRightBound[count]+look2future;
     return;
   }// end else
  }//end if

  else{
   actual_position = actual_position+look2future;
  }
 }while(actual_position < dataSize);

}
```

APPENDIX C: Cell Phone Application Flowcharts

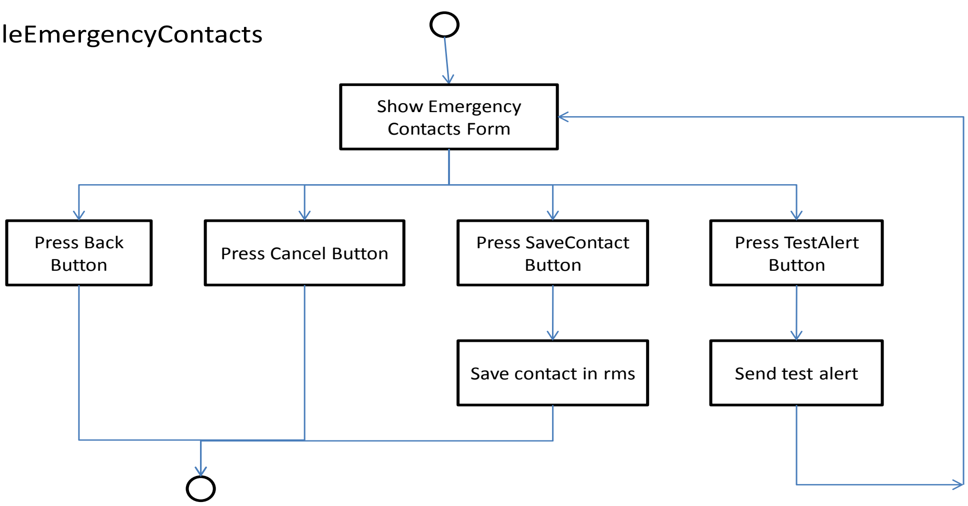



InitializeApp
Are emergency contacts set?
No
Prompt user to select at least one emergency contact
Selected emergency contacts?
No
Yes
Yes
Are baseline parameters set?
No
Prompt user to set the baseline parameters
Are baseline parameters set?
No
Yes
Yes
Send baseline parameters to Wireless ECG Monitor
Is Bluetooth connection established?
Yes
Alert the user that it must set the baseline parameters to establish the Bluetooth connection
Yes
No
Prompt user to retry establishing connection
Retry connection?
Start ReceivingThread
No

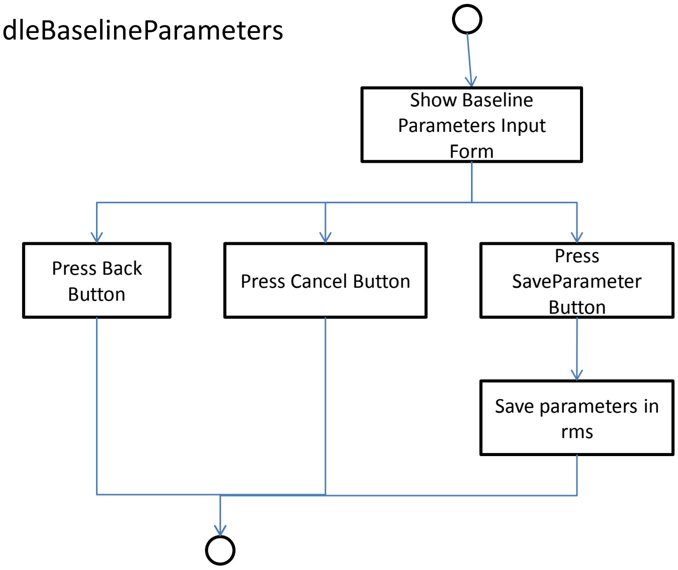
TraverseMenu



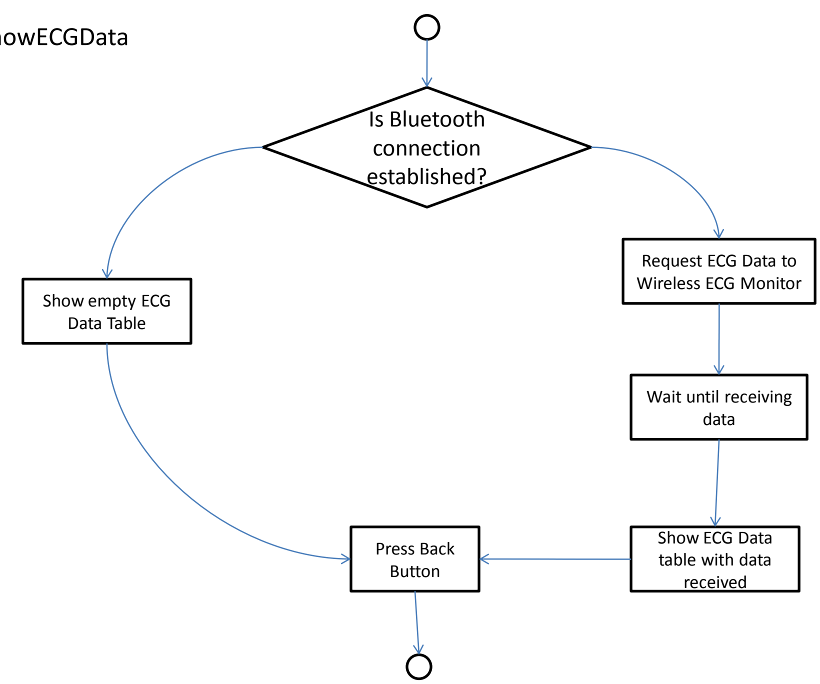
HandleEmergencyContacts

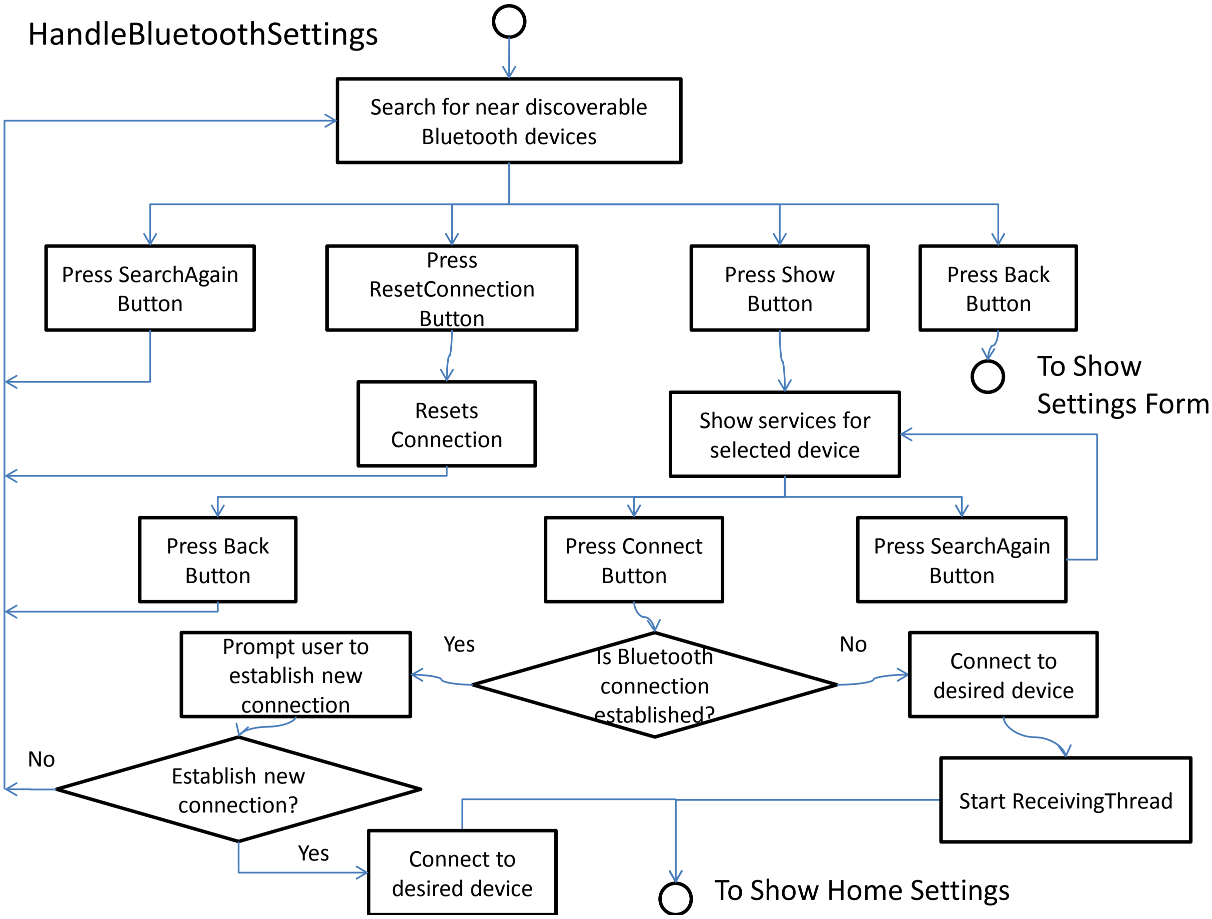


HandleBaselineParameters

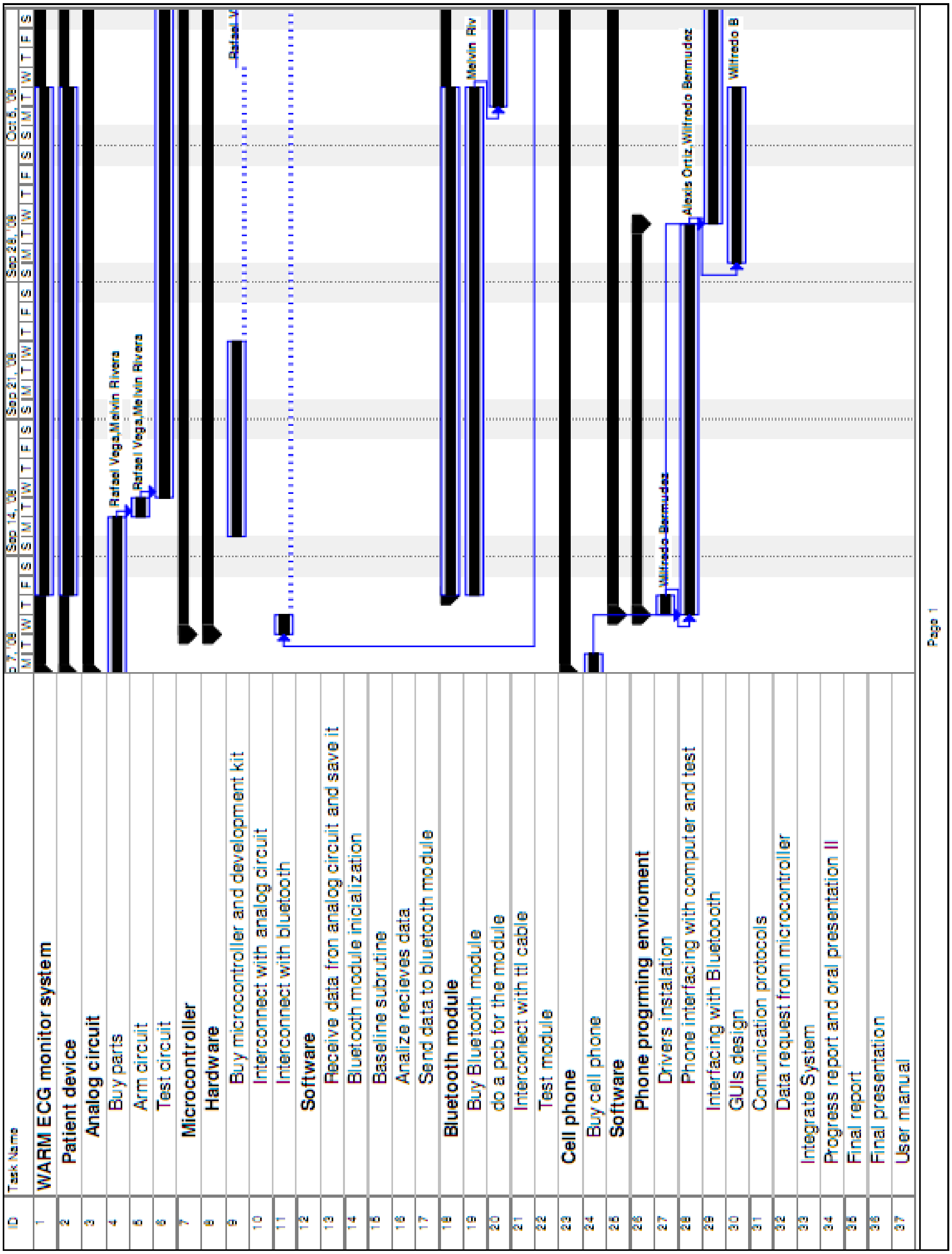


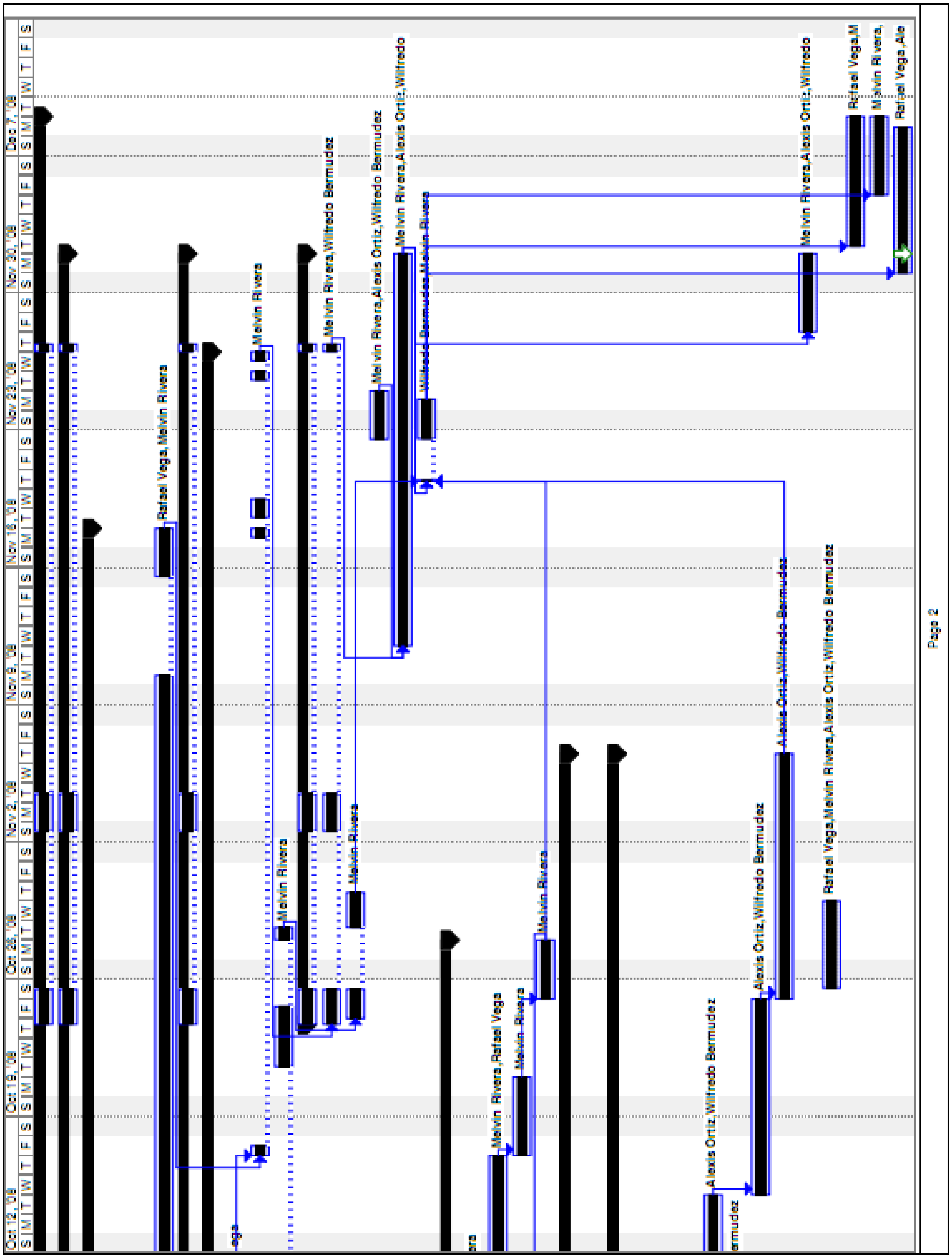
ShowECGData




HandleBluetoothSettings
Search for near discoverable Bluetooth devices
Press SearchAgain Button
Press ResetConnection Button
Press Show Button
Press Back Button
To Show Settings Form
Resets Connection
Show services for selected device
Press Back Button
Press Connect Button
Press SearchAgain Button
Prompt user to establish new connection
Yes
Is Bluetooth connection established?
No
Connect to desired device
No
Establish new connection?
Start ReceivingThread
Yes
Connect to desired device
To Show Home Settings

APPENDIX D: Gantt Chart


ID
Task Name
1 WARM ECG monitor system
2 Patient device
3 Analog circuit
4 Buy parts
5 Arm circuit
6 Test circuit
7 Microcontroller
8 Hardware
9 Buy microcontroller and development kit
10 Interconnect with analog circuit
11 Interconnect with bluetooth
12 Software
13 Receive data fron analog circuit and save it
14 Bluetooth module inicialization
15 Baseline subrutine
16 Analize recieves data
17 Send data to bluetooth module
18 Bluetooth module
19 Buy Bluetooth module
20 do a pcb for the module
21 Interconect with ttl cable
22 Test module
23 Cell phone
24 Buy cell phone
25 Software
26 Phone progrming enviroment
27 Drivers instalation
28 Phone interfacing with computer and test
29 Interfacing with Bluetooth
30 GUIs design
31 Comunication protocols
32 Data request from microcontroller
33 Integrate System
34 Progress report and oral presentation II
35 Final report
36 Final presentation
37 User manual
Sep 14, '08
Sep 21, '08
Sep 28, '08
Oct 5, '08
Rafael Vega,Melvin Rivera
Rafael Vega,Melvin Rivera
Wilfredo Bermudez
Alexis Ortiz,Wilfredo Bermudez
Wilfredo B
Melvin Riv
Page 1


Oct 12, '08
Oct 19, '08
Oct 26, '08
Nov 2, '08
Nov 9, '08
Nov 16, '08
Nov 23, '08
Nov 30, '08
Dec 7, '08
Rafael Vega,Melvin Rivera
Melvin Rivera
Melvin Rivera,Wilfredo Bermudez
Melvin Rivera,Alexis Ortiz,Wilfredo Bermudez
Melvin Rivera,Alexis Ortiz,Wilfredo
Melvin Rivera,Rafael Vega
Alexis Ortiz,Wilfredo Bermudez
Alexis Ortiz,Wilfredo Bermudez
Rafael Vega,Melvin Rivera,Alexis Ortiz,Wilfredo Bermudez
Page 2

APPENDIX E: Compatible Phone List

Phones available with Bluetooth technology and J2ME support:

For sake of simplicity only 20 models are shown by company, each carrier may have more the 20 wireless phones that meet this requirement. Only the mayor Puerto Rico carriers were taken in consideration [17]:

AT&T/Cingular

Audiovox SMT 5600 / SPV C500
HP iPAQ 510 Voice Messenger
HTC 2125 / 2100 (Faraday)
HTC 3125 / Smartflip / 8500 (Star Trek)
HTC 8125 / 8100 / MDA (USA)
K-JAM / P4300 (Wizard)
HTC Dash / S620 / S621 (Excalibur)
HTC Tilt 8925 / TyTN II (Kaiser)
HTC TyTN / 8525 / JasJam (Hermes)
LG CG300
LG CU400 / CU405
LG CU500
LG CU515
LG Shine CU720
LG Trax CU575
LG Vu / CU920 / CU915
Motorola KRZR K1
Motorola L2
Motorola L6
LG CU320

T-Mobile/Suncom

Motorola RAZR2 V8
Motorola RIZR Z3
Motorola ROKR E8
Motorola Sidekick Slide
Motorola V195 / V197
Motorola V360 / V361
Motorola V551 / V330 / V547 / V555 / V557
Motorola V600
Motorola W490
Nokia 2760
HTC 8125 / 8100 / MDA / K-JAM / P4300
HTC Dash / S620 / S621 (Excalibur)
HTC SDA (USA) / SP5m (Tornado)
HTC Shadow
HTC Wing / P4350 (Herald)
Motorola A630
Motorola KRZR K1
Motorola PEBL U6
Motorola RAZR V3
Motorola RAZR V3i / V3t / V3e / V3r

Sprint

HTC Touch (CDMA) / XV6900
LG Fusic / LX-550
LG LX-150
LG LX-160 / Flare
LG LX-350
LG LX-400
LG Muziq / AX-565 / UX-565
LG PM-325
LG Rumor / Scoop / UX-260
Motorola KRZR K1m
Motorola RAZR VE20
Motorola RAZR2 V9m
Motorola Renegade V950
Motorola SLVR L7c
Nokia 6165i
BlackBerry 7130e
BlackBerry 7250
BlackBerry 8703e
BlackBerry 8830
BlackBerry Curve 8330