

March 2020

Trauma Detection Personal Locator Beacon System

Sakshi Sharma

University of South Florida

Follow this and additional works at: <https://digitalcommons.usf.edu/etd>



Part of the [Electrical and Computer Engineering Commons](#)

Scholar Commons Citation

Sharma, Sakshi, "Trauma Detection Personal Locator Beacon System" (2020). *Graduate Theses and Dissertations*.

<https://digitalcommons.usf.edu/etd/8995>

This Thesis is brought to you for free and open access by the Graduate School at Digital Commons @ University of South Florida. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Digital Commons @ University of South Florida. For more information, please contact scholarcommons@usf.edu.

Trauma Detection Personal Locator Beacon System

by

Sakshi Sharma

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Electrical Engineering
Department of Electrical Engineering
College of Engineering
University of South Florida

Major Professor: Stephen E. Sadow, Ph.D.
Sylvia Thomas, Ph.D.
Alexandro Castellanos, Ph.D.

Date of Approval:
March 18, 2020

Keywords: Skin Conductance, Pulse Oximeter, Body Temperature, Freefall Detection &
Impact Detection, Microcontroller

Copyright © 2020, Sakshi Sharma

Dedication

This thesis is mainly dedicated to my brother Vaibhav Narula who guided me in the process of gaining admission to University of South Florida and sister Shweta Sharma who inspired me to enroll, stick into thesis track and was a great supporter during the journey of the thesis. Also, I would like to dedicate the thesis to my Major Professor Dr. Stephen Sadow, who apart from being a great guide was always patient and supportive throughout the journey and always showed his trust on me to accomplish challenges I did encounter during thesis. Finally, I would like to dedicate this thesis to my mother Pratibha Sharma who always motivated me to overcome challenges I encountered in the journey.

Acknowledgments

I would like to express my sincere gratitude to Dr. Stephen Sadow for showering his continuous support, patience, motivation and enthusiasm during my M.S. research study. His guidance helped me in resolving all kinds of problems I encountered at the time of research and writing of my thesis. I could not have imagined having a better guide and mentor for my M.S. thesis study.

Besides my advisor, I would like to acknowledge my committee members Dr. Alexandro Castellanos and Dr. Sylvia Thomas for being a part of my defense committee and also for showing their encouragement and insightful comments on my thesis work.

I would also like to sincerely thank Matt Hopper, Alex Otten and Mohammed Beygi for discussing the technicalities which I encountered during the research of my thesis.

I would like to acknowledge my sister, Shweta Sharma for motivating me to choose thesis option and my family Pratibha Sharma, Dinesh Kumar Sharma and Fido for being a continuous support during the process.

Table of Contents

List of Tables	iii
List of Figures.....	iv
Abstract.....	vii
Chapter 1: Introduction.....	1
1.1 Personal Locator Beacon.....	2
1.2 Trauma Detection Personal Locator Beacon.....	5
1.3 Thesis Organization.....	7
Chapter 2: Personal Locator System Overview.....	9
2.1 Theoretical Details.....	9
2.1.1 Signal Processing Chain.....	10
2.1.2 Inter-Integrated Circuit (I2C).....	13
2.2 Minor Details of Sensor Systems.....	14
2.2.1 Skin Conductance Module.....	15
2.2.2 Pulse Oximeter Module.....	16
2.2.3 Temperature Sensor.....	18
2.2.4 Accelerometer Sensor.....	20
Chapter 3: Design Simulation.....	21
3.1 Pulse Oximeter Module.....	21
3.1.1 Principle of Reflectance Oximeter.....	22
3.1.2 Concave Sensor Module.....	23
3.1.3 Sallen-Key Low Pass Filter.....	24
3.1.4 Bandpass Filter and Amplifier.....	27
3.1.5 Breadboard Prototype and Eagle Schematic.....	29
3.2 Skin Conductance Module.....	30
3.2.1 Voltage Regulator.....	31
3.2.2 Wheatstone Bridge and Instrumentation Amplifier.....	31
3.2.3 Low-Pass Filter.....	33
3.2.4 Breadboard Prototype and Eagle Schematic.....	34
3.3 Temperature Sensor System.....	34
3.3.1 Circuit Diagram.....	35

3.3.2 Breadboard and Eagle Schematic	35
3.4 Accelerometer Sensor System	36
3.4.1 Circuit Diagram	36
3.4.2 Breadboard and Eagle Schematic.....	37
3.5 Microcontroller	37
3.6 Power Supply and Battery Charger	38
3.7 Summary	38
Chapter 4: Implementation	40
4.1 Implementation of Skin Conductance Module	41
4.1.1 Device Testing.....	42
4.2 Implementation of Pulse Oximeter Module	44
4.2.1 Pulse Oximeter Programming	47
4.2.2 Device Testing and Calibration.....	49
4.3 Implementation of Accelerometer Sensor System	52
4.4 Implementation of Temperature Sensor System	54
4.5 Algorithm for Alarm Detection.....	55
4.6 PLB Transmitter	58
Chapter 5: Summary and Future Work	59
5.1 Summary.....	59
5.2 Future work	60
References	62
Appendix A: Eagle Schematics	65
Appendix B: Programming.....	68
B.1 Programming of Skin Conductance and Pulse Oximeter Module.....	68
B.2: Programming of Accelerometer and Temperature Sensor	75
Appendix C: Copyright Permission.....	87

List of Tables

Table 2.1 Comparison Between Various Temperature Sensor Types.....	19
Table 4.1: Readings Obtained During Testing	44
Table 4.2 Measured values for the SpO2 experiment	51

List of Figures

Figure 1.1 Personal Locator Beacon: Sequence of events starting with PLB activation to search and rescue operation initiation.....	4
Figure 1.2 Trauma Detection PLB System Block Diagram. 4 physiological parameters (left side of diagram) assess user function	6
Figure 2.1 Signal processing chain for a two-sensor system module	12
Figure 2.2 Signal Processing Chain for the temperature and accelerometer Sensor Chips	14
Figure 2.3 Section Plan of Concave Sensor Module.....	18
Figure 3.1 Pulse Oximeter Module showing main submodule components and the DC and AC signal paths	23
Figure 3.2 Sallen-Key low pass filter with 0.16Hz cut-off frequency	24
Figure 3.3 PSpice simulation: frequency response of Sallen-key low pass filter with 0.16Hz cut-off frequency using a generic op-amp	26
Figure 3.4: PSpice Transient Analysis of Sallen Key low pass filter with 0.16Hz cut-off frequency taking 500mV as offset and 1V as sine amplitude	26
Figure 3.5 Bandpass filter with amplifier with center frequency as 1Hz to detect the AC signal component from the Pulsed Oximeter Module.....	27
Figure 3.6 Bandpass filter PSpice simulation showing the Frequency response for a multiple feedback bandpass filter with an amplifier	28
Figure 3.7 Circuit diagram of Skin Conductance Sensor Module	31
Figure 3.8 PSpice skin conductance simulation for unamplified signal and amplified signal after passing through instrumentation amplifier	33

Figure 4.1 Breadboard prototype of Skin conductance module	42
Figure 4.2 PIC18LF45k50 microcontroller breadboard and connection to the PICKIT3 programmer used to load the code into the chip	42
Figure 4.3 Overlay plot: Graph between skin voltage vs. resistance	43
Figure 4.4 Skin voltage on PuTTY after fifth round of squats.....	44
Figure 4.5 Breadboard prototype of Pulse oximeter module	45
Figure 4.6 Photograph of the pulse oximeter 3D Printed Concave Sensor support.....	46
Figure 4.7 The pulsed signal obtained on channel 1 when the red LED is on	47
Figure 4.8 The pulse signal obtained from the IR LED on channel 1	47
Figure 4.9 Screenshot of readings displayed on serial Terminal (Putty window) during 10 squats.....	49
Figure 4.10 Filter outputs obtained for red LED on the oscilloscope	50
Figure 4.11 Filter outputs obtained from the IR LED on the oscilloscope	50
Figure 4.12 Graph between calibrated SpO2 readings vs. calculated R	52
Figure 4.13 Accelerometer readings obtained when in still-motion	53
Figure 4.14 Accelerometer readings obtained when the accelerometer is in presence of motion	54
Figure 4.15 Temperature sensor (MAX30205) readings obtained in degree Celsius when fingertip was touched on IC surface.....	55
Figure 4.16 Algorithm for Alarm detection, automatic triggering of PLB for Skin conductance, pulse oximeter and temperature sensor.....	56
Figure 4.17 Algorithm for accelerometer sensor.....	57
Figure 4.18 Functional block diagram of 406 MHz beacon showing the basic components: 406 MHz oscillator, UHF card (containing power amplifier, RF modulator) and GPS signal path feeing a UHV antenna.....	59

Figure 5.1 A simple Block diagram of Type-W ELT 406 MHz transmitter.....	62
Figure A.1 Eagle Schematic- Microcontroller basic connections with power supply and battery charger module.	66
Figure A.2 Eagle Schematic- Pulse Oximeter sensor system	67
Figure A.3 Eagle Schematic- Skin Conductance sensor system	67
Figure A.4 Eagle Schematic- Temperature sensor system.....	68
Figure A.5 Eagle Schematic- Accelerometer sensor system.....	68
Figure C1 Copyright permission for Figure 2.3	87

Abstract

This thesis details the development of an automatically-activated personal locator beacon system to facilitate search and rescue of injured users in remote locations. Trauma in the wilderness can result in brain injuries caused by falling from heights, the onset of bad weather condition, cardiological, pulmonary diseased condition, etc. This thesis initially describes the concept of a low-power trauma detection Personal Locator Beacon system and later focuses on detecting the traumatic condition, through four (4) physiological parameters: skin conductance, pulse oximeter, user movement via Freefall and activity/inactivity and impact detection, and finally body temperature. For these four physiological parameters respective sensor circuitry was designed and data from these sensors read and analyzed by a microcontroller, and displayed using PuTTY software on a laptop. Skin conductance and pulse oximeter data was read through the ADC port of the microcontroller and transferred to the laptop using UART communication. Accelerometer and temperature sensor data was read from the sensor using the I2C communication protocol and transferred to the laptop using UART communication. Later this data was analyzed and compared with pre-set threshold values and an alarm sequence initiated. Future work will be needed to implement the alarm system and 406 MHz emergency radio beacon before this system can be commercially viable.

Chapter 1: Introduction

Outdoor activities, such as mountain trekking, bicycling, boating, etc., continue to grow in popularity around the world with many of those engaging in these activities choosing to engage in these activities alone. Unfortunately, a growing number of accidents have been reported in remote location such as (ocean, seas etc.) with the number of victims growing each year [1]. During the period between 1992-2007; 65,439 search and rescue operations were initiated within US national parks, with 2,659 deaths in these remote locations [2]. Moreover, according to statistics from the International Civil Aviation Organization (ICAO); world aircraft accident rates went down from 4.7 million in 2008 to 2.59 million in 2018 [3]. All these scenarios require the deployment of an Emergency Locator Beacon to facilitate immediate rescue. An Emergency Locator Beacon (ELB) is a radio transmitter which broadcasts the location of a missing airplane, ship and hikers in distress and provides the possibility of immediate search and rescue support. When there is an emergency such as an aircraft crash, sinking ship or lost hiker, the ELB transmitter is activated which transmits a continuous radio signal in the appropriate frequency band. This continuous radio signal is then used by search and rescue teams to find the victim/airplane/ship thus rendering immediate aid. There are three main types of ELB and the one that this thesis addresses is one type called the Personal Locator Beacon.

This thesis describes a Personal Locator Beacon (PLB) which aims to provide continuous monitoring of the wearer's physiological parameters to detect any onset of trauma. It must differentiate between traumatic and non-threatening events, and activate when a traumatic event has been detected. This work is the continuation of thesis work performed by Matthew S. Hopper in Dr. Sadow's research group and uses the concept of trauma detection PLB mentioned in Chapter 4 of his thesis to further solve the problem [4]. In fact, this concept was never reduced to practice and this thesis work's aim was to translate the concept from [4] and reduce it to practice, first as a bread-board design and finally as a fully-packaged PLB system.

1.1 Personal Locator Beacon

When people are in remote areas, or first responders are in harm's way, there is always the risk of injury, either from falls from heights resulting in severe head injuries or broken bones, the onset of bad weather and they may be stuck in remote locations without the ability to return on their own [5]. There are case studies from the American Wilderness that involved blunt impact and extreme hazardous environmental conditions as the cause of death of hikers [6]. Moreover, in a questionnaire of Austrian Alpine hikers and skiers, 1,431 hikers and 1,043 skiers in total, 12.7% of the hikers and 11.2% of the skiers were suffering from at least one type of cardiovascular disease [7]. During these situations, when there is often no access to emergency services, having a Personal Locator Beacon can be a life saver.

The Personal Locator Beacon (PLB) is a distress radio beacon, which contains a battery powered radio transmitter which is triggered during an accident to initiate rescue

of the injured individual. It is a small wearable device which is generally used when people go for outdoor activities like hiking in remote locations. There are two types of PLBs; one in which 406MHz transmitter is embedded in it and second in which a 406MHz transmitter and GPS are embedded in it. Whenever there is any emergency situation the PLB is normally activated manually so that a search and rescue team can come, locate the person and provide immediate support. For the PLB without GPS, on powering on the device and deploying the antenna, the device sends a 406MHz radio signal which includes the person's details. This device sends a signal to High Earth Orbit (GEO) satellites which detects that a PLB has been activated and picks up the person's identification, if a person has registered their PLB with some of his contact information. The GEO satellites cannot determine the location of the beacon, so we have to wait for Low Earth Orbit (LEO) satellites to come into position and pick up the signal. Due to the orbital mechanics of LEO satellites it can take up to one hour and thirty minutes for the signal to be within range. Once the LEO satellite picks up the radio signal (406MHz), it then locates the position of the beacon using a Doppler Shift computation [8]. When the location is determined data is sent to a Local User Terminal then to a Mission Control Center and, finally, to the closest Rescue Coordination Center thus starting the search and rescue operation to provide emergency support [8]. When a PLB with GPS is activated, it not only sends the person's identification information but also their GPS location to the GEO satellites thus reducing the work of the LEO satellites and initiating the search and rescue operation immediately [8].

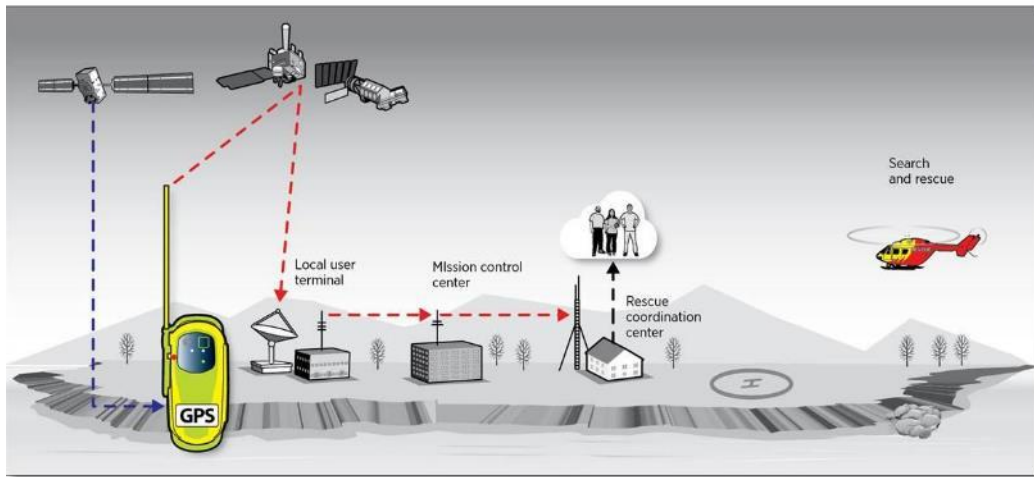


Figure 1.1 Personal Locator Beacon: Sequence of events starting with PLB activation to search and rescue operation initiation [9].

At times there are situations when a person is unable to activate the PLB due to several reasons: severely hypothermic due to bad weather conditions, unconscious due to a fall resulting in head injury due to blunt impact, has suffered a stroke and other traumatic condition, etc. In this unconscious state a user is obviously unable to activate the PLB manually and automatic activation is required. Automatic activation involves two key parts: First detection of a life-threatening injury or situation, and, second, automatic activation of the 406 MHz transmitter to request immediate assistance from a rescue team. This thesis proposes to reduce this concept to practice and design, build and test a PLB system that automatically senses when a traumatic event has occurred. To detect traumatic injury, we will monitor four (4) physiological parameters from the user: skin conductance (heat stroke, etc.), pulse rate and blood oxygen concentration (exertion and

cardiovascular/pulmonary issues), body temperature (heat stroke, exertion) and blunt impact (trauma such as a severe fall, gunshot wound, etc.).

1.2 Trauma Detection Personal Locator Beacon

The Trauma detection Personal Locator Beacon is a conceptual device which will be used by people having no close proximity emergency access and by emergency responders who are often working in life-threatening conditions (firefighters, police, etc.). The main purpose of the device is to monitor a few key physiological parameters continuously, estimate the health condition of the user and automatically activate the PLB if any life-threatening or traumatic event is detected [4]. This Personal Locator Beacon will monitor four physiological parameters, requiring four separate sensor modules. They are:

1. Pulse Oximeter Module: This module will monitor pulse rate and blood oxygen concentration of the human body to detect cardiovascular malfunction.
2. Electrodermal Activity Module: This module will measure sweat level of a person using Galvanic Skin Response; this is done by measuring skin conductance of the body to determine heat stroke, hypothermia and other health conditions.
3. Freefall and Impact Detection Module: To detect injuries from either freefall or blunt trauma impact, Impact detection will be measured using a 3-axis Digital Accelerometer.
4. Body Temperature Module: Human body temperature will be measured to assess the overall condition on the body. This will allow to keep a check on health conditions like over exertion, heat stroke, hypothermia, etc.

On measuring these physiological parameters, if any parameter value goes above or below a specific threshold value, the PLB will be activated on its own. On detection of a life-threatening event, the device will first activate a quiet alarm which is audible to the person wearing the PLB and people within the immediate vicinity. This alarm will ring for 2 minutes at low volume, followed by another alarm for 8 minutes at high volume. The 8 minute alarm will alert people who are in the range of a few miles before the PLB activates on its own [4]. A 10-minute alarm is set into the PLB to check the severity of the traumatic event; If the person wearing the trauma detection Personal Locator Beacon becomes conscious, or is able to help themselves, they can thus prevent/cancel automatic triggering of the PLB. If the user cannot help him/herself, then people nearby can render assistance, thus saving time and cost of a search and rescue operation from being conducted.

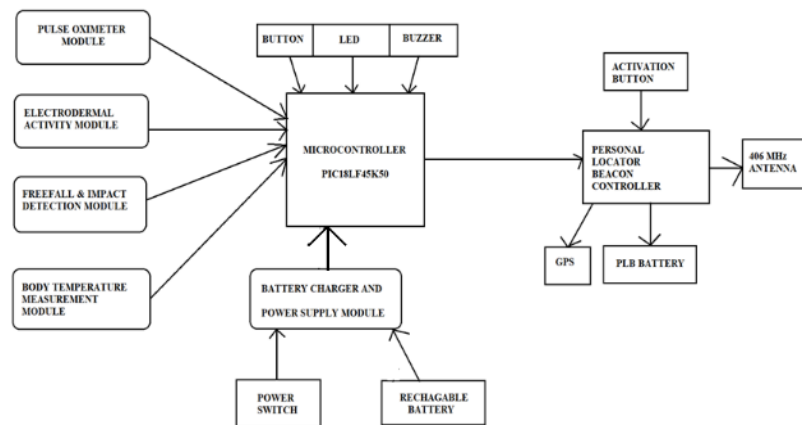


Figure 1.2 Trauma Detection PLB System Block Diagram. 4 physiological parameters (left side of diagram) assess user function. All data is fed into the system microcontroller (center) including user input, power system and PLB system which alerts emergency responders via a 406 MHz radio beacon (right side of diagram).

The trauma Detection PLB will be designed to be a low power device which is wrist wearable. The whole system has 4 sensor modules which are embedded in it operating at low power and transmitting data to a low power PIC microcontroller (model No. PIC18LF45k50). The system contains a long-life battery charger and power supply module. This module provides a 3.3V supply voltage to the system from a high long-life lithium ion (Li-ion) battery. It also includes a battery charger to charge the Li-ion battery when its voltage drops below a specified level. Furthermore, this system contains 2 batteries; a rechargeable battery which is used to supply power to the microcontroller and four sensor modules and a second non-rechargeable battery which is used to activate the PLB, supply power to PLB controller and deploy the antenna and GPS is used. It also has 2 activation buttons; the first button is used to disable the alarm and second button is used for manual activation of the PLB. Finally, the system has 2 alarms, one low volume which is used to alert the person if conscious to seek help on their own or receive help from immediate nearby people. The second alarm has a high volume which is used to alert people who are within a few miles range from the victim.

1.3 Thesis Organization

In chapter 2, certain reasons are explained to why particular physiological parameters are being used. Later, signal processing chain which is common to all sensor system is explained. This signal processing chain is divided in two parts i.e. signal processing chain for two sensor modules and then signal processing chain for temperature and accelerometer sensor module. In the I2C communication protocol, signal processing chain for temperature and accelerometer sensor is explained followed by brief detail on

I2C communication protocol. Further, minor details of each sensor is explained individually focusing on some previous work followed by current approach implemented.

In Chapter 3, technical details of each sensor system are described. PSpice simulation of pulse oximeter, skin conductance is shown. Breadboard and Eagle schematic details of each sensor system are described. At last, the microcontroller, power supply and battery charger module used is described.

Chapter 4 discusses about programming the device with PIC18LF45K50 microcontroller. For programming skin conductance and pulse oximeter, Analog to digital conversion and USART communication is used. For accessing data from temperature sensor and accelerometer, I2C communication is used and USART protocol is used to transfer data on desktop. At last device testing is explained followed by results.

Chapter 5 concludes this thesis research work and discusses the future aspects of this work.

Chapter 2: Personal Locator System Overview

As discussed in the previous chapter, this thesis involves the monitoring of a few key physiological parameters to monitor the physiological condition of the user and automatically notify emergency responders if need be. According to the statistical data in the *Whats's Killing America's Hiker's* blog, [10] the main cause of death is due to severe physical conditions suffered while hiking. This may include cardiovascular malfunction, pulmonary malfunction, over exertion, heat stroke, hypothermia, etc. The second main cause for a hiker becoming incapacitate is due to falling from heights or into steep holes, pits, etc. [10]. In this thesis, we are measuring four (4) physiological signals to measure certain physical conditions and head injuries (blunt impact) which contribute to the main cause for a hiker being lost and/or dying in the wilderness. The physiological signals being measured are pulse oximeter, skin conductance, freefall and blunt impact, and body temperature. These parameters were selected after conducting a thorough research on the cause of hiker's death as mentioned in [5],[6],[7], [10].

2.1 Theoretical Details

This chapter reviews a few theoretical details on the approach to extract the physiological signals mentioned above. For this, the system signal processing chain is explained which is used in each sensor module, albeit with differences in details and implementation. Then I2C communication concept is explained which is typically used in

two sensor systems. Later minor details of each sensor system are explained. In the details of each sensor module the previous work for each individual sensor module is reviewed and the current approach is introduced.

2.1.1 Signal Processing Chain

The signal processing chain elaborated in Figure 2.1 is identical to that used in the two sensor systems involving pulse oximeter and skin conductance). Each sensor module follows the same signal processing chain, it only differs in details and in its implementation, resulting in differences in calculation used to extract the relevant physiological parameter. The first step of the signal processing chain involves the acquisition of the physiological signal from the body. This can be done from various parts of the body; chest, finger, toes, elbows, wrist, etc. In this thesis, the measurement site selected is wrist. The physiological signals acquired from the body are generally in analog form which are very small in magnitude to read. They also contain unwanted noise in the signal. For this purpose, the analog signal is conditioned which helps in removing noise and increasing the signal to noise ratio. The signal to noise ratio (SNR) is the ratio between the electrical signal strength to the present noise/interference level. If the SNR is higher it improves the readability of the signal: it can be digitalized and analyzed more accurately with minimal erroneous errors. To condition an analog signal, amplification followed by filtration of the signal is done. Moreover, bandwidth limitation is also an important aspect for filtering and digitalizing the signal. According to the Nyquist-Shannon sampling theorem, the sampling rate at which the signal is digitized needs to be more than twice as high as the maximum frequency occurring in the signal ($f_s > 2f_{max}$); where f_s is called the

Nyquist frequency. This theorem ensures that no signal is lost if a signal is sampled above Nyquist frequency and no additional information is gained by sampling faster than this rate. If this theorem is violated, an aliasing effect is seen in the output signal. Aliasing causes different signals to become indistinguishable when sampled thus introducing errors in the signal. To avoid aliasing, these signals need to be bandwidth limited before digitalization. Many times, there are frequencies which are above Nyquist frequencies: these frequencies can either be part of the target signal or can be unwanted noise. To limit unwanted aliasing effects to a certain limit that the signal is sufficient for a required application, these signal components need to be attenuated. This can be done by using a low, or band pass, filter. A simple first order low pass filter consists of one resistor and capacitor connected in parallel which can sufficiently attenuate the signal and provide the desired output. For complex circuits, more complex n^{th} order filters can be designed. The bandwidth is limited depending on the actual signal characteristics and amount of noise (i.e., SNR) and is evaluated individually for two sensor systems which will be explained in more detail in Chapter 3.

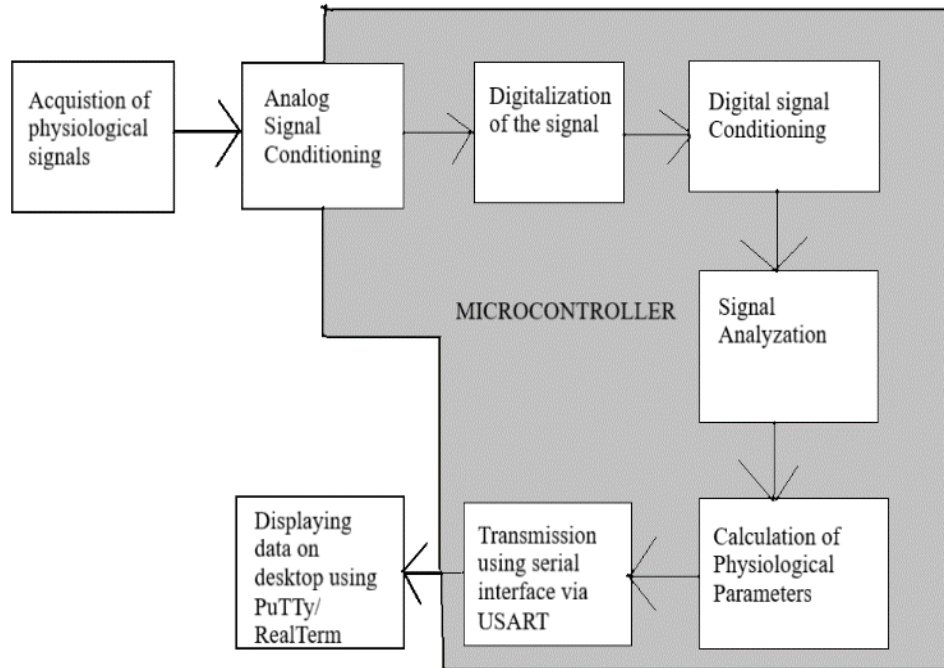


Figure 2.1 Signal processing chain for a two-sensor system module.

The third step in the signal processing chain is digitalization: it needs to be performed by using a sampling rate which it meets the Nyquist criteria, i.e. accounts for any bandwidth limitation of the signal. The fourth step is digital signal conditioning is optional and dependent on the necessity for the respective sensor. If this step is performed then the signal is filtered again before it is analyzed. The final step is signal is analyzed/the actionable parameter calculated which requires an algorithm which extracts certain features from the signal depending on the physiological parameter being monitored. On extracting a target feature, calculation of the desired physiological parameter is done. Next the computed value is transferred to a device to display the output on computer screen using serial communication known as USART and a USB. The data is displayed on a serial

monitor (serial port) using software like PuTTY, RealTerm, Arduino IDE etc. As in Figure 2.1, most of the steps in the signal processing chain are performed inside a single integrated circuit (IC) chip called the microcontroller: these steps start from the analog signal conditioning stage to transmitting the data via USART to a USB device. These steps are marked by a light grey background in Fig. 2.1.

2.1.2 Inter-Integrated Circuit (I2C)

In this thesis work, there are two sensor modules which use the concept of an Inter-Integrated Circuit (I2C) bus to communicate with the microcontroller. In the previous section, the signal processing chain was explained which follows the process of analog signal conditioning to serial interface using USART protocol in a single integrated chip known as a microcontroller. But for 2 sensor modules, namely the temperature and accelerometer sensors, analog signal conditioning followed by digitalization, digital signal conditioning, signal analysis and calculation of the physiological parameter is done in these single respective sensor module chips. Later, these sensors are connected with the microcontroller using I2C communication which allows system access to this data. This data is transferred to the microcontroller where it is further stored using I2C communication and is transferred to a serial port using the USART protocol. The communication between the sensor module and microcontroller is done using the I2C bus; which is a multi-master serial data communication protocol. Using this communication master device the microcontroller initiates the communication and the slave device(s), which are the sensor IC(s), communicate through an addressing protocol. The I2C bus has 2 specific connections: first a Serial Clock (SCL) and second a Serial Data (SDA)

connection. Both the SCL and SDA lines are bidirectional and each requires the use of pull-up resistors connected to the supply voltage. Filtering of the signal is often done by the use of decoupling capacitors to filter unwanted noise (i.e., simple LC filter). In the case of the temperature/accelerometer sensor chips almost all signal processing is done in the sensor chip itself shown in Fig. 2.2. The I2C bus is responsible for accessing the data, storing it in the microcontroller as shown.

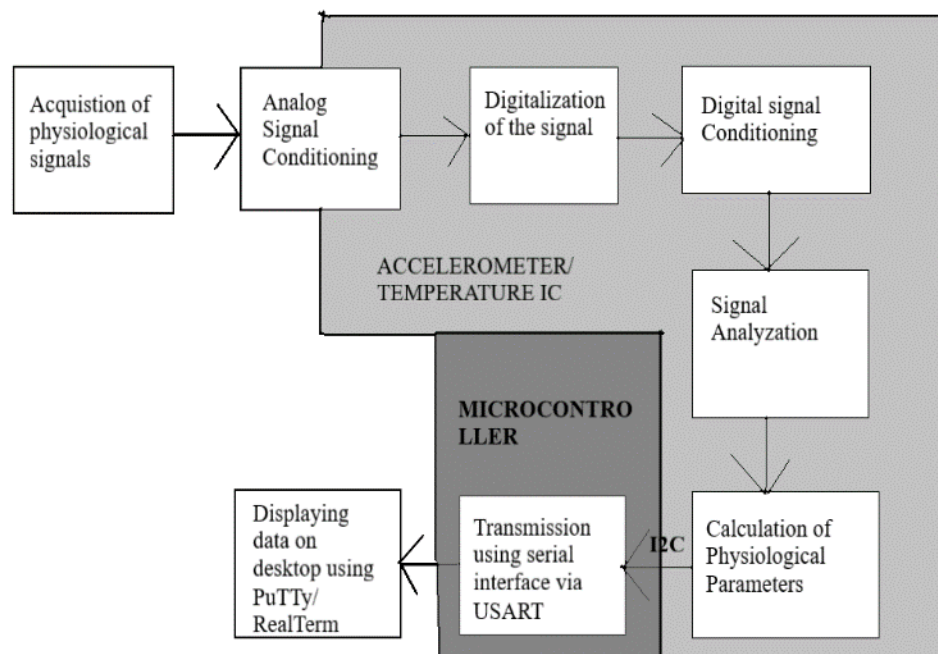


Figure 2.2 Signal Processing Chain for the temperature and accelerometer Sensor Chips.

2.2 Minor Details of Sensor Systems

In the following section, minor details of each sensor module used in this work are explained. Each sensor module is described with previous work followed by discussion of the methodology adopted in this work.

2.2.1 Skin Conductance Module

Skin conductance is typically measured using several approaches. Many use the constant voltage method for which circuits are provided with Beckman, Grass and other common polygraphs for signal conditioning [11]. Moreover, interdigitated array (IDA) microelectrodes have been used for measuring skin conductance [12] and nickel electrodes connected to an electronic board managed by a computer were also used to measure skin conductance on hands and feet [13]. In addition, the “Galvactivator glove” has also been used for measuring skin conductance [14]. Furthermore, a Biosensor Array Galvanic Skin Response sensor has also used to measure skin conductance [15]. This circuit is also known as the Truth Meter circuit which uses a low and high pass filter to create 0.48Hz to 4.8Hz bandpass filter and does not use a microcontroller thus reducing the complexity of circuits [15].

In this thesis, we are measuring subject skin conductance using Silver/Silver chloride (Ag/AgCl) electrodes with an increased contact area and an isotonic wet gel to improve performance. The measurement site is the wrist as it will be easy and comfortable for the user to wear the PLB on this location of the body. A low voltage of 0.5V is applied to two electrodes and the resistance between the electrodes measured. The resistance is the reciprocal of the conductance, hence when the resistance between two electrodes is measured they are easily converted to skin conductance. The SI unit of conductance is Siemens (S) and skin conductance is measured in micro Siemens (μS). A Wheatstone bridge configuration is used to measure the skin resistance i.e. resistance between these two electrodes. For this, the supply voltage applied to the Wheatstone bridge is limited to

a maximum of 0.5V, because the conductivity of human skin is almost linear at this voltage. Moreover, voltages applied to the human body should always be low for safety reasons. For this purpose, the SC4216H voltage regulator was used to provide 0.5V supply to the Wheatstone bridge. The differential output of the Wheatstone bridge is then amplified by an instrumentation amplifier and filtered using a first-order low pass filter to remove high frequency noise. Since the conductance of the skin changes slowly, the cut-off frequency of the low pass filter was set to 1.6Hz: which is good enough to filter high frequency components.

2.2.2 Pulse Oximeter Module

A pulse oximeter is a non-invasive device which is used for measuring heart rate and the concentration of oxygenated blood in the human body. People who have cardiac or pulmonary disease need to monitor their blood oxygen concentration and pulse rate especially when engaging in outdoor activities such as jogging, exercising, etc. Hiking is one of the outdoor activities in which hiker's die due to cardiac or pulmonary malfunction. Having a pulse oximeter sensor while hiking reduces the risk of cardiac or pulmonary malfunction and is thus an important component of our PLB.

There are two types of hemoglobin. The hemoglobin with oxygen is called oxygenated hemoglobin (oxy Hb), whereas hemoglobin without oxygen is called deoxygenated hemoglobin (deoxy Hb). A pulse oximeter sensor uses LED lights to measure the presence of both Hb forms via optical absorption. This value is then used to calculate the amount of oxygenated blood in the human body. For this, a red LED of 630 nm wavelength and an Infrared LED of 880 nm wavelength are used. Oxy Hb absorbs more

infrared light whereas deoxy Hb absorbs more red light. The pulse oximeter estimates oxygen saturation by calculating the ratio of red light absorbed in oxygen to that with infrared light.

Currently, there are many different pulse oximeters available which vary in size, quality and cost. The most common model is the finger model, in which the measurement site used is the fingertip. A wrist finger model was patented [16] in which at least one light source and light detector are used. The data of this wrist model is displayed on the screen worn on the wrist. Later, a small-size reflective pulse oximeter sensor design was studied to keep the design optimal [17]. In this paper, optimal distance between the light source and the detector were measured as well as the optimal pressure which needs to be applied to press a reflective pulse oximeter sensor head calculated. In the paper titled “A Reflectance Pulse Oximeter Design Using the MSP430F149” [18], an MSP430F149 microcontroller was used to calculate the pulse rate and oxygen saturation in the blood apart from the sensor module. The sensor module consists of red and infrared LED lights of 660 nm and 890 nm each.

In this thesis work, we implement a pulse oximeter using the reflectance oximetry principle. Though a reflectance oximeter can be used to measure pulse rate and oxygen saturation on the finger, ear, chest, wrist, etc., for a hiker or first responder the wrist is the most convenient and comfortable position for long duration and continuous monitoring of pulse rate. In this thesis work, a concave structure sensor module as in [19] was designed to mount the red LED of 630 nm and infrared LED of 880 nm which are placed on same

side of the wrist. The sensor module is designed to be placed at the inner part of the wrist. This is done to improve the operational efficiency when collecting the wrist pulse signals.

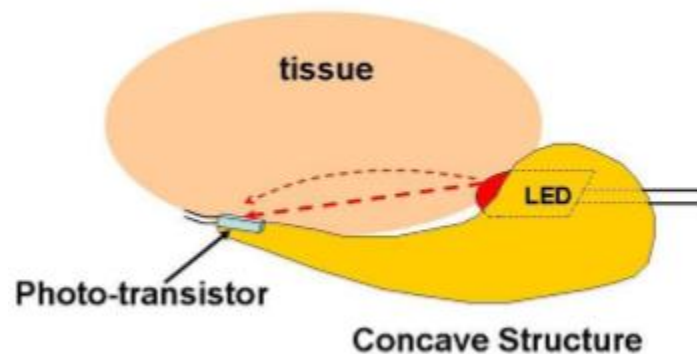


Figure 2.3 Section Plan of Concave Sensor Module [19]. Copyright permission for this picture is mentioned in Appendix C.

The concave design helps the sensor module to stay in the chosen location of the wrist. Once the signals are extracted from the built-in phototransistor, it is then passed through a low pass filter to filter higher frequency noise and extract DC component of the signal. The signal is also filtered using a bandpass filter to extract its AC component. The filtered AC component is very small in magnitude, so it is amplified and further these two AC and DC components are fed into the ADC pins of the microcontroller to digitize the signal and thus calculate the pulse rate and saturated oxygen level.

2.2.3 Temperature Sensor

Monitoring human body temperature is a vital parameter when monitoring physiological conditions of a user while engaged in outdoor activities. Previously, human body temperature has been measured using various temperature sensors such as a temperature sensor IC, thermistor, thermocouple and resistance temperature detectors

(RTD). Each of these temperature sensors has their own pros and cons which are listed in Table 2.1.

Table 2.1 Comparison Between Various Temperature Sensor Types.

Criteria	Temp Sense IC	Thermistor	RTD	Thermocouple
Temp Range [C]	-55 to 150	-100 to 500	-240 to 700	-267 to 2316
Accuracy	Meets requirements	Depends on Calibration	Meets requirements	Depends on cold junction compensation
Linearity	Best	Least	Better	Better
Sensitivity	Better	Best	Less	Least
Circuit Simplicity	Simplest	Simpler	Complex	Complex
Power	Lowest	Low	High	High
Cost	\$	\$-\$\$\$	\$\$\$	\$\$

Based on the tradeoffs presented in Table 2.1, the temperature sensing IC was selected as the best temperature sensor for the PLB. In this thesis, we are measuring human body temperature using a MAX30205 temperature sensing IC which accurately measures the temperature meeting clinical thermometry specifications of ASTM1112. This IC converts the temperature readings to digital form using a high resolution, sigma-delta, analog-to-digital converter (ADC). Moreover, this device communicates with the microcontroller using the I2C serial communication protocol thus providing access to the microcontroller to store and read temperature measurements in digital form.

2.2.4 Accelerometer sensor

In the past, various accelerometer ICs such as the LIS3DH, AN3151, AN3459, etc. have been used to measure free-fall and blunt impact caused due to falling from height, falling into holes, pits, etc. In this thesis we are measuring free-fall and blunt impact using a 3-axis digital accelerometer from Analog Devices (part number ADXL345). This accelerometer features a selectable measurement range of $\pm 2\text{-g}$, $\pm 4\text{-g}$, $\pm 8\text{-g}$, or $\pm 16\text{-g}$. It has various built in special sensing functions. It can detect the presence, or lack, of motion by comparing the acceleration in any axis with the initial user acceleration, called the threshold acceleration. For this it activates an activity, or inactivity, function and can also detect free-fall motion of a body by sensing a free-fall condition. It has a 'tap' sensing function which detects single or double taps in any direction. It is an ultralow power device which communicates and sends data to the microcontroller using either I2C or SPI serial communication. When it senses any change in motion it converts the analog values to digital using its high-resolution analog to digital converter (ADC) and sends the data to the microcontroller in digital form. With this sensor we are measuring any changes in the motion of a body in 3 axes, and can not only monitor the user's direction of motion but detect freefall as well as inactivity.

Chapter 3: Design Simulation

In this chapter, technical details of the proposed personal locator beacon approach are explained. On the basis of the signal processing chain described in Chapter 2, this chapter deals with all the technical details such as the hardware setup, components used, PSpice circuit simulations, breadboard prototype and Eagle schematic PCB layout for the individual sensor modules. Skin conductance and pulse oximeter modules were simulated using OrCAD PSpice Designer Lite software with OrCAD Capture demo version. The exceptions to PSpice simulation are the accelerometer and temperature sensors. These sensor modules were implemented on a breadboard using respective sensor breakout boards and then Eagle schematics were laid out for PCB implementation using surface mount technology. Later, the microcontroller, power supply and battery charger module technical details, hardware setup, breadboard prototype and Eagle schematic are presented.

3.1 Pulse Oximeter Module

The pulse oximeter, as the name suggests, is responsible for calculating pulse rate and blood oxygen concentration in the human body. This sensor helps in the detection or cardiac or pulmonary malfunction such as Bradycardia, Tachycardia, sudden cardiac arrest, hypoxia, heat stroke, hypothermia, etc. For measuring pulse rate and blood oxygen concentration, we designed a concave sensor module which consists of one concave

support structure, one red LED, one infrared LED and one phototransistor. The concave sensor module acquires the signal from the blood and tissue in the wrist using the principle of reflectance oximetry. The signal acquired from the phototransistor is then processed (filtered and amplified) to separate the AC and DC components. Later, the analog signal is converted into a digital signal using the built-in ADC feature of the microcontroller and the output is obtained and displayed on a desktop PC.

3.1.1 Principle of Reflectance Oximeter

Due to light scattering in tissue photon transport in the human body is considered as a diffusion process [19]. The light passing through wrist tissue using red and IR LEDs can be described by a photon diffusion equation as mentioned in [19]. In this situation, the density of the photons is determined by the flux of illumination and the detected light intensity. As mentioned in [19], the change ratio of the reflected light intensity. $W = I_{ac}/I_{dc}$ is proportional to the absorption coefficient of the tissue. This ratio is used to calculate the SpO₂ level in the human body. The SpO₂ level can be estimated by calculating the ratio of the amount of light, at different wavelengths, absorbed in the blood. In the formula $W = I_{ac}/I_{dc}$, the AC component represents a fluctuating wave induced by pulsing blood while the DC component is the absorption of human tissue and vessels. For calculating the SpO₂ level R, the ratio of reflective light intensity between two different wavelengths, is calculated as $R = [I^1_{AC}/ I^1_{DC}] / [I^2_{AC}/ I^2_{DC}]$ and the SpO₂ level calculated using $SpO_2 = \alpha - \beta R - \gamma R^2$ (3.1)

In equation 3.1, α , β , and γ are calculated experimentally. Pulse rate can be calculated by measuring the elapsed time between the peaks of the IR signal and then computed using the formula $BPM = 60/Period \text{ (seconds)}$ [20].

3.1.2 Concave Sensor Module

The concave sensor consists of a concave support structure, one red LED (630nm wavelength), one IR LED (880nm wavelength), and one phototransistor mounted on the other end of the structure. The 2 LEDs are placed on same side and inserted in 2 different mounting holes in the support. The phototransistor is placed directly in contact with the skin and has a spectral range from 400 - 1100 nm which covers the visible and near IR wavelength range, thus giving better signal accuracy. Once the signal is acquired from the concave sensor module it is then processed into separate AD and DC components. This separation is done by passing the signal through a low pass filter to obtain the DC component of the signal. To obtain the AC component, the signal is passed through the bandpass filter and then the amplifier. This is illustrated below in Fig. 3.1.

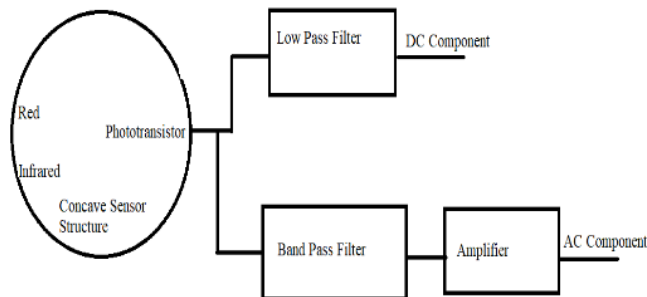


Figure 3.1 Pulse Oximeter Module showing main submodule components and the DC and AC signal paths.

3.1.3 Sallen-Key Low Pass Filter

The DC component of the signal can be obtained from a low pass filter. For this a low pass filter having a cut-off frequency of 0.16Hz was designed to filter the fluctuating pulse wave (i.e., AC) signal and high frequency noise such as electromagnetic interference from the power source or the environment. Figure 3.2 shows the Sallen-Key low pass filter topology.

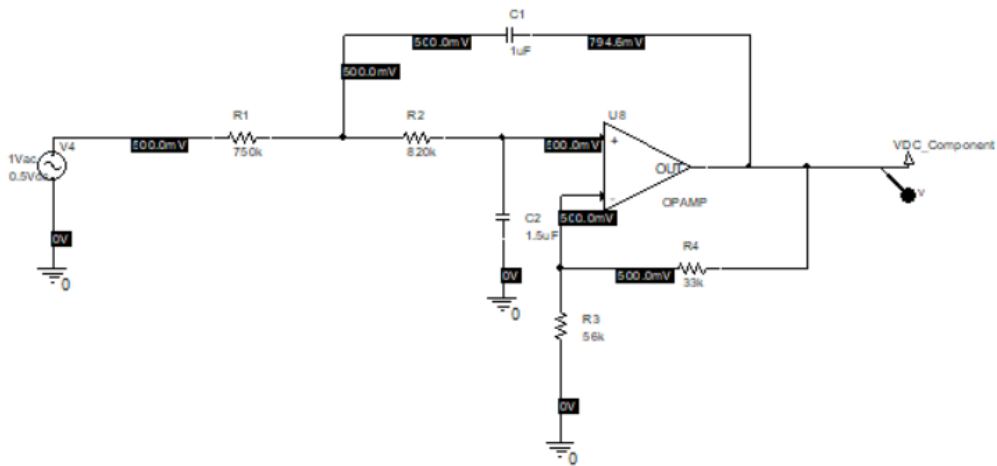


Figure 3.2 Sallen-Key low pass filter with 0.16Hz cut-off frequency

Using the Sallen-key topology in Figure 3.2, the transfer function for the following circuit can be calculated using equation 3.3. Then, assuming a gain of 1.59V/V, R3 and R4 can be calculated from (equation 3.4) as 56kΩ and 33kΩ, respectively.

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{\omega_o^2}{s^2 + 2\zeta\omega_o s + (\omega_o)^2} \dots\dots\dots (3.3)$$

$$G = \frac{R3+R4}{R3} \dots\dots\dots (3.4)$$

$$f_c = \frac{1}{2\pi RC\sqrt{mn}} \dots\dots\dots (3.5)$$

$$Q = \frac{\sqrt{mn}}{m+1+mn(1-k)} \dots\dots\dots (3.6)$$

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{2\pi f_c}{(2\pi f_c)^2 + 4\zeta\pi f_c s + (2\pi f_c)^2} \dots\dots\dots (3.7)$$

Considering these equations and considering $R_1=mR$, $R_2=R$, $C_1=C$ and $C_2=nC$, the cut-off frequency of the filter can be formulated as mentioned in (equation 3.5). Later assuming m , n and K as 0.91, 1.5 and 0.705, respectively, Q can be calculated from (equation 3.6) as 0.507. Then setting the cut-off frequency to 0.16Hz and assuming C , and knowing the m and n values, R can be calculated from (equation 3.5). The calculated R and C values for the Sallen-key low pass filter are $R_1=750k\Omega$, $R_2=820k\Omega$, $C_1=1\mu F$ and $C_2=1.5\mu F$, respectively. The simulated low pass filter response using these component values is shown in Fig. 3.3 below.

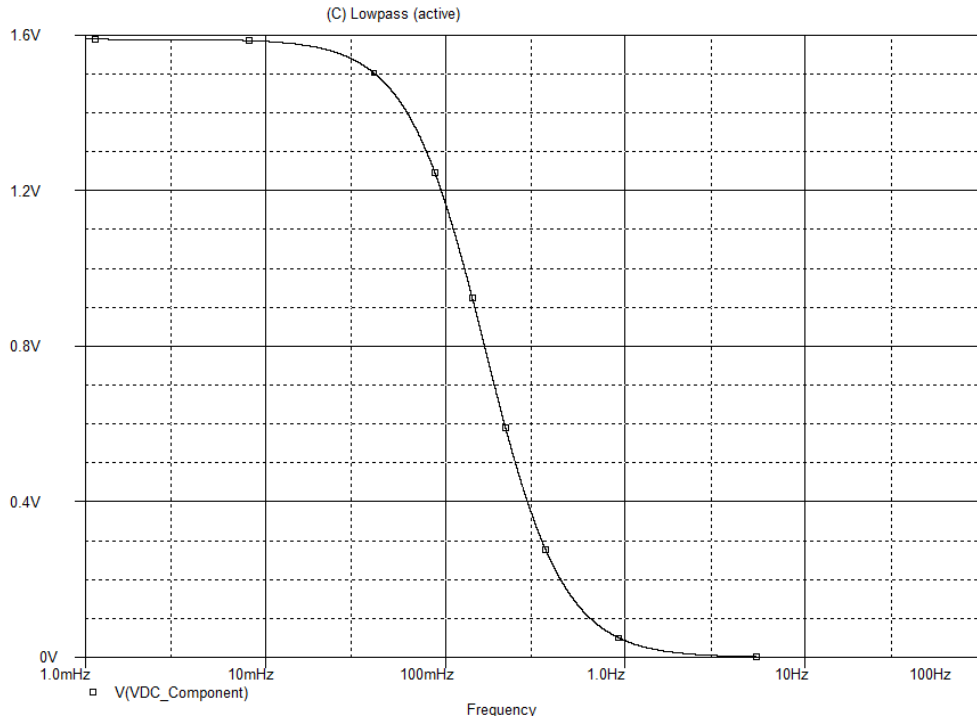


Figure 3.3 PSpice simulation: frequency response of Sallen-key low pass filter with 0.16Hz cut-off frequency using a generic op-amp.

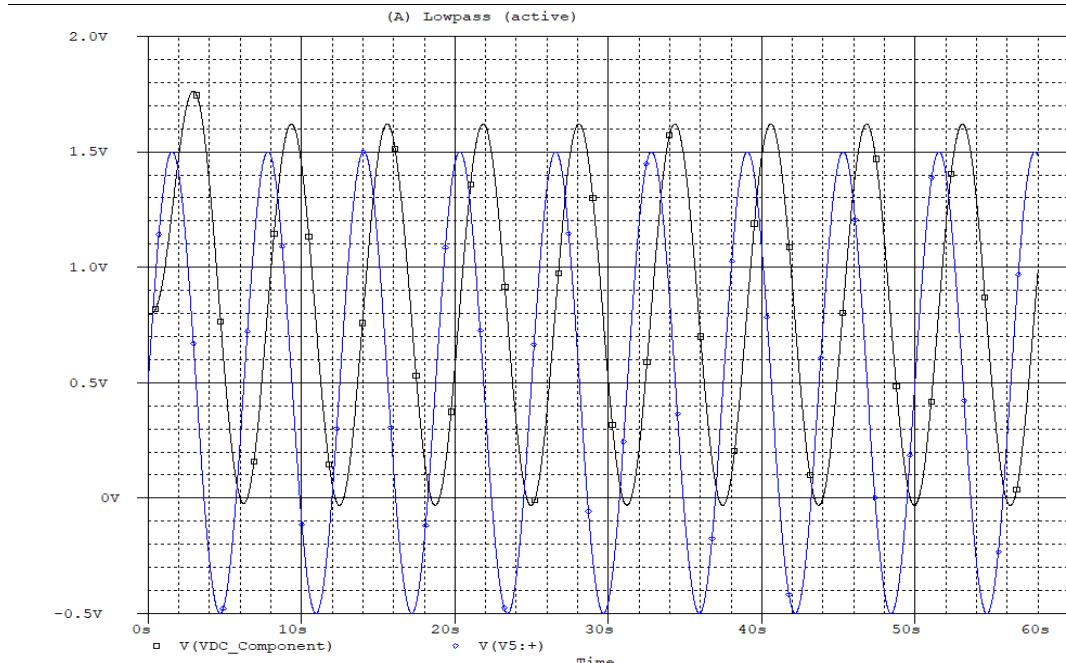


Figure 3.4: PSpice Transient Analysis of Sallen Key low pass filter with 0.16Hz cut-off frequency taking 500mV as offset and 1V as sine amplitude.

3.1.4 Bandpass Filter and Amplifier

To obtain the AC component from the acquired signal, a bandpass filter with an amplifier was designed. The center frequency was set to 1Hz since the normal pulse frequency of a human body is close to 1Hz [19]. The bandpass filter filters the high frequency noise such as environmental interference from the power source, wireless signals, etc. The output obtained from the bandpass filter is generally ~10mV which is not sufficient to be read by the microcontroller, hence the signal was amplified for further processing. The circuit diagram for amplified bandpass filter is shown in figure 3.4.

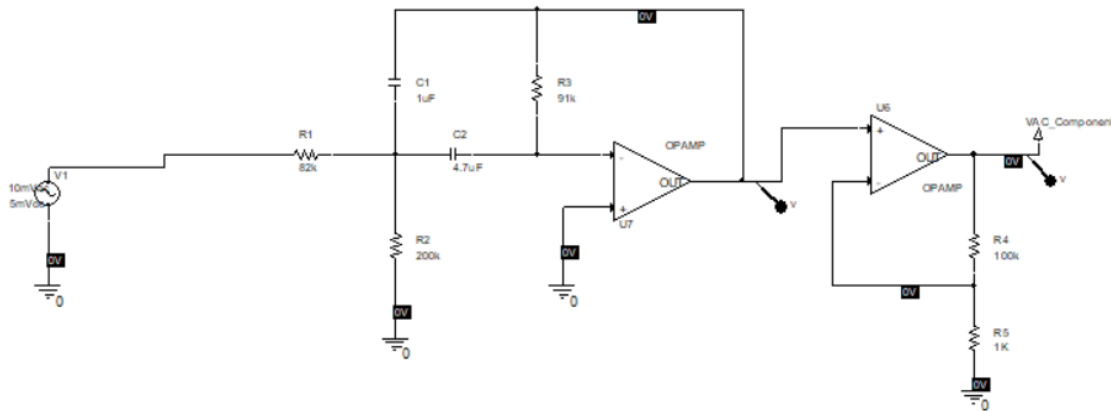


Figure 3.5 Bandpass filter with amplifier with center frequency as 1Hz to detect the AC signal component from the Pulsed Oximeter Module.

When looking at figure 3.4 the transfer function for this circuit is given by equation 3.7. Taking the center frequency as 1Hz and assuming $C1=1\mu\text{F}$ and $C2=4.7\mu\text{F}$, the circuit is solved for R1, R2 and R3 where K is gain at the center frequency.

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{2\zeta\omega_0 Ks}{s^2 + 2\zeta\omega_0 s + (\omega_0)^2} \dots\dots\dots (3.7)$$

$$\omega_o = 2\pi f_o = \sqrt{\frac{1}{R_3 C_1 C_2} \left(\frac{1}{R_1} + \frac{1}{R_2} \right)} \dots\dots\dots (3.8)$$

$$Q = \frac{1}{2\zeta} \dots\dots\dots (3.9)$$

$$K = \frac{R_3 - C_2}{R_1 C_1 + C_2} \dots\dots\dots (3.10)$$

Amplifying the signal from (10-30) mV to 3V a non-inverting amplifier configuration was used with a gain of 100V/V. Equation 3.11 calculates the closed loop gain of the inverting amplifier:

$$G = \left[1 + \frac{R_4}{R_5} \right] \dots\dots\dots (3.11)$$

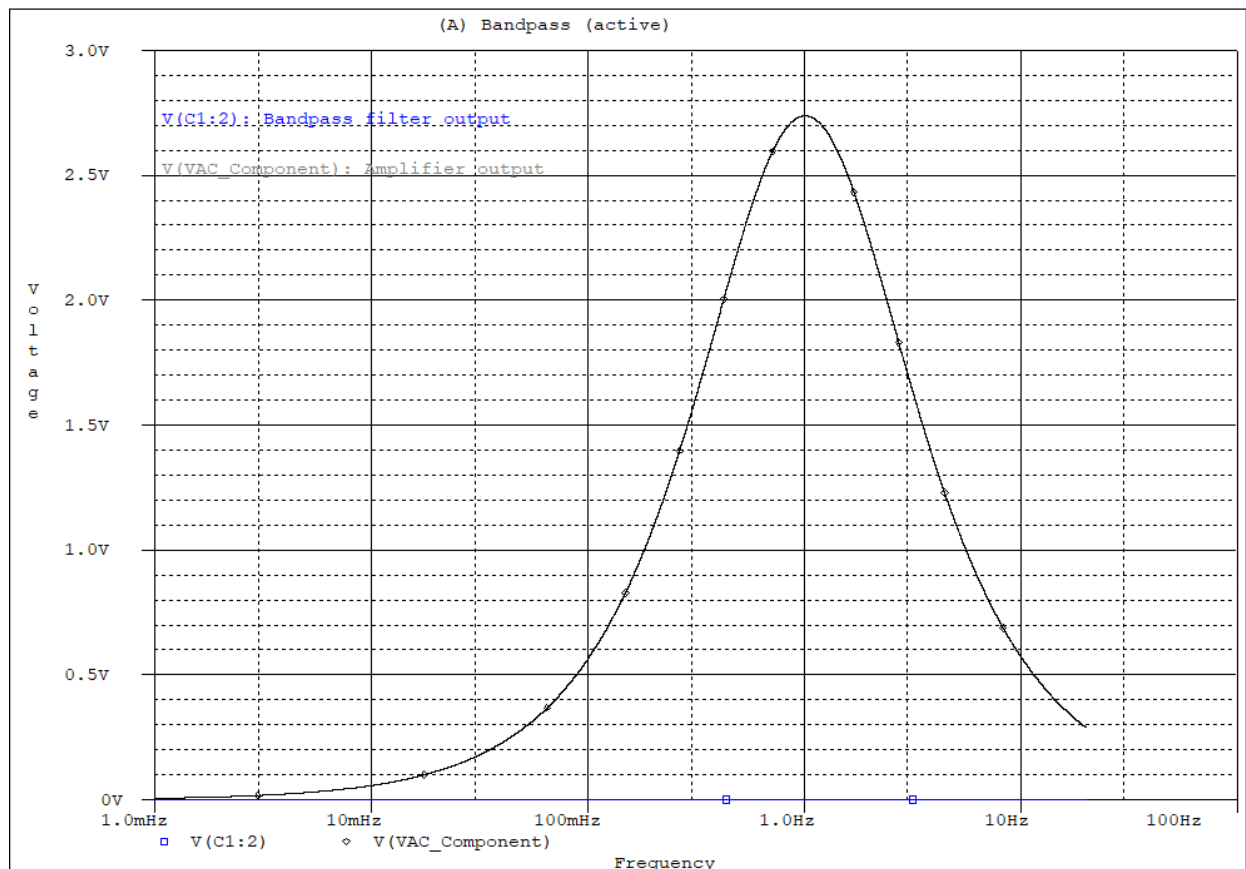


Figure 3.6 Bandpass filter PSpice simulation showing the Frequency response for a multiple feedback bandpass filter with an amplifier. Center frequency of the signal is taken as 1Hz and simulated using a generic op-amp.

3.1.5 Breadboard Prototype and Eagle Schematic

In the PSpice simulation, a generic op-amp was used in place of an advanced operational amplifier IC. For the breadboard prototype an MCP604 operational amplifier IC was used in the low pass filter, bandpass filter and inverting amplifier circuits. This IC is from the family of low-power operational amplifiers with a single supply (2.7V to 6.0V) and in a quad op-amplifier configuration. This IC was selected as it provides 4 operational amplifier configurations with 4 channels. It was also selected because it is used as a photodiode pre-amplifier and in data acquisition circuits. Because of its quad op-amp configuration, only one IC was used thus reducing the complexity and size of the circuit. A red LED (630nm wavelength) based on AllnGaP technology from Vishay Semiconductors was purchased. An IR LED (880nm wavelength) using GaAlAs was purchased. For the phototransistor, the same spectral range as the red and Ir LEDs was selected (i.e., covering 630 – 880 nm). For the breadboard the supply voltage to the MCP604 IC was set to 3.3V using a Tektronix PS2520G programmable power supply device.

For the PCB design, Eagle software was used. All the resistors, capacitors and MCP6044 IC will be soldered using surface mount technology. For the passive Surface Mount Device (SMD) 0805 package resistors and 0805 package capacitors were selected. For the operational amplifier, a MCP6044 was purchased in a Small Outline Integrated Circuit (SOIC) package. This IC is a quad operational amplifier that operates with a single supply voltage as low as 1.4V and draws less than 1uA of quiescent current per amplifier. It has a gain-bandwidth product of 14kHz and a stable unity gain which makes it

appropriate for low frequency applications. For connecting the red and IR LEDs and the Phototransistor (concave support structure) with the PCB male jumper headers were pinned out in the Eagle Schematic. Details of the Eagle Schematic for the pulse oximeter module are available in Appendix A.

3.2 Skin Conductance Module

Skin conductance is linear up to 0.5V on the skin, i.e. the voltage drop across two points on the skin must be less than or equal to 0.5V [21]. For this reason, voltage supplied to the circuit needs to be 0.5V. Resolution of the skin conductance needs to be high. If the relationship between the output voltage and skin conductance is linear, it will have better resolution [21]. The typical range of human skin conductance is 1uS to 20uS [22]. But when measuring this parameter, it was found that the skin conductance at the wrist varies from 0.1uS to 10uS. For measuring this range, AgCl electrodes were placed at the wrist of 10 different subjects and their skin resistance (resistance between the electrodes) were measured using multimeter. Later, considering all of these design considerations, the skin conductance system circuit was designed. Below is the circuit design for the skin conductance sensor.

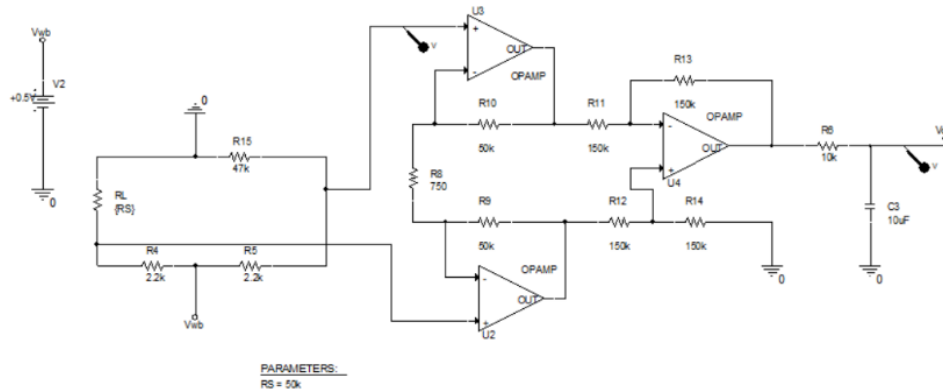


Figure 3.7 Circuit diagram of Skin Conductance Sensor Module.

3.2.1 Voltage regulator

As mentioned in Chapter 2, when measuring skin conductance, the skin resistance is measured using a Wheatstone bridge and the skin resistance may be measured connecting 0.5V to the bridge. This limitation was used to ensure human safety from any high voltage/current source. This was done by using a voltage regulator which regulates the voltage from 3.3V down to 0.5V at the output. For this purpose, a SC4216H IC with an SOIC-8-EDP package was used. This IC was specifically designed for a low input voltage (as low as $V_{in}=1.5V$) and very low programmable output voltage (as low as 0.5V). This IC was implemented in the Eagle schematic. Because of the unavailability of the SC4216H and a voltage regulator (output voltage as low as 0.5V) in through-hole mounting technology, the power supply to the Wheatstone bridge was implemented using a Tektronix PS2520G programmable power supply for demonstration of the circuit design.

3.2.2 Wheatstone Bridge and Instrumentation Amplifier

As mentioned above, the human skin conductance ranges from 0.1 μS to 10 μS at the wrist. Considering this conductance range a variable resistor (R_x) was selected from 100 -

1000kΩ and R1, R2 and R3 were calculated. Later, as the output of Wheatstone bridge was too low to be read by the microcontroller, the output was amplified and filtered using an instrumentation amplifier. The instrumentation amplifier used in this work was the AD623. The AD623 is a single/dual supply instrumentation amplifier with an excellent common mode rejection ratio. It amplifies the signal with common mode voltages as low as 150mV below ground. This instrumentation amplifier was selected because it drives low power (550uA maximum supply current) and has a gain range from 1V/V to 1000V/V, noise harmonics are rejected due to constant CMRR up to 200Hz, and is often used in the low-power medical instrumentation industry. For the breadboard prototype, this IC was ordered in a Plastic Dual In-line Package (PDIP) form. For the Eagle PCB schematic an SOIC package was ordered. The circuit operates as follows: The output of the Wheatstone bridge is fed into the instrumentation amplifier which is done to ensure effective communication with the microcontroller and also to maintain a linear relationship between the output voltage of the instrumentation amplifier and the skin conductance. The relationship between output voltage V_o and the R1, R2, R3 bridge values (equation 3.12) is as below:

$$V_o = 0.5 * G * \left(\frac{C1}{C1+C2} - \frac{C3}{C3+S} \right) \dots\dots\dots (3.12)$$

$$C1 = \frac{1}{R1}, C2 = \frac{1}{R2}, C3 = \frac{1}{R3}, S = \frac{1}{R_L} \dots\dots\dots (3.13)$$

Using equation 3.12, 3.13, considering R_L from 25 - 1000kΩ, R1, R2 and R3 are calculated as 2.2, 2.2 and 47kΩ, respectively.

3.2.3 Low-Pass Filter

The output of the instrumentation amplifier contains power line AC noise and other noise above 1.6Hz. In order to remove all the unwanted noise a first order low-pass filter with a cut-off frequency of 1.6Hz was designed. Using equation 3.14 and considering that the maximum recommended impedance for analog inputs on the PIC18LF45K50 is 10kΩ, R and C values were calculated to be 10kΩ and 10uF, respectively.

$$W_c = \frac{1}{2\pi RC} \dots\dots\dots (3.14)$$

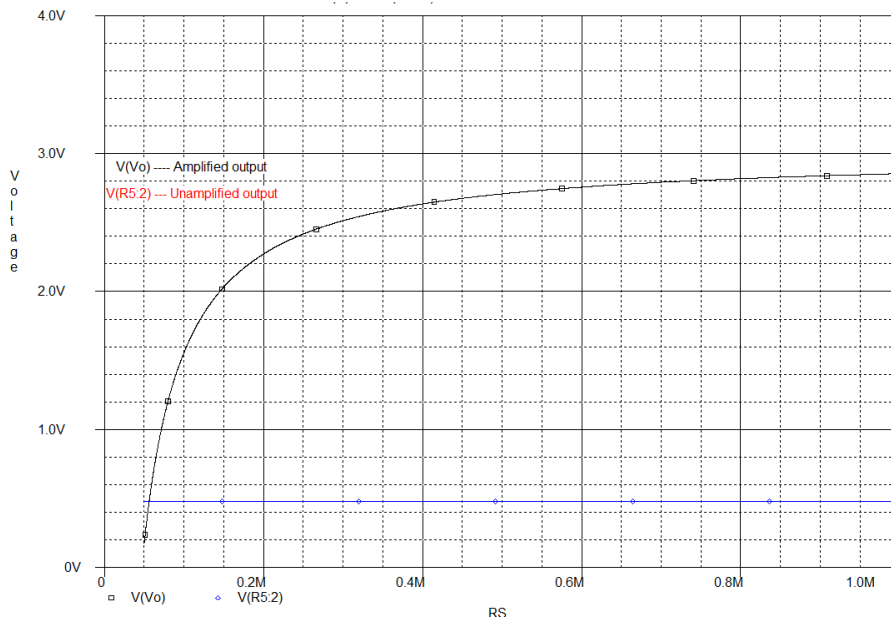


Figure 3.8 PSpice skin conductance simulation for unamplified signal and amplified signal after passing through instrumentation amplifier. Cut-off frequency for low pass filter 1.6Hz. Resistance swept from 50 to 1000kΩ.

Once the signal was filtered it was interfaced with the PIC18LF45k50 microcontroller and analog to digital conversion (ADC) was performed followed by digital signal conditioning and signal analysis.

3.2.4 Breadboard Prototype and Eagle PCB Schematic

The whole circuit was implemented on a breadboard using Silver/Silver Chloride electrodes. As stated above, Silver/Silver Chloride electrodes have better conductivity than other electrodes, hence they make better contact with the skin. For connecting Silver/Silver Chloride electrodes to the breadboard, snap connectors were used. The power supply to the Wheatstone bridge was implemented using a SC4216H voltage regulator but, in for the breadboard, it was provided directly using a Tektronix PS2520G programmable power supply. For implementing on the breadboard, the gain of the AD623 was set to 100V/V. A dual power supply powered the AD623 using the Tektronix PS2520G programmable power supply and the relationship between the output voltage and skin conductance was analyzed. Ag/AgCl electrodes in this sensor module were connected to the printed circuit board using male jumpers. These male jumpers are pinned out in the Eagle schematic, which can be found in appendix A.

3.3 Temperature Sensor System

The monitoring of human body temperature while hiking is a necessity. This is because many people die due to hypothermia, heat stroke, etc. That is often from physical conditions which are caused by a change in weather conditions leading to a drastic change in their body temperature. For measuring human body temperature, the MAX30205 human body temperature sensor IC was used. This sensor was selected as it measures human body temperature accurately meeting the clinical thermometry specification of ASTM1112. It also provides an overtemperature alarm/interrupt/shutdown output. This sensor requires a

2.7 - 3.3V supply voltage range, low 600uA supply current which meets the low power specification of the trauma detection personal locator beacon system. This IC is available in an 8-pin Thin Dual Flat No-Lead (TDFN) plastic package. When this device reads temperature, it converts the analog measurement into digital form using 16-bit temperature resolution and a sigma-delta analog to digital converter. As mentioned in Chapter 2, I2C communication of the sensor with the microcontroller allows access of temperature data to the system.

3.3.1 Circuit Diagram

The circuit diagram for this sensor module is simple. Since the device works on I2C communication, the serial data (SDA) and serial clock (SCL) of the slave IC (MAX30205) was connected with the serial data and serial clock bus of the master IC (microcontroller). This was done to establish I2C communication between master and slave. SDA, SCL and OS pins are connected with 4.7k Ω pull-up resistors to ensure sufficient filtering. V_{DD} of the IC needs to be connected to V_{DD} of the master IC.

3.3.2 Breadboard and Eagle Schematics

This IC is available in surface mount technology with a TDFN package, thus a breakout board for this IC was used for implementing the circuit on the breadboard. For the printed circuit board this IC was purchased and the equivalent circuit for this sensor system designed. The overtemperature shut-down output pin of this IC was connected with the interrupt pin of the microcontroller. The Eagle schematic of this sensor system is available in appendix A.

3.4 Accelerometer Sensor System

The most common reason for hiker, mountaineer, and skier death is falling, either into holes and pits or falling from a height making a blunt impact causing head injuries, bone fracture,s etc. For this reason, it is necessary to measure motion of a body to detect any freefall condition or a motionless body. For measuring motion of a human body, we are using a 3-axis digital accelerometer, which measures normal motion, freefall, and inactivity of the body. It also lets us know if any impact falls (i.e., sudden deceleration) have occurred. The accelerometer selected is model no. ADXL345 from Analog Devices. This device is an ultralow power, 3-axis digital accelerometer with a 13-bit resolution measurement capability up to ± 16 g's. This device generates a signal from all 3 axis directions and converts them into a digital signal using a high-resolution analog to digital converter. It communicates and transfers data to a master device using the I2C communication protocol thus providing data to the master. It senses the dynamic acceleration resulting from the presence of, or lack of, motion of the body or resulting from shock and senses static acceleration, such as gravity. It has a special function such as activity, inactivity, freefall, impact, single tap, and double tap which gives information about body motion and freefall motion, and detects when an impact has occurred.

3.4.1 Circuit Diagram

The circuit diagram of the accelerometer is very simple. Its serial data (SDA) port is connected with serial data (SDA) port of master IC (microcontroller) and the serial clock (SCL) port is connected with the serial clock (SCL) port of the master IC (microcontroller).

The SCL and SDA lines must be connected with pull up resistors for proper I2C operation.

A 10uF tantalum capacitor, 0.1uF ceramic decoupling capacitor, and a 90Ω ferrite bead ($|Z_L(100\text{ MHz})| = 90\ \Omega$) are connected to the supply pin to decouple the accelerometer from any unwanted noise. Vdd is also connected with a bypass capacitor to filter unwanted noise.

3.4.2 Breadboard and Eagle Schematic

This IC is available in both surface mount LGA packages. Due to the unavailability of this IC in through hole technology format, its breakout board was purchased and connected with the microcontroller to access the output. For the printed circuit board, the LGA package was purchased. The Eagle Schematic for the accelerometer system is also available in Appendix A.

3.5 Microcontroller

As mentioned previously, this system is an ultra-low power system design, hence the PIC18LF45K50, which is a low power, high performance microcontroller, was selected. It has a wide operating voltage range from 1.8V to 3.6V which also fits the overall system power supply specifications. The device was connected with decoupling capacitors (0.001uF and 0.1uF) at power supply pins (Vdd and Vss) to shunt any high frequency noise. A 20MHz external quartz oscillator was also connected and, in the Eagle schematic, male header pins are used to connect it with the UART to USB connector (i.e., the PICKIT 3). The microcontroller was connected with a few extra peripherals on the breadboard and the PCB such as an activation button for cancelling the automatic trigger, buzzer alarm and LED power supply.

3.6 Power Supply and Battery Charger

For the power supply module, the power supply to breadboard circuit was implemented using an Elegoo MB1002 3.3, 5V power supply connected to a 9V battery. For the printed circuit board, the power supply was powered with an Adafruit Lithium Ion Polymer 3.7V 500mAh battery. Apart from this a DC buck boost converter circuit was designed using the IC LT1304 to achieve a regulated 3.3V value. This IC operates at voltages ranging from 1.5V to 8V generating a regulated output of 3.3V and 5V. It can be used in small, low voltage, battery operated and medical instrumentation devices. For charging the Battery, a Type B Micro USB port was added to the Eagle schematic to provide power from the USB power sources and the BQ24155 IC was also connected to charge the Lithium Ion polymer battery. This IC was selected because of its charge voltage regulation accuracy ($\pm 0.5\%$), charge current regulation accuracy ($\pm 5\%$) and its chemical composition. This IC is available in a Quad Flat No-Lead (QFN) package format. The Eagle Schematic of the microcontroller with its basic connections, power supply, battery charger and other external peripherals can be found in Appendix A.

3.7 Summary

In this chapter, each sensor module was explained in depth along with its technical details. For the pulse oximeter, a bandpass filter was designed to separate the AC component of the output signal and a low pass filter was designed to separate the DC component. The SpO₂ and pulse rate calculation concepts were introduced and analyzed. For the skin conductance sensor, a Wheatstone bridge was designed to measure skin resistance and then the signal from the Wheatstone bridge amplified using an

instrumentation amplifier. For measuring body temperature, the MAX30205 breakout board was used and for measuring body motion (freefall and impact detection) the ADXL345 breakout board is used.

In the next chapter details of breadboard implementation will be presented along with the measured performance of each system component.

Chapter 4: Implementation

In this chapter implementation of all sensor systems along with the programming of individual sensor modules is presented. Initially skin conductance and pulse oximeter readings were recorded. For this, a subject is asked to participate in a physical activity to alter their heart rate. Prior to this the subject baseline (skin conductance and SpO₂) was recorded using the breadboard prototype. Later the subject did ten (10) squat rounds, followed by an increment of 5 squats in each round and, after each round, data was read, analyzed and displayed on a serial terminal using PuTTY software. Skin conductance and pulse oximeter module output were initially displayed on an oscilloscope. Apart from the breadboard prototype a C code was developed in MPLAB X IDE software and, using the ADC port of the PIC18LF45k50, the data was read and analyzed. For displaying the data on PuTTY (serial terminal) UART communication was used between the microcontroller and Desktop. Later this data was applied to equations mentioned in Chapter 3 and particular parameter values were found. Also, the graph between voltage obtained and skin conductance and SpO₂ and R was plotted using equation 3.1 in Excel. For the accelerometer and temperature sensors the data were collected and read by the microcontroller using I²C communication. The data from the microcontroller was then sent to the serial terminal (PuTTY) using the UART serial communication protocol. For the breadboard prototype,

breakout boards of these two sensors were purchased and interfaced with a PIC18LF45K50 microcontroller.

4.1 Implementation of Skin Conductance

For implementing the measurement of skin conductance, this circuit was assembled on a breadboard. A 0.5volt power supply was provided to the Wheatstone bridge using a Tektronix PS2520G programmable power supply. A dual supply was connected to the AD623 instrumentation amplifier using a Tektronix PS2520G programmable power supply. For measuring skin conductance AgCl electrodes were placed on the front side of the wrist and voltage between the two electrodes measured. The output of this sensor was read by the ADC of PIC18LF45K50 microcontroller. The output of the skin conductance module was fed into the AN0 analog pin of the microcontroller. The obtained digital output was then displayed on PuTTY using UART communication. As the data is of double data type, to make it readable in the PuTTY window this data was converted into its ASCII value and then sent using the UART protocol. For converting the data into ASCII, double data type data was converted into a string using the dtoa function(). Then this string was sent to the serial terminal (Putty) using UART communication at 9600 baud rate. The PIC18LF45k50 microcontroller was connected the breadboard with its basic connections and a PICKIT3 programmer used to program the chip.

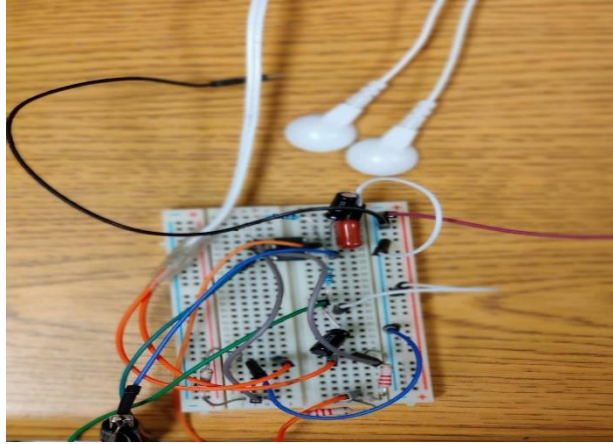


Figure 4.1 Breadboard prototype of Skin conductance module.

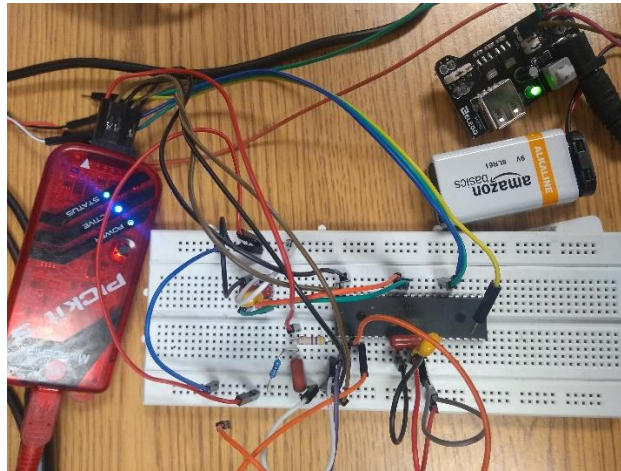


Figure 4.2 PIC18LF45k50 microcontroller breadboard and connection to the PICKIT3 programmer used to load the code into the chip.

4.1.1 Device Testing

For testing the skin conductance module, a few experiments were performed on a subject. Initially the subject's skin voltage (voltage between the AgCl electrodes) was displayed on PuTTY and served as the baseline value. Then the subject did ten (10) rounds of squats followed by an increment of ten (10) squats for another four (4) experiments. The voltage across the electrodes was measured and displayed on PuTTY. When measuring

voltage, the resistance between the electrodes was simultaneously measured using a digital multimeter. Once the voltage was measured across the electrodes, skin resistance was calculated using Equation 3.12. and a graph of skin resistance and voltage plotted using Excel. On performing this experiment, it was observed that, after increasing the amount of physical activity, the skin voltage dropped resulting in an logarithmic relationship between voltage versus resistance across the electrodes obtained. As the conductance is reciprocal of skin resistance, skin conductance tends to increase as expected.

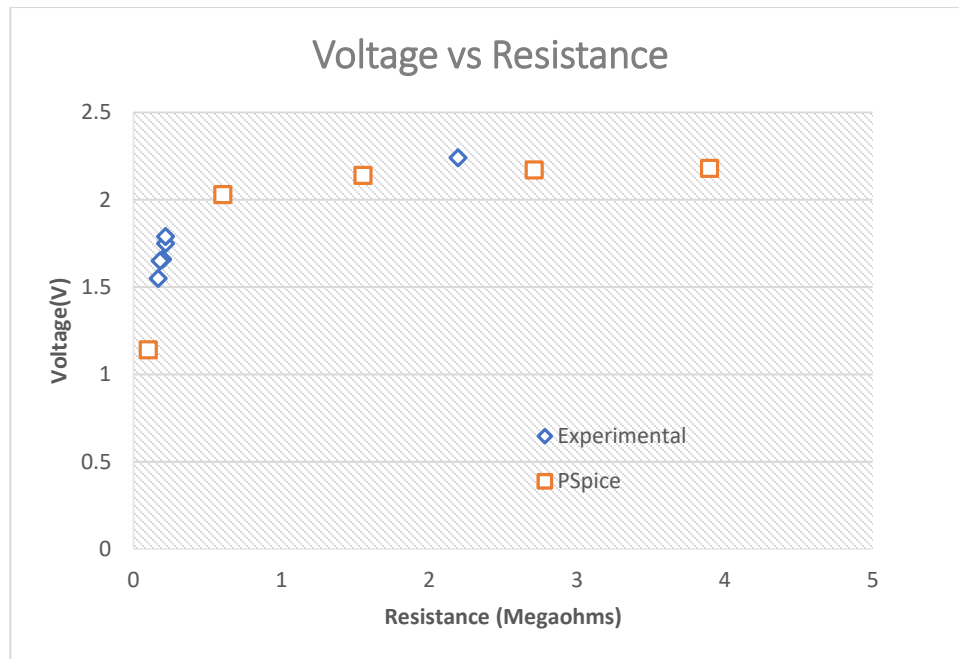


Figure 4.3 Overlay plot: Graph between skin voltage vs. resistance. Graph on comparing with PSpice simulation shows similar logarithmic response

Table 4.1: Readings Obtained During Testing

Readings duration	Voltage (V)	Calculated Resistance (k Ω)	Skin Conductance (uS)	Multimeter Resistance (k Ω)	Error(%)
Baseline	2.24	2197	0.45	2100	4.6
10 squats	1.75	218	4.5	216	0.9
20 squats	1.66	197	5.0	198	0.5
30 squats	1.65	181	5.5	178	1.6
40 squats	1.79	217	4.6	205	5.8
50 squats	1.55	167	5.9	155	7.7

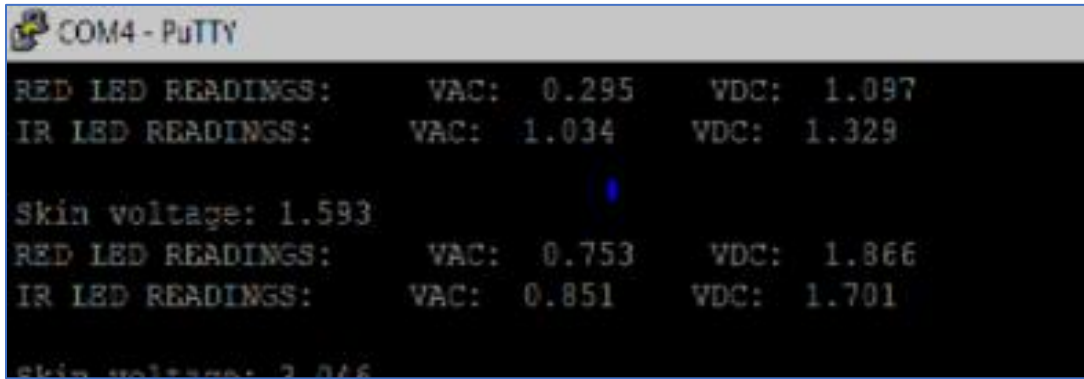


Figure 4.4 Skin voltage on PuTTY after fifth round of squats. Program in while loop giving readings after delay of 2 sec.

4.2 Implementation of Pulse Oximeter

For implementing the pulse oximeter, the bandpass filter with amplifier and low pass filter were assembled on a breadboard and test results displayed on an oscilloscope.

Later the AC component output from the bandpass filter and amplifier was fed into AN1 (ADC channel) of the PIC18LF45K50 microcontroller for reading the analog data and converted into digital form. The DC component value obtained from the low pass filter was fed into the AN2 channel of the PIC18LF45k50 microcontroller. For reading the signal, Red and IR LEDs were placed on one side of a concave structure module and a phototransistor placed on the other side. The concave structure was designed in Solidworks and 3D printed using FlashFinder software.

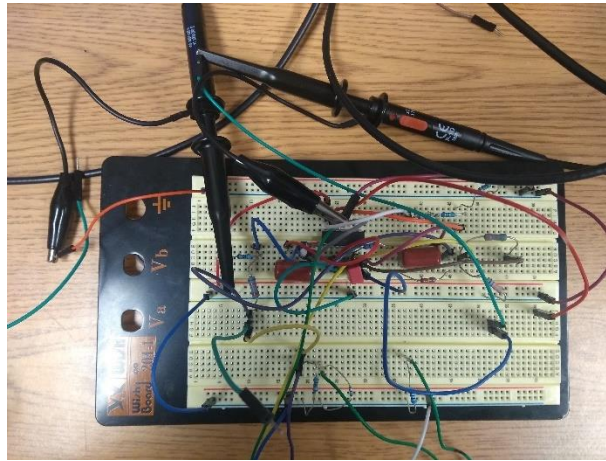


Figure 4.5 Breadboard prototype of Pulse oximeter module. Bandpass filter with amplifier and lowpass filter were implemented using a MCP604 IC which is a quad-opamp configuration. 3 separate channels were used for the filters and amplifier.

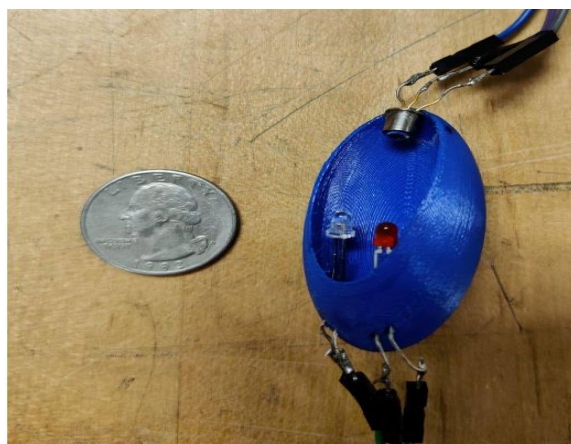


Figure 4.6 Photograph of the pulse oximeter 3D Printed Concave Sensor support. Phototransistor is on the left while the red (top) and IR (bottom) LEDs on the right.

To start the process, the red and IR LEDs were connected with PORTB of the microcontroller; Red being connected to RB0 and IR to RB1. Both the LEDs were turned on via the program; red being first, turned on for 6 seconds, and later IR for another 6 seconds. The emitter of the phototransistor was connected in series to a 10k Ω resistor which then connects to ground while the collector was supplied with the 3.3 V power supply. The 10k Ω resistor was selected to ensure that the circuit was more sensitive to light and the voltage generated by the phototransistor greater compared to lower resistance values. The output of the phototransistor was amplified using a non-inverting operational amplifier with a gain of 16V/V as the signal obtained from the human body is always low in magnitude ranging from 0.5mV to 5mV. Below are the signals obtained from the red and IR LEDs obtained on the oscilloscope.

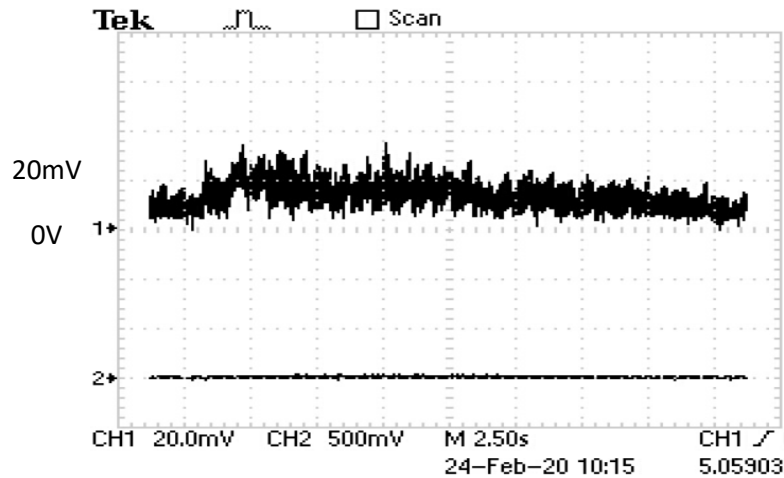


Figure 4.7 The pulsed signal obtained on channel 1 when the red LED is on. Marker on channel 1 shows 0V as the baseline and each square represents 20mV as its size.

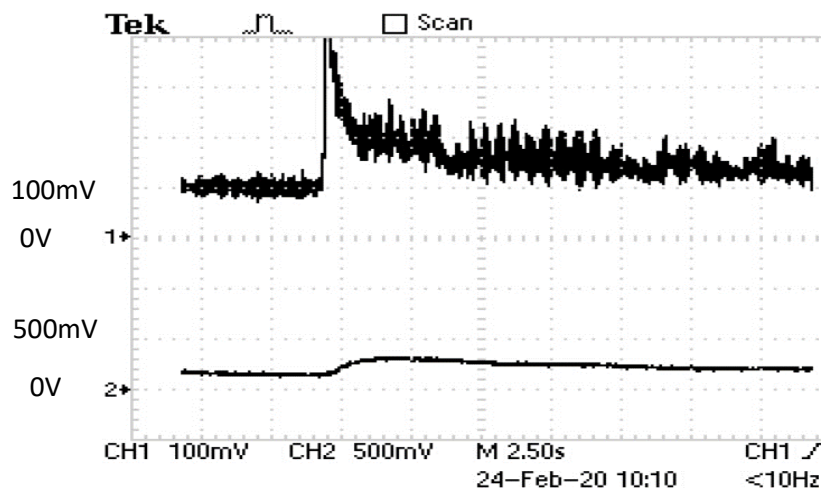


Figure 4.8 The pulse signal obtained from the IR LED on channel 1. Marker on channel 1 shows 0V as the baseline and each square represents 100mV as its size. Channel 2 is the DC output obtained from Sallen-Key low pass filter.

4.2.1 Pulse oximeter programming

As mentioned above, the output of the bandpass filter and amplifier chain was connected to the AN1 pin (AC component output) and output of low pass filter was connected to the AN2 pin (DC component output) of the microcontroller. AN1 provides

the signal AC component while AN2 the DC component. For taking pulse oximeter readings, the Pulsereading() function was created. This function contains the logic of obtaining AC and DC component values. Initially the red LED was turned on programmatically for 6 seconds; then AC and DC component values for both filters and amplifiers was read using the Pulsereadings() function. In this function, a ‘for loop’ was created which attempts to collect 60 sampled readings. For this, inside the ‘for loop’ pin AN1 (ADC channel1 analog pin) was activated first which reads the AC component value. Then pin AN2 (ADC channel 2 analog pin) was activated to collect DC component samples. The first 10 samples, read in 1 sec, were filtered to ensure filtration of ambient light and noise, later when $i > 10$ the loop then enters an ‘if’ statement which attempts to collect AC values for the voltage1 variable and DC values for the voltage2 variable. The loop mainly collects an average of 50 samples of AC and DC component readings separately for the VAC and VDC variables.

$$VAC = \frac{\sum_{n=1}^{n=50} voltage1_n}{n} \dots\dots\dots (4.1)$$

$$VDC = \frac{\sum_{n=1}^{n=50} voltage2_n}{n} \dots\dots\dots (4.2)$$

Once these values are calculated the ‘if’ statements are closed and the ‘for’ loop is closed. The data is sampled for 100 milliseconds, (AC and DC readings are obtained in 100 milliseconds) cumulating in 60 sampled values each for AC and DC components. Once the program comes out of the ‘for’ loop, VAC and VDC readings (float values) are

converted into a string using the `dtoa()` function and displayed on PuTTY using the `UART_Txstring()` function. Then the program comes out of the `Pulsereadings` function and the IR LED is turned on. Later the `Pulsereadings` function is again called which calculates the average of the AC and DC components for the IR LED and displays the output. Once these readings are displayed the program comes out of 'while' loop using a break statement. As mentioned above the program for the pulse oximeter is integrated with the program for the skin conductance module. Below is the screenshot of the Putty window when operated during squats.

```
COM4 - PuTTY
IR LED READINGS:      VAC:  0.952      VDC:  1.602
Skin voltage: 2.408
RED LED READINGS:      Skin voltage: 2.704
RED LED READINGS:      VAC:  1.179      VDC:  0.297
IR LED READINGS:      VAC:  0.461      VDC:  0.069
Skin voltage: 2.415
RED LED READINGS:      VAC:  0.479      VDC:  0.026
IR LED READINGS:      VAC:  1.096      VDC:  0.425
Skin voltage: 2.415
```

Figure 4.9 Screenshot of readings displayed on serial Terminal (Putty window) during 10 squats.

4.2.2 Device Testing and Calibration

For testing the device, 6 experiments were performed on a subject. The first experiment recorded the normal SpO2 level of the subject (i.e., baseline reading). The second experiment was from squats activity done by the subject. The response of the filters

were recorded on the oscilloscope and an average of the AC and DC components obtained programmatically.

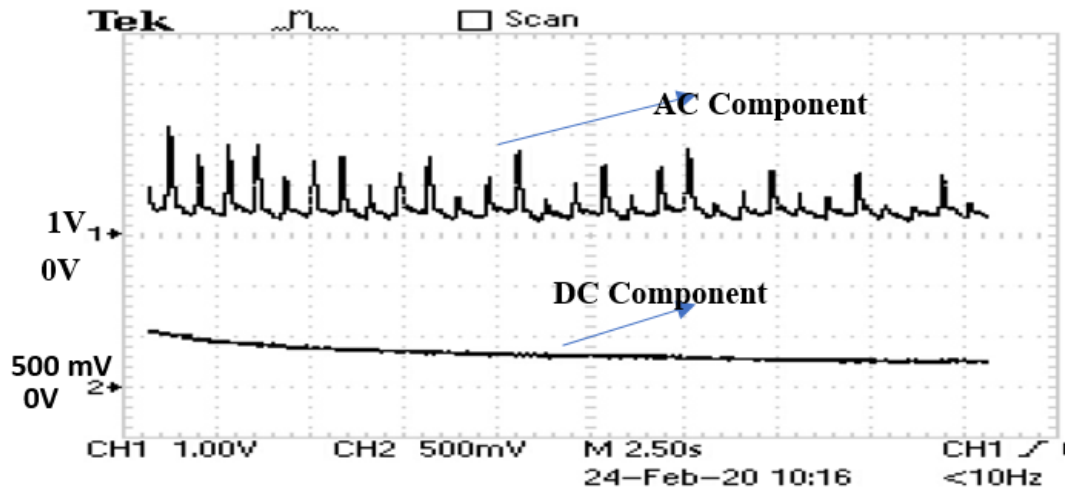


Figure 4.10 Filter outputs obtained for red LED on the oscilloscope. Channel 1 shows the bandpass filter and amplifier output with a gain of 100V/V while Channel 2 shows the output from the low pass filter, having a gain of 1.59V/V.

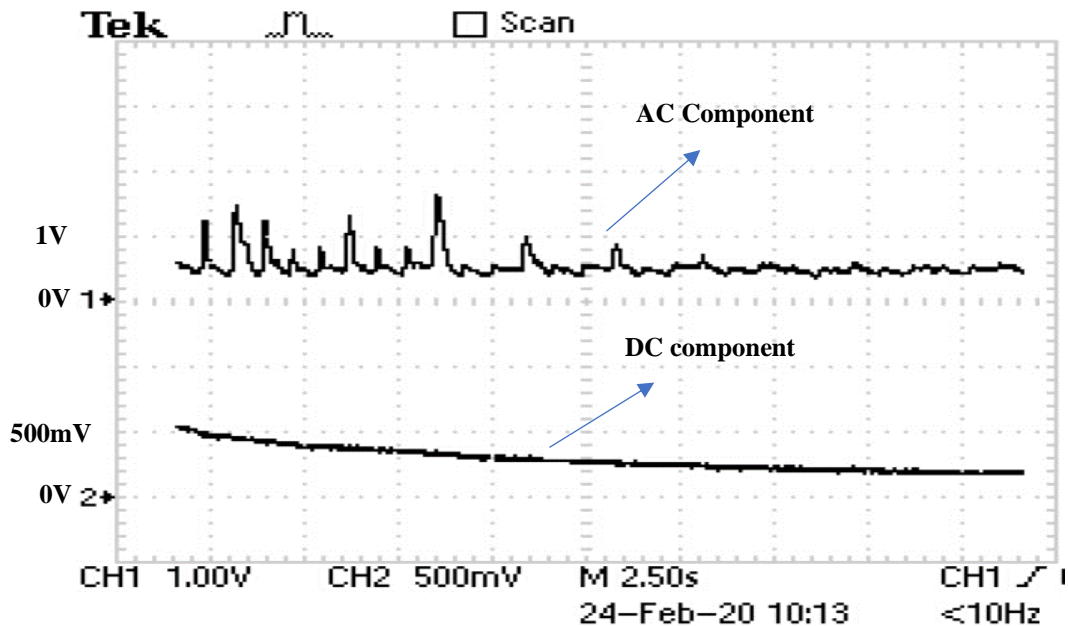


Figure 4.11 Filter outputs obtained from the IR LED on the oscilloscope. Channel 1 shows the bandpass filter and amplifier output with a gain of 100V/V while Channel 2 shows the output from the low pass filter, having a gain of 1.59V/V.

On obtaining the average AC and DC readings for the two LEDs, these values were applied in the R formula as mentioned in Section 3.1.1 of Chapter 3 using excel and R value was calculated for each experiment. For calibrating the sensor, the SpO2 was measured using a calibrated SpO2 sensor from the Yuwell YX302 Fingertip Pulse Oximeter simulator. The data from the simulator was taken simultaneously in each experiment and these SpO2 and R values used in equation 3.1 The α , β , γ values were then calculated for the experiment. The calculated α , β and γ values are 99.6603, -3.6427, 1.7190 giving SpO2 equation as

$$\text{SpO}_2 = 99.63 + 3.6427R - 1.7190R^2$$

Table 4.2 Measured values for the SpO2 experiment

Physical Activity	VAC(red) (V)	VDC(red) (V)	VAC(IR) (V)	VDC(IR) (V)	Ratio (V/V)	SpO2 from calibrated device (%)
Baseline	0.446	0.164	0.398	0.474	0.308	98.70
20 squats	1.102	1.221	0.610	0.710	0.952	97.75
20 squats	0.414	3.129	0.273	3.183	0.643	99.00
15 squats	0.546	1.733	0.482	1.944	0.784	97.86
15 squats	0.576	0.227	0.323	0.733	0.173	98.85
10 squats	0.480	0.110	0.462	0.108	0.980	98.83



Figure 4.12 Graph between calibrated SpO2 readings vs. calculated R. As SpO2 value tends to increase, ratio R tends to decrease. The two data points where SpO2 is 99 and 98.83 shows certain discrepancies because of the mechanical design of the concave sensor making difficult to collect accurate readings.

4.3 Implementation of Accelerometer Sensor

For implementing the accelerometer sensor, the ADXL345 breakout board was purchased and interfaced with a PIC18LF45K50 microcontroller. As this sensor is a 3-axis digital accelerometer, the data from this sensor was read by the microcontroller using I2C communication. On reading the data from this sensor and transfer to the microcontroller, the data was sent to the laptop using the UART communication protocol at a baud rate of 9600 and displayed using PuTTY software. Being a triple-axis-digital accelerometer, this sensor displays the data in X, Y, Z axes. For reading the data from the ADXL345, the data for loop is created which collects 16 sampled values from xhi, xlo, yhi, ylo, zhi, zlo

registers using the I2Cread() function. Later xhi, xlo 8-bit registers are combined to get 16-bit data in the Xaccumulate variable. A similar method is performed for the yhi, ylo and zhi, zlo registers to obtain 16-bit data using Yaccumulate and Zaccumulate variables. Once the for loop is closed the average of 16 readings stored in the Xaccumulate, Yaccumulate and Zaccumulate variables are obtained using the div_t structure and the quotient of these average readings displayed as the output. For displaying the output xavg.quot, which is an integer value, this data is converted into a string to obtain its ASCII value using the itoa() function and later transmitted and displayed through the UART_Txstring() function.

```
COM4 - PuTTY
X: -54      Y: -21      Z: 229
X: -54      Y: -21      Z: 229
X: -54      Y: -24      Z: 230
X: -54      Y: -22      Z: 230
X: -54      Y: -22      Z: 230
X: -54      Y: -22      Z: 229
X: -55      Y: -22      Z: 229
X: -55      Y: -22      Z: 230
X: -55      Y: -23      Z: 229
X: -54      Y: -22      Z: 229
```

Figure 4.13 Accelerometer readings obtained when in still-motion. X,Y,Z axis values tend to be almost similar. Units of X,Y,Z is LSB.


```

COM4 - PuTTY
X:  -41      Y:  -93      Z:  260
X:   66      Y: -216      Z:  -81
X:   35      Y: -185      Z: -189
X:   -6      Y: -221      Z: -126
X: -110      Y: -241      Z:   35
X:  -36      Y: -203      Z: -100
X:  157      Y: -228      Z:   10
X:  230      Y: -173      Z:  -13

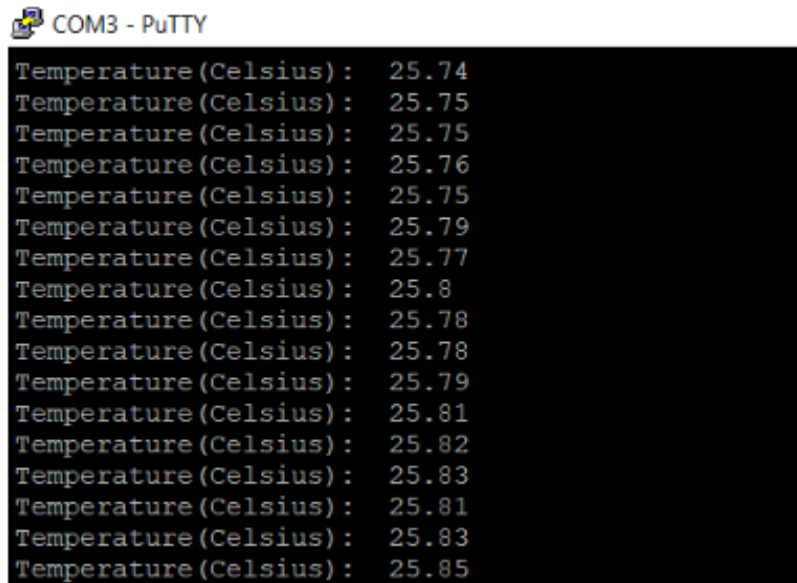
```

Figure 4.14 Accelerometer readings obtained when the accelerometer is in motion. As shown, X, Y, Z axis values vary due to the presence of motion. Units of X,Y,Z is LSB.

4.4 Implementation of Temperature Sensor

For implementing the temperature sensor, the MAX30205 breakout board was interfaced with the PIC18LF45K50 microcontroller. As this sensor has 32 different slave address bytes, the slave address is dependent on the A0, A1, A2 address pins of the MAX30205. The 0x90 slave address with R/W bit is selected amongst 32 slave address bytes which is done by connecting A0, A1 and A2 with GND. The temperature data is obtained by configuring the temperature register and reading this data using the I2Cread() function. As the temperature register is 2 bytes and I2C communication is in 7 bit addressing mode, the data is read from the temperature register in two variables temp_upper (which collects 8 bits of data starting with MSB) and temp_lower (which collects lower 8 bits of data ending with LSB). Later temp_upper and temp_lower, each containing 8 bits of data, are combined to obtain 16-bit data in the temp variable. Then this temp variable is converted to the Celsius temperature range to obtain the temperature reading in Celsius. As the Celsius value is float data type, the Celsius data is converted to

a string using the `dtoa()` function and transmitted and displayed on the laptop using PuTTY via the `UART_Txstring()` function. For performing all these steps the `max_read_temp()` function is called.



```
COM3 - PuTTY
Temperature (Celsius) : 25.74
Temperature (Celsius) : 25.75
Temperature (Celsius) : 25.75
Temperature (Celsius) : 25.76
Temperature (Celsius) : 25.75
Temperature (Celsius) : 25.79
Temperature (Celsius) : 25.77
Temperature (Celsius) : 25.8
Temperature (Celsius) : 25.78
Temperature (Celsius) : 25.78
Temperature (Celsius) : 25.79
Temperature (Celsius) : 25.81
Temperature (Celsius) : 25.82
Temperature (Celsius) : 25.83
Temperature (Celsius) : 25.81
Temperature (Celsius) : 25.83
Temperature (Celsius) : 25.85
```

Figure 4.15 Temperature sensor (MAX30205) output readings obtained in degree Celsius when fingertip was touched on IC surface.

4.5 Algorithm for Alarm Detection

Once the data is collected from the respective sensors, this data is compared with their minimum and maximum threshold values. If the readings are outside of threshold value, the alarm sequence is generated as described in Chapter X. In the PLB alarm sequence, when the first alarm is activated, it sounds locally for 2 minutes. If the user does not deactivate it, a second louder alarm activate for eight minutes to alert someone in the vicinity of the user. If deactivated before the time expires, automatic activation of the PLB is cancelled. However, if this does not occur then the PLB 406 MHz radio beacon is

activated. Figure 4.16 shows the PLB activation algorithm flow diagram which shows the sequence of events leading to automatic activation:

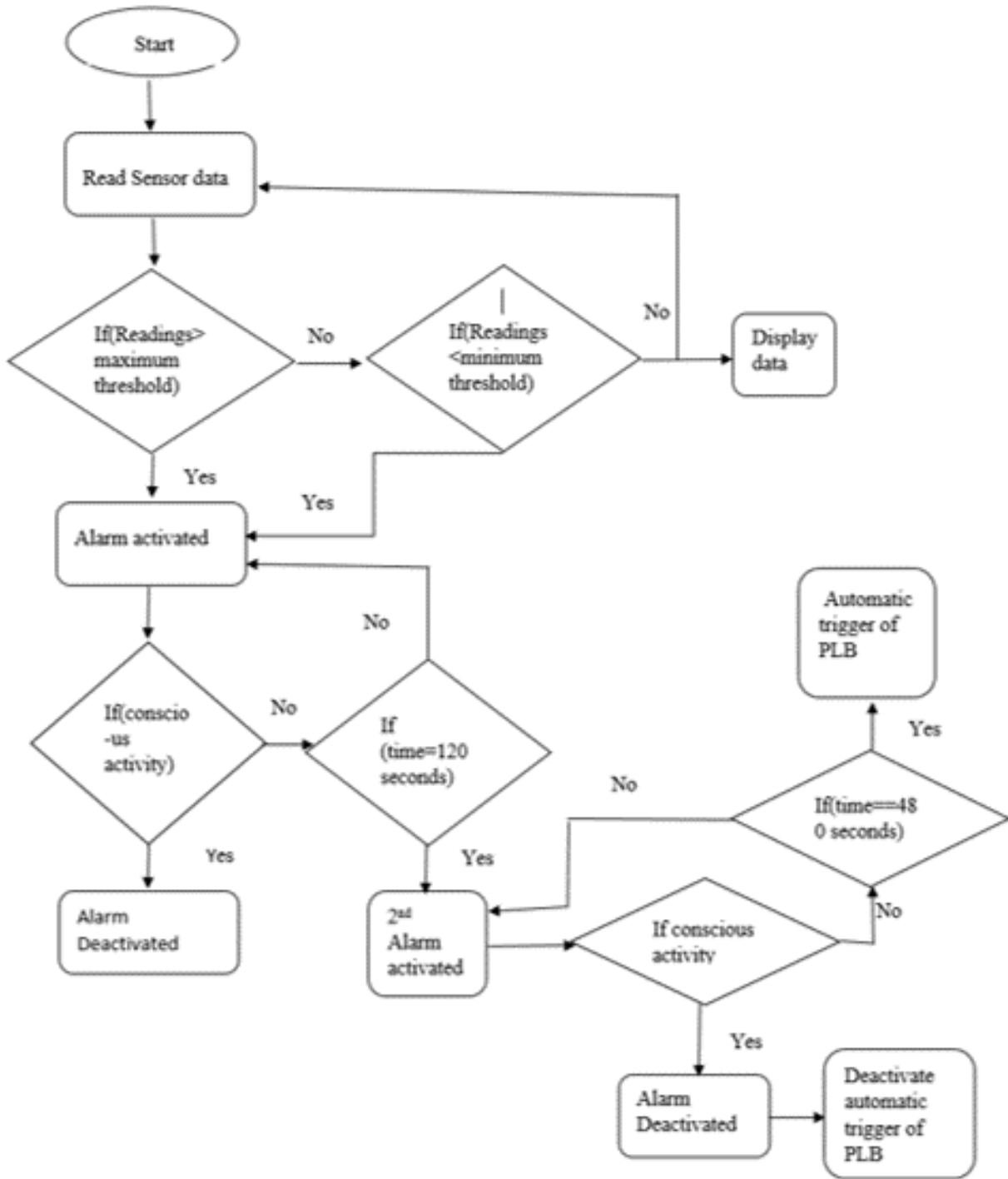


Figure 4.16 Algorithm for Alarm detection, automatic triggering of PLB for Skin conductance, pulse oximeter and temperature sensor.

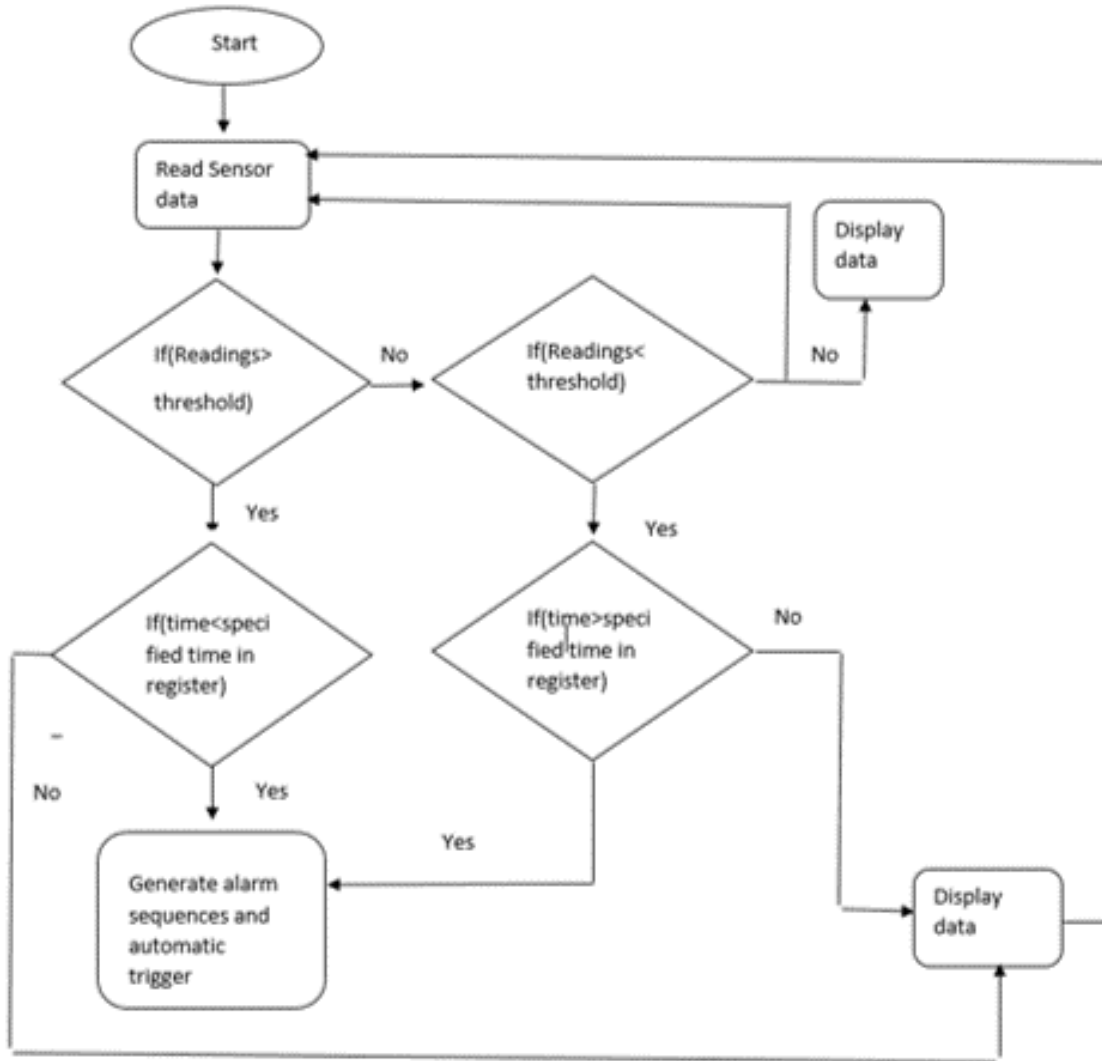


Figure 4.17 Algorithm for accelerometer sensor.

As mentioned in Figure 4.16 the maximum and minimum threshold values play an important role in activating the PLB. For the pulse oximeter if the user encounters an SpO₂ level below 90 a hypoxia condition is present. Depending upon the severity of the hypoxia condition of the user, the short alarm sequence will be generated. For heartrate, the maximum threshold is 100 BPM and minimum threshold is 60 BPM. For the temperature sensor, the maximum and minimum temperature values are written in THYST, TOF registers of MAX30205 and later the Comparator Mode is activated using configuration

register which allows MAX30205 to generate alarm if value goes above or below the threshold values. In Figure 4.17 is the Algorithm for the accelerometer sensor which is a bit different as single tap detection, double tap detection, activity, inactivity and freefall detection depends on the value stored in THRESH_TAP, THRESH_ACT, THRESH_INACT registers, while time is stored in the DUR register, TIME_INACT register and TIME_FF register. If the reading obtained is greater than the value stored in THRESH_TAP and occurs for less time than the time specified in the DUR register, it indicates single and double tap detection. In double tap detection, the second tap starts after the time specified in the Latent register and within the time specified in the window register. Activity detection is experienced when acceleration is greater than the value stored in THRESH_ACT. Inactivity detection is experienced when acceleration is less than the value stored in THRESH_INACT and occurs for more time than is specified in TIME_INACT. Similarly, freefall detection is experienced when the acceleration is less than the value stored in THRESH_FF and occurs for more time than the time specified in TIME_FF. The values stored in these registers are user dependent. According to the readings obtained from the accelerometer sensor, and comparison with the values stored in the respective registers, the decision for alarm generation and further processes takes place.

4.6 PLB Transmitter

Once the PLB system detects user trauma/injury, the emergency 406MHz radio beacon is activated to start the search and rescue operation. The PLB transmitter consists of the RF circuitry as well as a UHF antenna (UHF band 300 MHz to 3 GHz). Since the antenna size depends on the RF wavelength, a collapsible antenna can be designed into the

wristband which deploys automatically when needed. The RF wavelength at 406 MHz is ~ 0.74 m, which is quite large. The antenna can be designed using flexible antenna design found on devices McMurdo FastFind Max or the OceanSignal RescueME PLB, this antenna could be wrapped around the wrist wearable PLB and a mechanical actuator can be used to release the antenna when triggered. A basic block diagram of a 406 MHz radio transmitter is shown below:

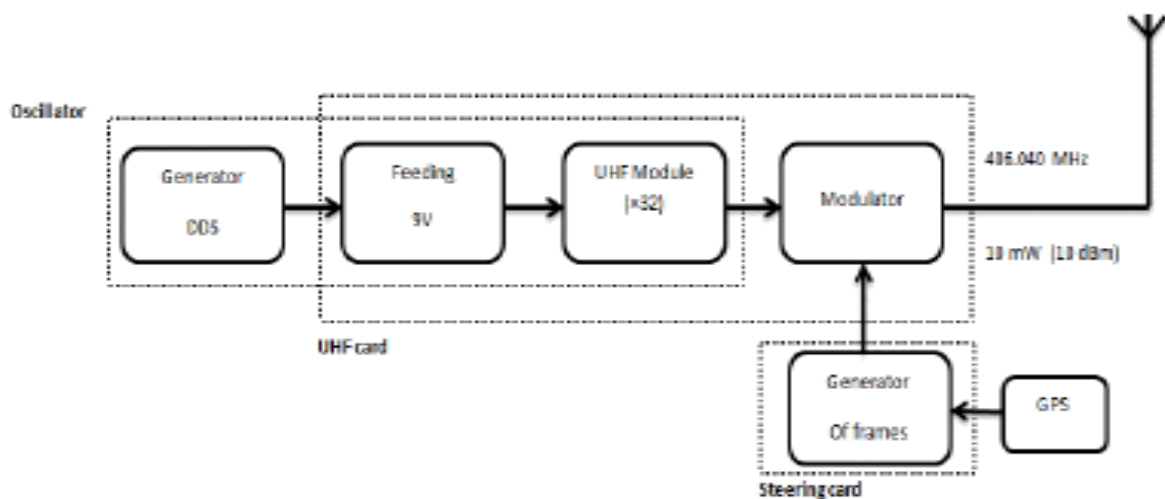


Figure 4.18 Functional block diagram of a 406 MHz beacon showing the basic components: 406 MHz oscillator, UHF card (containing power amplifier and RF modulator) and GPS signal path feeding a UHF antenna [23].

Chapter 5: Summary and Future Work

5.1 Summary

In this thesis, a personal locator beacon for trauma detection and automatic signaling to the search and rescue network has been developed. Due to the nature of this work the thesis work focused mostly on four physiological parameters: Skin conductance, Pulse oximeter, Body Motion (Freefall and Impact Detection) and Body Temperature. This was used to design low power trauma detection personal locator beacon that will feed a 406 MHz emergency distress beacon. This design differs from most systems available in the commercial marketplace which require the user to manually activate the beacon. Obviously, if the user is debilitated, and therefore unable to activate the beacon, an automatic means to both sense severe trauma and automatically activate the beacon is needed.

The physiological sensor sub-systems were reduced to practice as follows: PSpice simulation of Skin conductance module, followed by Pulse oximeter module was done to design the required electronics. Later the circuits were implemented on breadboards followed by ADC and UART programming. For human body motion detection, the ADXL345 accelerometer breakout board was used and for detecting human body temperature the MAX30205 temperature sensor breakout board was used. Data from the

system's respective sensors were successfully collected and displayed on a PuTTY window at a baud rate of 9600. The algorithm for the collecting all sensor data and initiating an alarm sequence, followed by automatic triggering of a 406 MHz radio beacon, was developed and presented.

5.2 Future Work

While the core components of the personal locator beacon have been designed and reduced to practice, there remain a few details that must be completed prior to commercialization of this system. First the breadboards for each module/sub-module need to be integrated onto a system PCB in order to reduce cost, size and weight. This is easily accomplished using surface mode technology (SMT). Naturally during this process debugging will need to be performed and, likely, more than one design/build out cycle will be required.

The second area of future work pertains to the modules/sub-modules that have yet to be implemented. The first is the alarm module, which had not been implemented due to a lack of availability of the parts initially selected. The second is the wireless broadcasting system, mainly the 406 MHz UHF transmitter and antenna. This portion of the system should be straightforward to design and implement as it is a high-volume produced commercial system. Finally, the custom-designed wearable package needs to be designed and prototyped. Figure 5.1 shows an example of the 406 MHz wireless system:

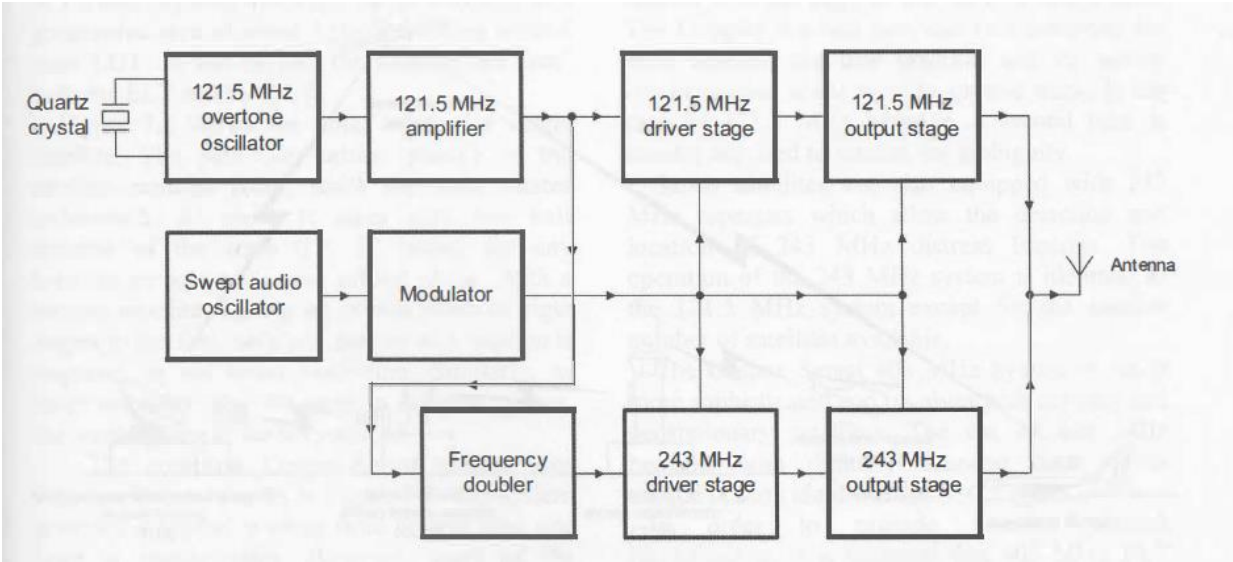


Figure 5.1 A simple Block diagram of Type-W ELT 406 MHz transmitter. [24].

While none of these remaining system/sub-system action items are difficult to realize, a lack of time in their implementation has resulted in them being left for a follow-on activity. Once completed a compact, cost-effective, user friendly system will result based on the base work presented in this thesis.

References

- [1] “Facts + Statistics: Marine Accidents | III”, *Iii.org*, 2019. [Online]. Available: <https://www.iii.org/fact-statistic/facts-statistics-marine-accidents>. [Accessed: 30- Sep- 2019].
- [2] T. Heggie and M. Amundson, “Dead Men Walking: Search and Rescue in US National Parks”, *Wilderness & Environmental Medicine*, vol. 20, no. 3, pp. 244-249, 2009. Available: 10.1580/08-weme-or-299r.1.
- [3] “Accident Statistics”, *Icao.int*, 2019. [Online]. Available: <https://www.icao.int/safety/iStars/Pages/Accident-Statistics.aspx>. [Accessed: 30- Sep- 2019].
- [4] M. Hooper, *Embedded System Design of Low-Power Wearable Bioelectronic Devices*, M.S. Thesis, Electrical Engineering, University of South Florida, 2018.
- [5] “Causes of Backpacking and Hiking Deaths”, *Whimsical Hikers*, 2019.
- [6] T. Heggie and T. Heggie, “DEAD MEN HIKING: CASE STUDIES FROM THE AMERICAN WILDERNESS”, *Medicina Sportiva*, vol. 16, no. 3, pp. 118-121, 2012. Available: 10.5604/17342260.1011392.
- [7] M. Faulhaber, M. Flatz, H. Gatterer, W. Schobersberger and M. Burtscher, “Prevalence of Cardiovascular Diseases Among Alpine Skiers and Hikers in the Austrian Alps”, *High Altitude Medicine & Biology*, vol. 8, no. 3, pp. 245-252, 2007. Available: 10.1089/ham.2007.1005.
- [8] “How Do PLBs Work – PLB Review – Personal Locator Beacon Reviews”, *PLB Review – Personal Locator Beacon Reviews*, 2019. [Online]. Available: <https://www.plbreview.com/personal-locator-beacons/how-do-plbs-work/>. [Accessed: 30-Sep- 2019].
- [9] AOPA, *HOW IT WORKS: PERSONAL LOCATOR BEACON*. 2019.

- [10] "What's Killing America's Hikers?", *SkyAboveUs*, 2017. .
- [11] D. Lykken and P. Venables, "DIRECT MEASUREMENT OF SKIN CONDUCTANCE: A PROPOSAL FOR STANDARDIZATION", *Psychophysiology*, vol. 8, no. 5, pp. 656-672, 1971. Available: 10.1111/j.1469-8986.1971.tb00501.x.
- [12] L. Majer, V. Stopjakova and E. Vavrinsky, "WIRELESS MEASUREMENT SYSTEM FOR NON-INVASIVE BIOMEDICAL MONITORING OF PSYCHO-PHYSIOLOGICAL PROCESSES", *Journal of ELECTRICAL ENGINEERING*, vol. 60, pp. 57-68, 2020. Available: http://iris.elf.stuba.sk/JEEEC/data/pdf/2_109-01.pdf. [Accessed 19 March 2020].
- [13] D. Hubert, P. Brunswick, J. Calvet, D. Dusser and I. Fajac, "Abnormal electrochemical skin conductance in cystic fibrosis", *Journal of Cystic Fibrosis*, vol. 10, no. 1, pp. 15-20, 2011. Available: 10.1016/j.jcf.2010.09.002.
- [14] *Media.mit.edu*, 2020. [Online]. Available: <https://www.media.mit.edu/galvactivator/faq.html>. [Accessed: 18- Mar- 2020].
- [15] "Biosensor Array Galvanic Skin Response - Pumping Station: One Wiki", *Wiki.pumpingstationone.org*, 2020. [Online]. Available: https://wiki.pumpingstationone.org/Biosensor_Array_Galvanic_Skin_Response. [Accessed: 18- Mar- 2020].
- [16] I. Sarussi, "DEVICE FOR USE WITH REFLECTIVPULSE OXMETRY", US20070038050A1, 2007.
- [17] H. Santha, N. Stuban and G. Harsanyi, "Design Considerations of Small Size Reflective Type Pulse Oximeter Heads in Special Applications", in *Electronics Systemintegration Technology Conference*, Dresden, 2020, pp. 404-408.
- [18] G. Di, X. Tang and W. Liu, "A Reflectance Pulse Oximeter Design Using the MSP430OF149", in *International Conference on Complex Medical Engineering*, Beijing, 2020, pp. 1081-1084.
- [19] G. Pang and C. Ma, "A Neo-Reflective Wrist Pulse Oximeter", *IEEE Access*, vol. 2, pp. 1562-1567, 2014. Available: 10.1109/access.2014.2382179.
- [20] N. Watthanawisuth, T. Lomas, A. Wisitsoraat and A. Tuantranont, "Wireless Wearable Pulse Oximeter for Health Monitoring using ZigBee Wireless Sensor Network", in *ECTI-CON2010: The 2010 ECTI International Confernce on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, Chiang Mai, 2020.

- [21] R. AMBA, "VARIATION OF ELECTRODERMAL ACTIVITY ACROSS HUMAN FINGERS - TOWARDS A NOVEL DIAGNOSTIC TECHNIQUE", Master of Technology, Indian Institute of Technology Madras, 2020.
- [22] *Apsycho*server.psych.arizona.edu, 2020. [Online]. Available: <http://apsychoserver.psych.arizona.edu/JJBAREprints/PSYC501A/Readings/Chapter%208.pdf>. [Accessed: 18- Mar- 2020].
- [23] B. Srihan, j. Yonnet and M. Benslama, "Design and realization an ELT beacon and decoders of frames 406 MHz", in *2017 International conference on Engineering and MIS*, Munastir, 2020.
- [24] "Avionics Navigation System", *Emergency Locator Transmitter*, 2020. .
- [25] GeeksforGeeks. 2020. *Implement Your Own Itoa() - Geeksforgeeks*. [online] Available at: <<https://www.geeksforgeeks.org/implement-itoa/>> [Accessed 18 March 2020].
- [26] C. string, S. Pal and d. kitten, "Convert a float to a string", *Stack Overflow*, 2020. [Online]. Available: <https://stackoverflow.com/questions/2302969/convert-a-float-to-a-string>. [Accessed: 18- Mar- 2020].
- [27] E. Forums and C. Repository, "ADXL345 Accelerometer and 18F4550 in Hi-Tech C", *Electronics Forum (Circuits, Projects and Microcontrollers)*, 2020. [Online]. Available: <https://www.electro-tech-online.com/threads/adxl345-accelerometer-and-18f4550-in-hi-tech-c.140017/>. [Accessed: 18- Mar- 2020].

Appendix A: Eagle Schematics

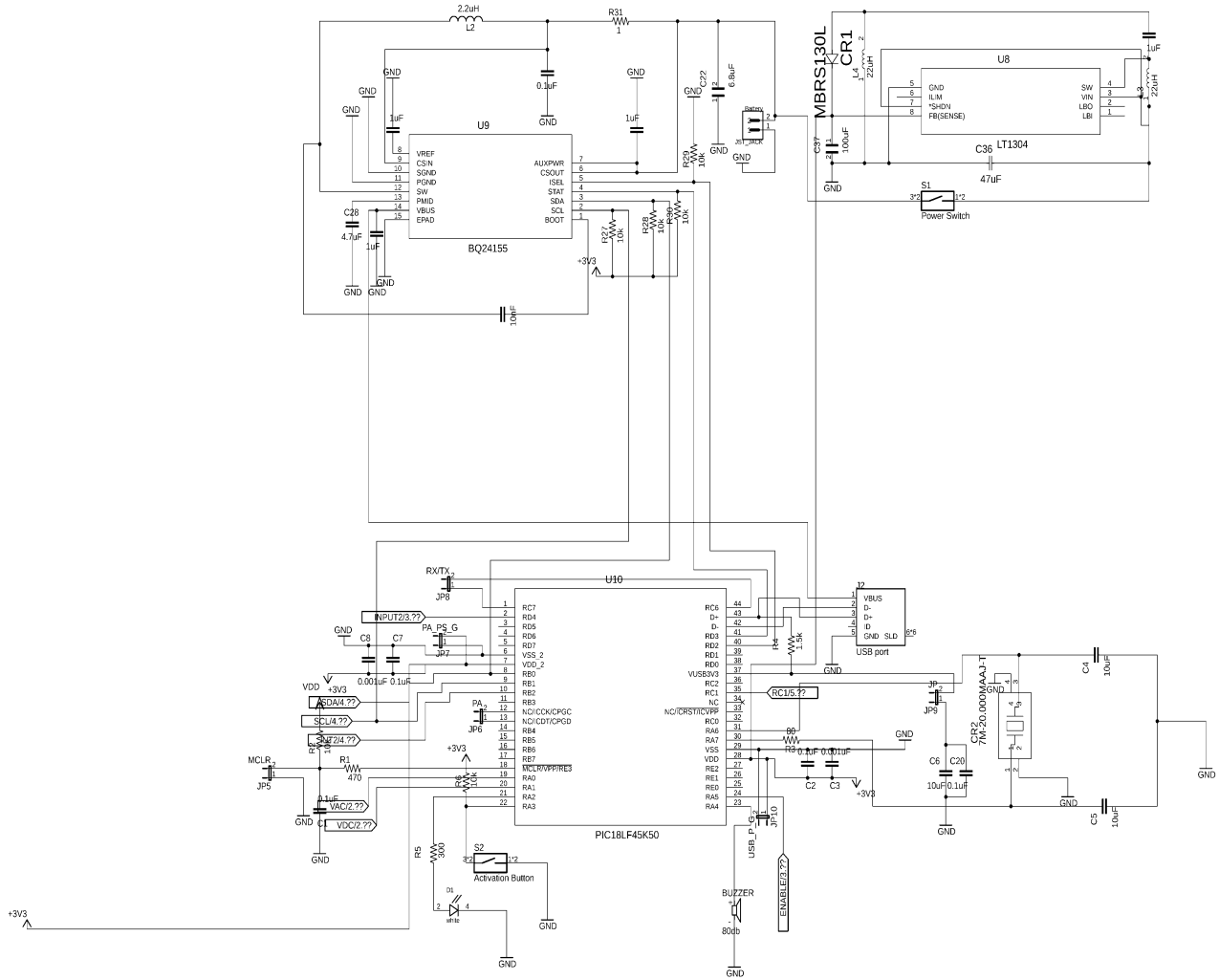


Figure A.1 Eagle Schematic- Microcontroller Basic connections with power supply and battery charger module.

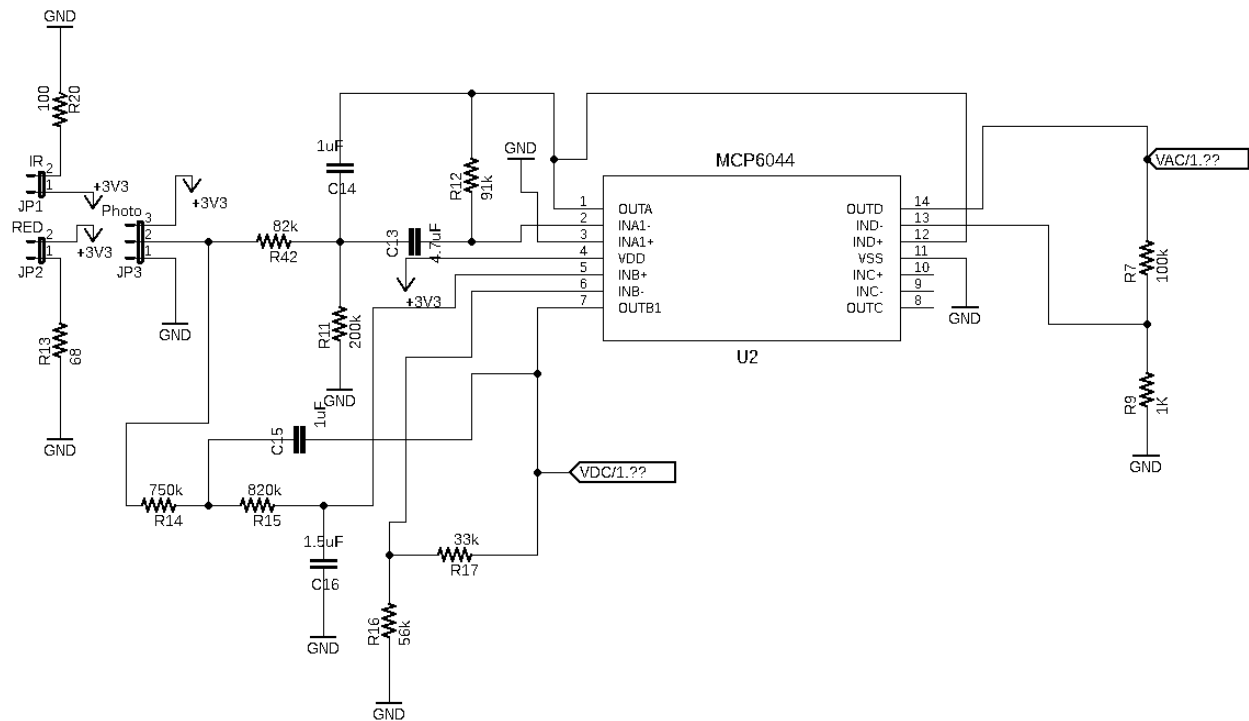


Figure A.2 Eagle Schematic- Pulse Oximeter sensor system

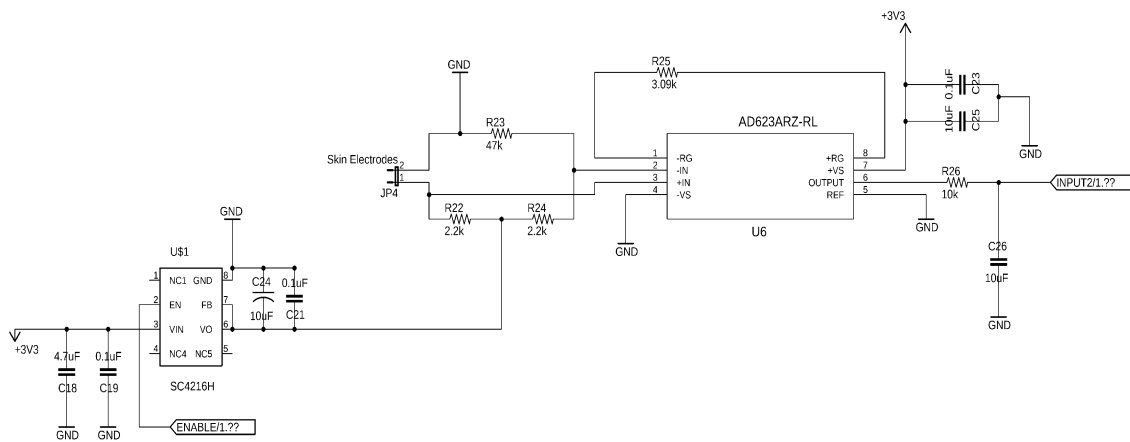


Figure A.3 Eagle Schematic- Skin Conductance sensor system

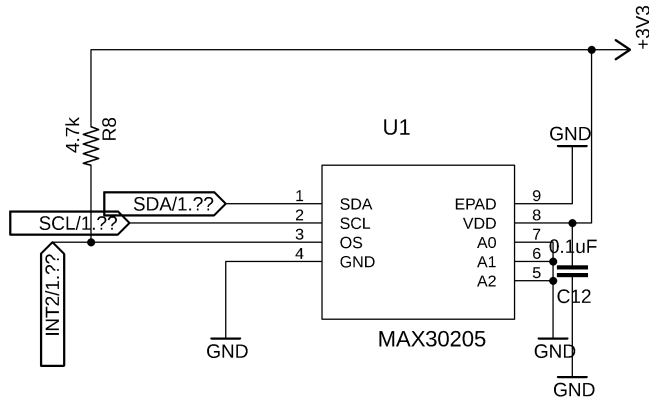


Figure A.4 Eagle Schematic- Temperature sensor system

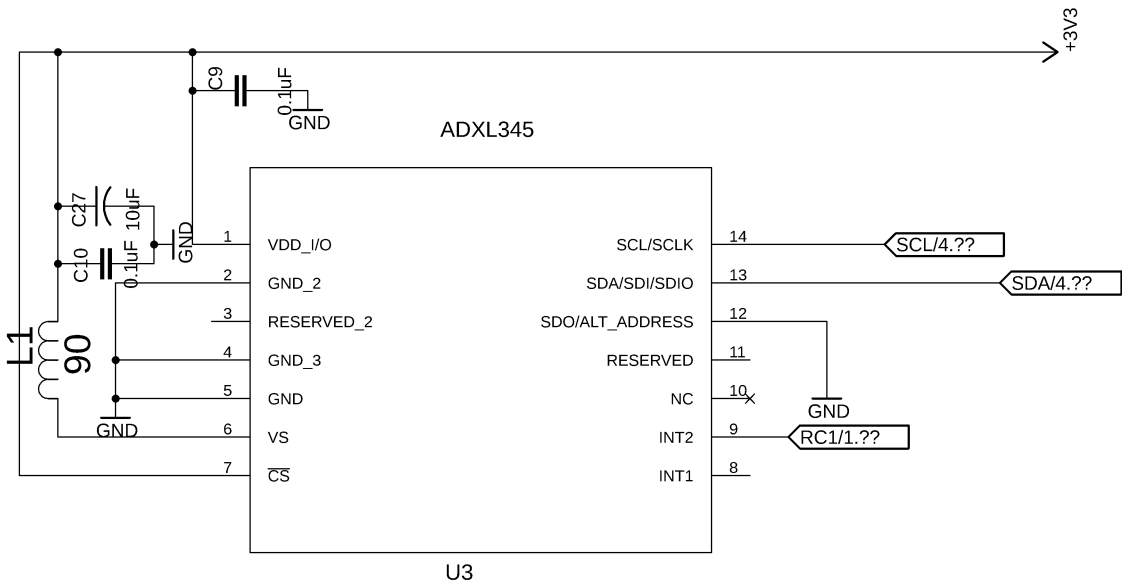


Figure A.5 Eagle Schematic- Accelerometer sensor system

Appendix B: Programming

B.1 Programming of Skin Conductance and Pulse Oximeter Module [25], [26].

```
#include <xc.h>
#include "ADC.h"
#include <pic18lf45k50.h>
#include <stdbool.h>
#include <math.h>
#include <string.h>
#define _XTAL_FREQ 8000000

void adc_init();
unsigned int adc_read();
void UART_Init(void);
void UART_Txstring(char data[60]);
char *itoa(long int num, char*str, int base);
void reverse(char str[], int length);
char *dtoa(double n, char *s);
void Pulsereadings();
int digi, digi1, digi2;
char digi_Buffer[10], Voltage_Buffer[10], VAC_Buffer[20], VDC_Buffer[20];
double voltage, voltage1, voltage2, VAC, VDC;
static double PRECISION = 0.01;

#define Vref 3.375
#define swap(x,y) (x^=y, y^=x, x^=y)
bool MODE;

void main() {

    OSCCON=0X62;
    TRISA=0x06;
    ANSELA=0x06;
    TRISC=0x82; // check RC1
    TRISB=0x00;
    ANSELC=0;
```



```

    adc_init();
    UART_Init();
while(1)
{
    ADCON0=0x01;
    digi= adc_read();
    voltage = digi*((float)(Vref)/(float)1024);
    dtoa(voltage, Voltage_Buffer);
    UART_Txstring("Skin voltage: ");
    UART_Txstring(Voltage_Buffer);
    UART_Txstring(" ");
    UART_Txstring("\n\r");
    __delay_ms(1000);
    __delay_us(3.4);//2TAD delay for next conversion*/
    PORTB=0x01;
    UART_Txstring("RED LED READINGS: ");
    Pulsereadings();
    PORTB=0x02;
    UART_Txstring("IR LED READINGS: ");
    Pulsereadings();
    __delay_us(3.4);
    PORTB=0x00;
    UART_Txstring("\n\r");
    __delay_ms(2000);
}
}
void adc_init()
{
    ADCON0bits.ADON=1; //ADON=1
    ADCON1=0x00; //VDD, VSS
    ADCON2=0x9F;//RIGHT JUSTIFIED, 6TAD, FRC (FRC TAD=1.7us,
TACQQ=7.45us, 6TAD as in datasheet.)
    ADRESH=0;
    ADRESL=0;
}
unsigned int adc_read()
{
    int digital;
    __delay_ms(2);//2ms for charge holding capacitor to get data
    ADCON0bits.GO_DONE=1;
    while(ADCON0bits.GO_nDONE==1);
    digital=((ADRESH<<8)+ADRESL);
    return(digital);
}

```

```

}
void UART_Init(void)
{
    SPBRG=12; /*baud rate=9600, SPBRG = (F_CPU /((64*9600))-1*/
    TXSTA=0x20; /*Transmit Enable(TX) enable*/
    RCSTA=0x90;
    TXREG=0; /*Receive Enable(RX) enable and serial port enable */
    RCREG=0;
}
void UART_Txstring(char tx_data[60]) // function to pass string
{
    unsigned int i;
    for(i=0;i<60;i++)
    {
        if(tx_data[i]=='\0')
        {
            tx_data[i]='\0';
            break;
        }
    }
    while(PIR1bits.TXIF ==0);
    TXREG=tx_data[i];
}
}
char *itoa(long int num, char*str, int base)
{
    unsigned int i=0;
    bool isNegative = false; //Handle 0 explicitly, otherwise empty string is printed for 0
    if(num==0)
    {
        str[i++]='0';
        str[i]='\0';
        return str;
    }
    //in standard itoa negative numbers are handled only with base 10. Otherwise numbers
    are considered unsigned.
    if (num<0&&base==10)
    {
        isNegative = true;
        num = -num;
    }
    //Process individual digits
    while(num !=0)
    {

```

```

    int rem = num % base;
    str[i++] = (rem > 9)? (rem-10) + 'a': rem + '0';
    num = num/base;
}
// If number is negative, append '-'
if (isNegative)
    str[i++] = '-';
// Append string terminator
str[i] = '\0';
// Reverse the string
reverse(str,i);
}
void reverse(char str[], int length)
{
    int start =0;
    int end=length-1;
    while(start<end)
    {
        swap(*(str+start), *(str+end));
        start++;
        end--;
    }
}
void Pulsereadings()
{
    unsigned int i;
    VAC=0; VDC=0;

    unsigned int j=1;
    for(i=1;i<=60;i++)
    {
        ADCON0 = 0x05; // AC component readings // AN1 channel selected
        digi1=adc_read();
        voltage1 = digi1*((float)Vref/(float)1024); //AC voltage reading
        __delay_us(3.4);
        ADCON0=0x09; //DC component readings in channel AN2
        digi2=adc_read();
        voltage2 = digi2*((float)Vref/(float)1024);

        if(i>10)
        {
            VAC=(VAC+voltage1)/j;
            VDC=(VDC+voltage2)/j;

```

```

    j++;
}
__delay_ms(96);
}
dtoa(VAC, VAC_Buffer);    //shift after for loop
dtoa(VDC, VDC_Buffer);
UART_Txstring("VAC: ");
UART_Txstring(VAC_Buffer);
UART_Txstring("  ");
UART_Txstring("VDC: ");
UART_Txstring(VDC_Buffer);
UART_Txstring("\n\r");
}
/**
 * Double to ASCII
 */
char *dtoa(double n, char *s) {
    // handle special cases
    if (isnan(n)) {
        strcpy(s, "nan");
    } else if (isinf(n)) {
        strcpy(s, "inf");
    } else if (n == 0.0) {
        strcpy(s, "0");
    } else {
        int digit, m, m1;
        char *c = s;
        int neg = (n < 0);
        if (neg)
            n = -n;
        // calculate magnitude
        m = log10(n);
        int useExp = (m >= 14 || (neg && m >= 9) || m <= -9);
        if (neg)
            *(c++) = '-';
        // set up for scientific notation
        if (useExp) {
            if (m < 0)
                m -= 1.0;
            n = n / pow(10.0, m);
            m1 = m;
            m = 0;
        }
    }
}

```

```

if (m < 1.0) {
    m = 0;
}
// convert the number
while (n > PRECISION || m >= 0) {
    double weight = pow(10.0, m);
    if (weight > 0 && !isinf(weight)) {
        digit = floor(n / weight);
        n -= (digit * weight);
        *(c++) = '0' + digit;
    }
    if (m == 0 && n > 0)
        *(c++) = '.';
    m--;
}
if (useExp) {          // convert the exponent
    int i, j;
    *(c++) = 'e';
    if (m1 > 0) {
        *(c++) = '+';
    } else {
        *(c++) = '-';
        m1 = -m1;
    }
    m = 0;
    while (m1 > 0) {
        *(c++) = '0' + m1 % 10;
        m1 /= 10;
        m++;
    }
    c -= m;
    for (i = 0, j = m-1; i < j; i++, j--) {
        // swap without temporary
        c[i] ^= c[j];
        c[j] ^= c[i];
        c[i] ^= c[j];
    }
    c += m;
}
*(c) = '\0';
}
return s;}

```

B2. Programming of Accelerometer and Temperature Sensor [25], [26], [27].

```
#include "Configuration.h"
#define _XTAL_FREQ 8000000
#include <pic18lf45k50.h>
#include<stdint.h>
#include <stdbool.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>

#define Vref 3.25
#define swap(x,y) (x^=y, y^=x, x^=y)
bool MODE;

#define SlaveAddress 0x90
#define Temperature 0x00
#define Configuration 0x01
#define THYST 0x02
#define Tos 0x03
#define MAX_SHUTDOWN 0b00000001
#define MAX_COMPARATOR 0b00000010
#define MAX_OSPOLARITY 0b00000100
#define MAX_FAULTQUEUE 0b00001000
#define MAX_DATAFORMAT 0b00000000
#define TimeoutEnable 0b00000000
#define MAX_ONESHOT 0b10000000

char IDDV =0;
#define BW_RATE 44 //0x2C
#define POWER_CTL 45 //0x2D
#define DATA_FORMAT 49 //0x31
#define DATA_X0 50 //0x32
#define DATA_X1 51 //0x33
#define DATA_Y0 52 //0x34
#define DATA_Y1 53 //0x35
#define DATA_Z0 54 //0x36
#define DATA_Z1 55 //0x37
#define FIFO_CTL 56 //0x38 ADXL1345 1010011 0x53 Device
#define CHIP_Write 0xA6 // adxl345 address for writing 10100110 0xA6
Write
#define CHIP_Read 0xA7 // and reading
```

```

char X_Buffer[10], Y_Buffer[20], Z_Buffer[20];
static double PRECISION = 0.01;

void UART_Init(void);
void UART_Txstring(char data[60]);
char *itoa(long int num, char*str, int base);
void reverse(char str[], int length);
char *dtoa(double n, char *s);
void max_write_config(int8_t config);
void max_read_temp();

void I2CInit(void);
void I2CStart();
void I2CStop();
void I2CRestart();
void I2CAck();
void I2CNak();
void I2CWait();
void I2CSend(unsigned char dat);
unsigned char I2CRead(void);
char E_Write(int addr, unsigned char ch);
unsigned char E_Read(int addr);
char Temp_Buffer[10];
char ErrFlags;

void ini_adxl345()
{
E_Write(FIFO_CTL,0x9f);
__delay_ms(10);
E_Write(DATA_FORMAT,0x09);
__delay_ms(10);
E_Write(BW_RATE,0x0d);
__delay_ms(10);
E_Write(POWER_CTL,0x08); // activate
}

char ErrFlags; // Read error. (I don't use it, but you might!)
void main()
{
OSCCON=0X62;
TRISC=0x82;
ANSEL=0; // text variable
int index = 0; // text variable

```

```

unsigned char ch;
UART_Init();
I2CInit();
ini_adx1345();
signed long x,y,z;
signed long xhi,xlo,yhi,ylo,zhi,zlo;
signed long Xaccumulate, Yaccumulate, Zaccumulate;
signed long Xaverage, Yaverage, Zaverage;
int i;
while(1)
{
    Xaccumulate = Yaccumulate = Zaccumulate = 0;

    for (i=0; i<16; i++) { // Read sequentially 16 times then get an average.

```

```

        ErrFlags = 0; // Clear error
        I2CStart();
        I2CSend(CHIP_Write);
        I2CSend(DATA_X0); // DATA_X0 is the first of 6 bytes
        I2CRestart();
        I2CSend(CHIP_Read); //
        xlo = I2CRead(); // read character
        I2CAck();
        xhi = I2CRead(); // read character
        I2CAck();
        ylo = I2CRead(); // read character
        I2CAck();
        yhi = I2CRead(); // read character
        I2CAck();
        zlo = I2CRead(); // read character
        I2CAck();
        zhi = I2CRead(); // read character
        I2CNak();
        I2CStop();

```

```

        Xaccumulate += ((xhi<<8) | xlo); //Xaccumulate = Xaccumulate + (xhi*256 + xlo)
        Yaccumulate += ((yhi<<8) | ylo);
        Zaccumulate += ((zhi<<8) | zlo);

```

```

    } // for loop
    div_t xavg ; // div_t is a structure explained above

```



```

div_t yavg ;
div_t zavg ;

xavg = div(Xaccumulate, 16);
yavg = div(Yaccumulate, 16);
zavg = div(Zaccumulate, 16);

itoa(xavg.quot, X_Buffer, 10);
itoa(yavg.quot, Y_Buffer, 10);
itoa(zavg.quot, Z_Buffer, 10);
UART_Txstring("X: ");
UART_Txstring(X_Buffer);
UART_Txstring(" ");
UART_Txstring("Y: ");
UART_Txstring(Y_Buffer);
UART_Txstring(" ");
UART_Txstring("Z: ");
UART_Txstring(Z_Buffer);
UART_Txstring(" \n\r");
__delay_ms(1000);
max_write_config(MAX_ONESHOT);
__delay_ms(50);
max_read_temp();
__delay_ms(500);
}
}

void UART_Init(void)
{
    SPBRG=12; /*baud rate=9600, SPBRG = (F_CPU /((64*9600))-1*/
    TXSTA=0x20; /*Transmit Enable(TX) enable*/
    RCSTA=0x90;
    TXREG=0; /*Receive Enable(RX) enable and serial port enable */
    RCREG=0;
}

void UART_Txstring(char tx_data[60]) // function to pass string
{
    unsigned int i;
    for(i=0;i<60;i++)
    {
        if(tx_data[i]=='\0')
        {
            tx_data[i]=='\0';
        }
    }
}

```

```

        break;
    }
    while(PIR1bits.TXIF ==0);
    TXREG=tx_data[i];
}
}
char *itoa(long int num, char*str, int base)
{
    unsigned int i=0;
    bool isNegative = false; //Handle 0 explicitly, otherwise empty string is printed for 0
    if(num==0)
    {
        str[i++]='0';
        str[i]='\0';
        return str;
    }
    //in standard itoa negative numbers are handled only with base 10. Otherwise numbers
    are considered unsigned.
    if (num<0&&base==10)
    {
        isNegative = true;
        num = -num;
    }
    //Process individual digits
    while(num !=0)
    {
        int rem = num % base;
        str[i++] = (rem > 9)? (rem-10) + 'a': rem + '0';
        num = num/base;
    }
    // If number is negative, append '-'
    if (isNegative)
        str[i++] = '-';
    // Append string terminator
    str[i] = '\0';
    // Reverse the string
    reverse(str,i);
}
void reverse(char str[], int length)
{
    int start =0;
    int end=length-1;
    while(start<end)

```

```

    {
        swap(*(str+start), *(str+end));
        start++;
        end--;
    }
}
char *dtoa(double n, char *s) {
    // handle special cases
    if (isnan(n)) {
        strcpy(s, "nan");
    } else if (isinf(n)) {
        strcpy(s, "inf");
    } else if (n == 0.0) {
        strcpy(s, "0");
    } else {
        int digit, m, m1;
        char *c = s;
        int neg = (n < 0);
        if (neg)
            n = -n;
        // calculate magnitude
        m = log10(n);
        int useExp = (m >= 14 || (neg && m >= 9) || m <= -9);
        if (neg)
            *(c++) = '-';
        // set up for scientific notation
        if (useExp) {
            if (m < 0)
                m -= 1.0;
            n = n / pow(10.0, m);
            m1 = m;
            m = 0;
        }
        if (m < 1.0) {
            m = 0;
        }
        // convert the number
        while (n > PRECISION || m >= 0) {
            double weight = pow(10.0, m);
            if (weight > 0 && !isinf(weight)) {
                digit = floor(n / weight);
                n -= (digit * weight);
                *(c++) = '0' + digit;
            }
        }
    }
}

```

```

    }
    if (m == 0 && n > 0)
        *(c++) = '.';
    m--;
}
if (useExp) {
    // convert the exponent
    int i, j;
    *(c++) = 'e';
    if (m1 > 0) {
        *(c++) = '+';
    } else {
        *(c++) = '-';
        m1 = -m1;
    }
    m = 0;
    while (m1 > 0) {
        *(c++) = '0' + m1 % 10;
        m1 /= 10;
        m++;
    }
    c -= m;
    for (i = 0, j = m-1; i < j; i++, j--) {
        // swap without temporary
        c[i] ^= c[j];
        c[j] ^= c[i];
        c[i] ^= c[j];
    }
    c += m;
}
*(c) = '\0';
}
return s;
}
void I2CInit(void)
{
    TRISBbits.TRISB0 = 1; /* SDA and SCL as input pin */
    TRISBbits.TRISB1 = 1; /* these pins can be configured either i/p or o/p */
    SSPSTAT |= 0x80; /* Slew rate disabled */
    SSPCON1 = 0x28; // Enable SDA and SCL, I2C Master mode, clock = FOSC/(4 *
(SSPADD + 1))
    /* SSPEN = 1, I2C Master mode, clock = FOSC/(4 * (SSPADD + 1)) */
    SSPCON2 = 0x00; // Reset MSSP Control Register
}

```

```

SSPADD = 19; //20000000 / 4= 5000000, 5000000/ 100000= 50 50-1=49
PIR1bits.SSPIF=0; // Clear MSSP Interrupt Flag
PIR2bits.BCLIF=0;
/* *** For different Frequencies ***
FOSC  FCY  SSPADD Value FCLOCK
40 MHz 10 MHz  63h    100 kHz
32 MHz  8 MHz  4Fh    100 kHz
20 MHz  5 MHz  31h, 49d  100 kHz
16 MHz  4 MHz  27h    100 kHz
8 MHz   2 MHz  13h, 19d 100 kHz
4 MHz   1 MHz  09h    100 kHz
*/
}
void I2CStart()
{
    SEN = 1;    /* Start condition enabled */
    while(SEN); /* automatically cleared by hardware */
                /* wait for start condition to finish */
}

void I2CStop()
{
    PEN = 1;    /* Stop condition enabled */
    while(PEN); /* Wait for stop condition to finish */
                /* PEN automatically cleared by hardware */
}

void I2CRestart()
{
    RSEN = 1;   /* Repeated start enabled */
    while(RSEN); /* wait for condition to finish */
}

/*
Function: I2CAck
Return:
Arguments:
Description: Generates acknowledge for a transfer
*/
void I2CAck()
{
    ACKDT = 0; /* Acknowledge data bit, 0 = ACK */
    ACKEN = 1; /* Ack data enabled */
    while(ACKEN); /* wait for ack data to send on bus */
}

```

```

}

/*
Function: I2CNck
Return:
Arguments:
Description: Generates Not-acknowledge for a transfer
*/
void I2CNak()
{
    ACKDT = 1;    /* Acknowledge data bit, 1 = NAK */
    ACKEN = 1;    /* Ack data enabled */
    while(ACKEN); /* wait for ack data to send on bus */
}

/*
Function: I2CWait
Return:
Arguments:
Description: wait for transfer to finish
*/
void I2CWait()
{
    while ((SSPCON2 & 0x1F) || ( SSPSTAT & 0x04 ));
    /* wait for any pending transfer */
}

/*
Function: I2CSend
Return:
Arguments: dat - 8-bit data to be sent on bus
            data can be either address/data byte
Description: Send 8-bit data on I2C bus
*/
void I2CSend(unsigned char dat)
{
    SSPBUF = dat; /* Move data to SSPBUF */
    while(BF);    /* wait till complete data is sent from buffer */
    I2CWait();    /* wait for any pending transfer */
}

/*
Function: I2CRead

```

Return: 8-bit data read from I2C bus

Arguments:

Description: read 8-bit data from I2C bus

*/

```
unsigned char I2CRead(void)
```

```
{
```

```
    unsigned char temp;
```

```
    /* Reception works if transfer is initiated in read mode */
```

```
    RCEN = 1;    /* Enable data reception */
```

```
    while(!BF); /* wait for buffer full */
```

```
    temp = SSPBUF; /* Read serial buffer and store in temp register */
```

```
    I2CWait();    /* wait to check any pending transfer */
```

```
    return temp; /* Return the read data from bus */
```

```
}
```

```
void max_write_config(int8_t config)
```

```
{
```

```
    I2CStart(); // start the i2c stream
```

```
    I2CSend(SlaveAddress); // call the MAX address
```

```
    I2CSend(Configuration); // Config register address
```

```
    I2CSend(config); // write in configuration bits
```

```
    I2CStop(); // stop the i2c stream
```

```
}
```

```
void max_read_temp()
```

```
{
```

```
    uint16_t temp;
```

```
    uint8_t temp_lower;
```

```
    uint8_t temp_upper;
```

```
    float celsius = 0;
```

```
    I2CStart();
```

```
    I2CSend(SlaveAddress);
```

```
    I2CSend(Temperature);
```

```
    I2CRestart();
```

```
    I2CSend(SlaveAddress + 1);
```

```
    temp_upper = (uint8_t) I2CRead(); // read the two bytes in the temperature register
```

```
    I2CAck();
```

```
    temp_lower = (uint8_t) I2CRead();
```

```
    I2CNak();
```

```
    I2CStop(); // stop the i2c stream
```

```
    __delay_ms(50);
```

```
    max_write_config(MAX_SHUTDOWN);
```

```
    __delay_ms(50);
```

```
    temp = ((temp_upper << 8) | temp_lower);
```

```

    celsius = (float) temp * 0.00390625;
    dtoa(celsius, Temp_Buffer);
    UART_Txstring("Temperature(Celsius): ");
    UART_Txstring(Temp_Buffer);
    UART_Txstring("\n\r");
}
// This is random writing. Write to a specified address
char E_Write(int addr, unsigned char ch)
{
    /* Send Start condition */
    I2CStart();
    /* Send ADXL1345 slave address with write operation */
    I2CSend(CHIP_Write);
    /* Send subaddress, we are writing to this location */
    I2CSend(addr);
    /* send I2C data one by one */
    I2CSend(ch);
    /* Send a stop condition - as transfer finishes */
    I2CStop();
    return 1;                // All went well
}

// The function takes an address and Returns a character
// This is random reading. Read from a specified address
unsigned char E_Read(int addr)
{
    unsigned char byte;
    unsigned char ch;
    ErrFlags = 0;           // Clear error
    /* Send Start condition */
    I2CStart();
    /* Send ADXL1345 slave address with write operation */
    I2CSend(CHIP_Write);
    /* this address is actually where we are going to read from */
    I2CSend(addr);
    /* Send a repeated start, after a dummy write to start reading */
    I2CRestart();
    /* send slave address with read bit set */
    I2CSend(CHIP_Read);


    ch = I2CRead();        // read character
    I2CNak();
    I2CStop();
}

```



```
return ch;  
}
```

Appendix C: Copyright Permission



IEEE
Requesting permission to reuse content from an IEEE publication

A Neo-Reflective Wrist Pulse Oximeter
Author: Grantham Pang
Publication: IEEE Access
Publisher: IEEE
Date: 2014
Copyright © 2014, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

- 1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
- 2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
- 3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

- 1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
- 2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
- 3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

[BACK](#) [CLOSE](#)

Figure C1 Copyright permission for Figure 2.3