

March 2018

Embedded System Design of Low-Power Wearable Bioelectronic Devices

Matthew S. Hopper

University of South Florida, hopperms91@gmail.com

Follow this and additional works at: <http://scholarcommons.usf.edu/etd>

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

Scholar Commons Citation

Hopper, Matthew S., "Embedded System Design of Low-Power Wearable Bioelectronic Devices" (2018). *Graduate Theses and Dissertations*.

<http://scholarcommons.usf.edu/etd/7168>

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

Embedded System Design of Low-Power Wearable Bioelectronic Devices

by

Matthew S. Hopper

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.
Sriram Chellappan, Ph.D.
Andrew M. Hoff, Ph.D.

Date of Approval:
March 9, 2018

Keywords: Wearables, Eye Tracking, Polysomnography, PLB, Microcontroller, DSP

Copyright © 2018, Matthew S. Hopper

ACKNOWLEDGEMENTS

I would first like to express my gratitude to Dr. Stephen Sadow for being an exemplary advisor and professor through my time at USF and providing me with opportunities to continue developing as an electrical engineer. I would like to thank Dr. Andrew Hoff and Dr. Sriram Chellappan for being committee members for this Master's degree.

I would also like to thank my brother, Michael Hopper, for always being there for me to bounce ideas off him. Thank you to my parents Cindy Murdocca, Jeff Hopper and Vicki Horvath-Hopper for your continued support through this process.

As the eye-tracking computer mouse originated as a capstone project for my BSEE, I would also like to acknowledge and thank Mohammed Alshelaly, Jonathan Daniels and Shayna White for their roles in the development of the device. I also wish to thank Dr. Fabiola Araujo Cespedes of our research group for her assistance at the start of the capstone project and all members of the USF SiC Group over the years.

Last but not least, I would like to thank ISPA Technology and SOFWERX for providing funding for me during the duration of this work.

TABLE OF CONTENTS

LIST OF TABLES.....	iii
LIST OF FIGURES	iv
ABSTRACT.....	vii
CHAPTER 1 : INTRODUCTION.....	1
1.1 Wearable Bioelectronics Overview.....	1
1.2 Motivation and Current Capabilities	2
1.2.1 Computer Mouse for Physically Disabled.....	2
1.2.2 Polysomnography.....	3
1.2.3 Personal Locator Beacon	3
1.3 Thesis Organization	4
CHAPTER 2 : ELECTROOCULOGRAPHIC EYE-TRACKER	5
2.1 Device Overview.....	5
2.2 Initial Design	8
2.2.1 Amplification and Active Filter Design.....	8
2.2.2 Breadboard Prototype.....	13
2.2.3 PCB Design.....	15
2.2.4 Device Programming and Algorithms.....	17
2.2.4.1 Calibration Sequence	17
2.2.4.2 Main Loop	19
2.2.5 Device Testing.....	20
2.3 Miniaturized Wearable Device Design.....	21
2.3.1 Integrated System Solution.....	22
2.3.2 PCB Design.....	22
2.3.3 Device Programming and Algorithms.....	23
2.3.4 Device Testing.....	25
2.4 Summary	28
CHAPTER 3 : WIRELESS SYSTEM FOR IN-HOME SLEEP STUDIES.....	29
3.1 Device Concept Overview.....	29
3.1.1 Epidermal Patch Design	30
3.1.2 Electronics Module Design	32
3.2 Device Design.....	34
3.2.1 Analog Front End.....	36
3.2.2 Respiratory Impedance Plethysmography	37
3.2.3 Piezoelectric Transducer with Temperature Detection	43
3.2.4 Pulse Oximeter and Inertial Measurement Unit.....	43
3.2.5 Digital Signal Processing and Data Storage	44
3.3 Device Testing	45
3.4 Summary	49

CHAPTER 4 : TRAUMA-DETECTING PERSONAL LOCATOR BEACON	51
4.1 Device Overview	51
4.2 Device Design.....	51
4.2.1 SPO ₂ and Pulse.....	52
4.2.2 Electrodermal Activity	53
4.2.3 Freefall and Impact Detection	54
4.2.4 Skin Temperature	55
4.2.5 Microcontroller and Algorithms	55
4.2.6 PLB Transmitter.....	57
4.3 Summary	57
CHAPTER 5 : CONCLUSIONS AND FUTURE WORK	58
5.1 Conclusions	58
5.2 Future Work.....	59
REFERENCES	60
APPENDICES.....	62
Appendix A: Initial Eye-Tracker Schematics	63
A.1 Breadboard.....	63
A.2 Initial PCB.....	64
Appendix B: Initial Eye-Tracker Program	67
Appendix C: Wearable Wireless Eye-Tracker Schematics.....	72
Appendix D: Eye-Tracker Program for MSP430F5529	75
Appendix E: Wearable Wireless Polysomnography Schematics.....	86
Appendix F: Copyright Permissions.....	93

LIST OF TABLES

Table 2.1: Eye tracker initial design breadboard components	14
Table 2.2: Eye tracker initial design PCB components	16
Table 2.3: Wearable wireless eye-tracker part list	23
Table 2.4: AFE configuration register values	24
Table 2.5: SNR of 2 nd iteration eye-tracker signals	26
Table 3.1: AFE configuration register values for wearable sleep sensor.	36
Table 3.2: Wearable sleep sensor filtering and sampling requirements	44
Table 4.1: Examples of emergency conditions and sensor criteria for alarm activation	55

LIST OF FIGURES

Figure 2.1: Corneoretinal potential	6
Figure 2.2: Proposed EOG electrode placement for eye tracker	7
Figure 2.3: Orbicularis oculi signal artifacts in EOG signal	7
Figure 2.4: Instrumentation amplifier circuit.....	9
Figure 2.5: Sallen-Key topology low-pass filter.....	10
Figure 2.6: 5th order low-pass Butterworth filter circuit.....	11
Figure 2.7: PSpice frequency response of each stage of filtering	11
Figure 2.8: PSpice frequency response of overall filter.....	11
Figure 2.9: Precision half-wave rectifier	12
Figure 2.10: Block diagram of initial eye tracker system.....	13
Figure 2.11: Digital photograph of initial design breadboard circuit.....	13
Figure 2.12: Digital photograph of the initial design printed circuit board (no solder mask).....	15
Figure 2.13: Measured Butterworth filter response.....	21
Figure 2.14: Digital photograph of miniaturized wearable design printed circuit board.....	22
Figure 2.15: Wearable eye-tracker microcontroller program flow chart.....	25
Figure 2.16: EOGH eye rotations in 5° increments.....	27
Figure 2.17: EOGV eye rotations in 5° increments	27
Figure 3.1: Comparison of conventional polysomnography (left) to proposed system (right)	29
Figure 3.2: Elastomer patch locations and sensors	30
Figure 3.3: Proposed wearable sleep sensor device electronics module.....	32
Figure 3.4: Block diagram of proposed wireless sleep sensor electronics module.....	33
Figure 3.5: Sleep sensor prototype testing configuration.....	34

Figure 3.6: Digital photograph of wireless wearable sleep sensor PCB.....	35
Figure 3.7: Device power logic truth table (left) and circuit (right)	35
Figure 3.8: Wein bridge oscillator configuration.....	38
Figure 3.9: PSpice transient response of Wein bridge oscillator	38
Figure 3.10: PSpice frequency response of Wein bridge oscillator	39
Figure 3.11: Voltage-to-current converter circuit.....	40
Figure 3.12: PSpice voltage-to-current converter voltage simulation through R_L	40
Figure 3.13: PSpice voltage-to-current converter current simulation through R_L	41
Figure 3.14: Precision envelope detector circuit configuration.....	42
Figure 3.15: Transient response PSpice simulation of precision envelope detector	42
Figure 3.16: Digital photograph of respiratory flow sensor board.....	43
Figure 3.17: Digital photograph of facial sensor	43
Figure 3.18: EEG signal obtained by electrodes placed in the F7 and Fp1 locations.....	45
Figure 3.19: EEG signal obtained by electrodes placed on the nape of the neck	46
Figure 3.20: EOG signal obtained by electrodes placed along the lateral edge of the eye	46
Figure 3.21: EMG signal obtained by electrodes placed on the masseter muscle of the face....	47
Figure 3.22: EKG signal obtained by electrodes placed on the chest	47
Figure 3.23: Respiratory flow signal obtained through piezoelectric pressure transducer.....	48
Figure 3.24: Respiratory flow signal obtained through thermistor	49
Figure 4.1: Trauma-activated PLB block diagram.....	52
Figure 4.2: Pulse oximeter concave support structure.....	53
Figure 4.3: Wheatstone bridge configuration	54
Figure 4.4: Flow diagram of trauma-detecting PLB microcontroller program	56
Figure A.1: Breadboard schematic of eye-tracker	63
Figure A.2: Initial eye-tracker PCB microcontroller interface.....	64
Figure A.3: Initial eye-tracker PCB power supply	65

Figure A.4: Initial eye-tracker PCB signal conditioning	66
Figure C.1: Wearable wireless eye-tracker microcontroller and peripherals interface.....	72
Figure C.2: Wearable wireless eye-tracker power supply	73
Figure C.3: Wearable wireless eye-tracker analog front end interface.....	74
Figure E.1: Wearable wireless sleep study system power supply.....	86
Figure E.2: Wearable wireless sleep study system DSP and eMMC interface	87
Figure E.3: Wearable wireless sleep study system power supply logic and JTAG header.....	88
Figure E.4: Wearable wireless sleep study system analog front end interface	89
Figure E.5: Wearable wireless sleep study system respiratory impedance plethysmography	90
Figure E.6: Wearable wireless sleep study system flash, RTC and external board interface	91
Figure E.7: Wearable wireless sleep study system external IMU and SPO ₂ module	92
Figure E.8: Wearable wireless sleep study system external piezoelectric module.....	92

ABSTRACT

The miniaturization of electronics in modern times has enabled the possibility of creating a “continuity of care” using small wearable bioelectronic devices. Using wearable devices, such as the Fitbit or Garmin fitness trackers, allows for the exchange of data between devices which can be used to improve the accuracy of data analysis and thus patient health.

In this thesis work, three wearable bioelectronic devices are proposed: an EOG-based eye-gaze tracking assistive technology device for the physically disabled to control a computer cursor, a battery-operated miniaturized polysomnograph that can store and transmit data wirelessly to sleep technicians and a trauma-detecting personal locator beacon. The first two system designs are outlined and simulated, followed by the testing of a prototype while the third system is a proposed design that will be reduced to practice at a later date.

With continued development needed in the signal processing algorithms, the eye-gaze tracking computer mouse demonstrated capability and repeatable results. The wearable sleep sensor system also demonstrated capability and provided data with high signal-to-noise ratios on most channels before any filtering, allowing for comparable signal quality to conventional polysomnography devices.

CHAPTER 1 : INTRODUCTION

1.1 Wearable Bioelectronics Overview

The use of bioelectronics has become an essential part of the healthcare system for patient diagnostics, the treatment of medical conditions and restoration of functionality to those with disabilities. The invention of the transistor revolutionized the healthcare industry by creating a level of treatment that would otherwise have not been possible. However, until recently, most of these devices have been limited to use by medical professionals in healthcare settings due to the difficulty of use, large system size and high cost.

Thanks to the new possibilities of small wearable electronics, or wearables, the development of wearable bioelectronics has exploded in recent years. Wearables have enabled the adoption of “continuity of care” within the healthcare community, allowing for the possibility of prompt access to medical data without limiting or interrupting the patient’s activities. This broadened scope of telemedicine, as it is also referred to, can be used to improve healthcare access and quality, regardless of patient location [1]. With the developing scope of the internet of things (IoT), this is already truer than ever. The exchange of information between devices has allowed for larger datasets and can ultimately allow for better accuracy in data interpretation.

However, these consumer-level medical devices require additional considerations in their development. As with any wearable device, the system must remain low-profile and low-power in order to be unobtrusive to the user’s activities. This presents a significant challenge for bioelectronic wearables due to the large variety of sensor locations and the significant levels of signal amplification and filtering needed. In addition, poor interfacial contact or improper placement of the various sensors can cause inaccurate readings of the physiological conditions.

In this thesis work, the design of three wearable bioelectronic devices are proposed: an electro-oculographic (EOG) eye-gaze tracking computer mouse for persons with higher levels of disability or paralysis, a wireless polysomnography device for in-home sleep studies, and a trauma-detecting personal locator beacon. These devices each present their own set of challenges for miniaturization while retaining signal integrity and the relevant challenges will be presented with each case.

1.2 Motivation and Current Capabilities

This section outlines the motivation of the proposed devices. The current capabilities and limitations of similar devices to the proposed designs are also discussed.

1.2.1 Computer Mouse for Physically Disabled

In the United States, 32,159 people developed some degree of paralysis in the United States between 1972 and 2016, or about 40.1 per million people [2, 3]. Of these people, 16,398 people have either complete or incomplete forms of quadriplegia [2]. With the growing reliability of computers and smartphones in our daily life, many of these individuals have difficulties maintaining access to these devices. A number of assistive technology devices have been developed to help them regain access, ranging from joystick control with the mouth and head movement, such as the QuadStick or TetraMouse, to tracking head motion, such as the HeadMouse. However, injuries to C1 or C2 of the cervical spinal cord, in addition to neurodegenerative disorders such as multiple sclerosis or cerebral palsy, can cause partial or complete loss of head movement as well, rendering these devices impractical for these individuals [4].

Due to the high prices of these assistive devices, many often cannot afford them due to other expenses associated with treating their conditions. The development of an inexpensive

computer mouse that doesn't rely on head movement, extremities or speech recognition could help improve the lives of a large group of these individuals.

1.2.2 Polysomnography

Every year, approximately 1,170,000 sleep studies are performed in the United States. With only 2,500 sleep centers in operation as of 2012, patients often have to wait between 2-10 months until an available appointment [5, 6]. Currently, the gold standard method of measuring sleep stages and detecting problems is using polysomnography. However, due to alteration of sleep architecture caused by the limitations of sleep positions and the unfamiliar location, also known as the "first night effect" (FNE), patients often have difficulties sleeping during the sleep studies [7]. Often times, this causes data from the entire first night of a polysomnography to be excluded from the analysis report [7].

Current polysomnograph studies often do provide conclusive results. However, an improvement in the patient's comfort could help reduce the number of nights the patient is needed for the study. While other factors prevent all sleep studies from being performed at home, improvement of the patient's interaction with the device, including application/removal time and mobility during sleep, could help improve the accuracy of the study. In attempt to overcome this problem, other devices such as the WM Home Sleep Test Device allow for in-home sleep studies. However, these devices have limited sensors and cannot replace the conventional polysomnography.

1.2.3 Personal Locator Beacon

Whether due to unexpected bad weather or clumsiness, medical emergencies are always a possibility when exploring the wilderness. Within just the US National Park Service, 65,439 search-and-rescue incidents were reported between 1992 and 2007, ultimately resulting in 2,659 deaths [6]. With no direct access to emergency services in many of these locations,

possession of a personal locator beacon (PLB) becomes necessary. Monitored by the International Cospas-Sarsat Programme, emergency responders can be dispatched to the GPS locations of distress signals to provide emergency care.

While this has become an invaluable resource for individuals exploring remote locations, many lives have been lost due to incidents preventing the ability of PLB activation (i.e. loss of consciousness). Automatic detection of this trauma or medical emergency could help provide access to emergency services for these individuals.

1.3 Thesis Organization

In Chapter 2, a proposed solution for an inexpensive computer mouse assistive technology is outlined. The basic design of the device is first built and tested. A simpler and more effective design is then proposed that further miniaturizes the device to make it more feasible as a wearable device. Chapter 3 outlines a proposed solution to improving patient experience in sleep studies. A concept is first outlined, simulated and a prototype board is then explained and tested. Chapter 4 outlines a concept for a trauma-detection device that can trigger a distress beacon if an injured hiker is unable to enable it themselves. Chapter 5 includes an analysis of the testing results, provides a conclusion and recommends future development of each device. Finally, the appendices include relevant schematics and programs used in the discussed devices.

CHAPTER 2 : ELECTROOCULOGRAPHIC EYE-TRACKER

2.1 Device Overview

In order to provide individuals with computer access in even the worst cases of quadriplegia, eye-gaze tracking was chosen for this device. In order to achieve this, three methods of eye-tracking are possible:

- *Eye-Attached Tracking:* The use of tight-fitted contact lenses with sensors surrounding the eye [8],
- *Optical/Video Tracking:* The use of cameras to detect either reflected light or machine learning algorithms for the detection of pupil location [9], or
- *Electric Potential:* Correlating electric potential changes around the eyes to eye movement [10].

As costly user customization is required for eye-attached tracking, and optical tracking is often bulky or otherwise inaccurate, the electric potential method was used in this design.

This method utilizes the corneoretinal potential (CRP) that can be found between the front and the back of the eye. Due to the higher metabolic rates of the retina in comparison to the cornea, a negative charge is built up along the back of the eye, generating an electric potential difference of 0.4 – 1.0 mV [11]. By placing electrodes on either side of the eye, rotation towards the right can cause the right electrode to have a larger positive potential than the left, and vice versa, as shown in Figure 2.1.

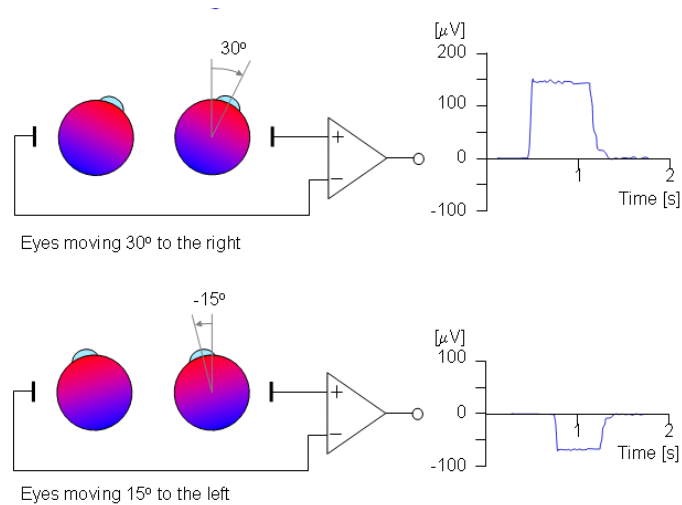


Figure 2.1: Corneoretinal potential. Detected through comparing eye movement of 30° to the right (top) and 15° to the left (bottom). Note an electrical pulse, corresponding to eye movement based on EOG, is readily observed [11].

Using this concept, electrodes can be placed around the eyes for interpretation into x and y-direction cursor movement. By placing electrodes in the configuration of Figure 2.2, this can be achieved. For the EOGH (electrooculography-horizontal) direction, electrodes can be placed near the lateral corners of the eyes. While the optimal placement of the electrodes would be on either side of the same eye, one electrode would be required to be placed along the nasal bridge, causing discomfort to the user. However, due to the identical movements of each eye, the same potential profile exists across both eyes. In addition, the reference electrode on the forehead can help remove signals originating from the electroencephalographic (EEG) signals of the brain using common mode rejection.

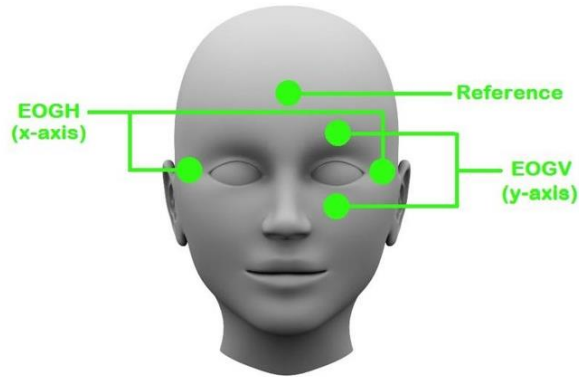


Figure 2.2: Proposed EOG electrode placement for eye tracker. One reference electrode placed on the forehead.

In order to enable the user to left or right-click something on their computer screen, blink patterns can be used to trigger a left or right click. Due to prominent signal artifacts from the orbicularis oculi muscles, as can be observed particularly in the y-axis channel, the rapid spikes in the signal can be correlated to a blink as shown in Figure 2.3.

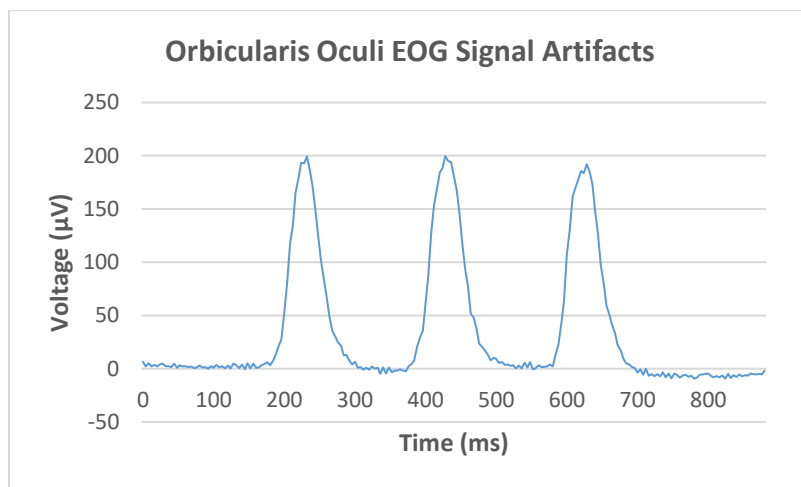


Figure 2.3: Orbicularis oculi signal artifacts in EOG signal. Can be used to perform mouse click functions on-screen.

In order to maintain calibration when the head is moved, an inertial measurement unit (IMU) can be used to track head movement. By tracking the Earth's magnetic field using the magnetometer to find the rotation angle, a gyroscope for head tilt and an accelerometer for

head displacement, head movement can be considered in the calibration sequence to maintain accuracy. This has not yet been incorporated into the design but can readily be implemented in a commercial system at a later date.

2.2 Initial Design

Due to the small signal received by the EOG electrodes, the signal must be amplified and filtered before use. The device was first designed using discrete cascaded operational amplifiers (op-amps) for amplification, isolation, filtering and rectification. As the signal ultimately received by the analog-to-digital converter (ADC) has a linear correlation with eye movement, a calibration sequence can allow the signal to be translated directly into cursor coordinates. This initial design was first assembled and verified on a breadboard before being designed into a printed circuit board (PCB).

2.2.1 Amplification and Active Filter Design

In order to process the EOG signals for measurement by the ADC, the signal was processed through 8 stages of amplification and filtering. These stages were designed for an overall amplification of 1000 – 43000 V/V and filtering out frequencies greater than 20Hz.

1. *Instrumentation Amplifier* – Due to the EOG signals measuring only a few microvolts (μV) in amplitude, this op-amp configuration must be used to measure the differential potential. In contrast to a conventional differential amplifier layout, the instrumentation op-amp has a high input impedance that prevents unwanted drops in voltage, greatly improving its accuracy. The gain of this circuit can be calculated using equation 2.1.

$$\frac{V_{out}}{V_2 - V_1} = \left(1 + \frac{2R_1}{R_{gain}}\right) \frac{R_3}{R_2} \quad (2.1)$$

Due to the difficulty in obtaining highly impedance-matched resistors for the circuit, instrumentation amplifier ICs were used in this device. In this stage, R_{gain} was configured to provide a 100 V/V open loop gain.

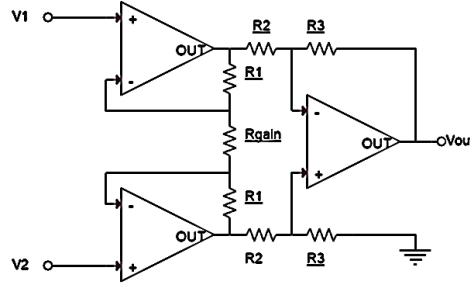


Figure 2.4: Instrumentation amplifier circuit. Gain was set to 100 V/V.

2. *Non-Inverting Amplifier* – As a means for additional amplification of the signal, a standard non-inverting op-amp configuration was used. Using a potentiometer in place of the feedback resistor, an additional gain of up to 51 V/V was added.
3. *Linear Optoisolation Amplifier* – In order to provide the user with additional isolation from potential electrical faults, a linear optoisolator was used. Due to the non-linear characteristics of a diode, an LED/photodiode isolation barrier also has a non-linear output. While ideal for digital applications, additional challenges must be overcome for use in analog applications. Components such as the Texas Instruments ISO124 can be used for precision linear isolation. By using a duty cycle modulation-demodulation technique, isolation can be achieved through a 2pF capacitive barrier [12]. Unity gain was used in this stage.
4. *Low-Pass Filter* – After providing enough amplification to the signal, frequencies outside the needed range can be filtered. Since EOG signals are largely in the 0 – 15 Hz frequency range, a 5th order active low-pass Butterworth filter was designed to provide a sharp cutoff without attenuating the signals of interest. This was developed using two Sallen-Key topology low-pass filters, selected due to its simplicity for a

high-impedance input and 2-pole filter response, and one additional low-pass RC filter.

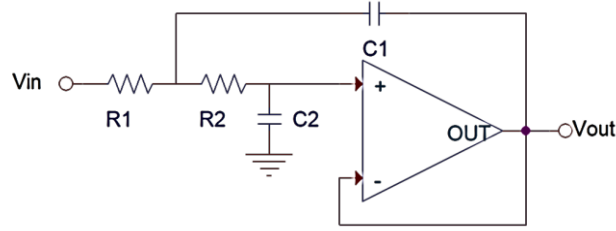


Figure 2.5: Sallen-Key topology low-pass filter. Design goal was a corner frequency of 20 Hz with the measured corner frequency of ~19.5 Hz which was more than sufficient for this application.

Using the Sallen-Key topology shown in Figure 2.5, the transfer function $H(s)$ in equation (2.2) can be used to calculate its frequency response. The natural frequency ω_0 can be calculated using equation (2.3) and the damping factor ζ using equation (2.4).

$$H(s) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \quad (2.2)$$

$$\omega_0 = 2\pi f_0 = \frac{1}{\sqrt{R_1 R_2 C_1 C_2}} \quad (2.3)$$

$$2\zeta\omega_0 = \frac{1}{C_1} \left(\frac{R_1 + R_2}{R_1 R_2} \right) \quad (2.4)$$

In this case, the two Sallen-Key filters were designed to have a natural frequency of 18.94 Hz and 19.43 Hz, respectively. The additional RC filter was designed for a frequency cutoff of 19.41 Hz, as shown in equation (2.5).

$$\omega_C = \frac{1}{\tau} = \frac{1}{RC} \quad (2.5)$$

Using these parameters, the circuit in Figure 2.6 was obtained and verified through a frequency response analysis in PSpice. Each individual stage's response can be found in Figure 2.7 and the overall cascaded response can be found in Figure 2.8.

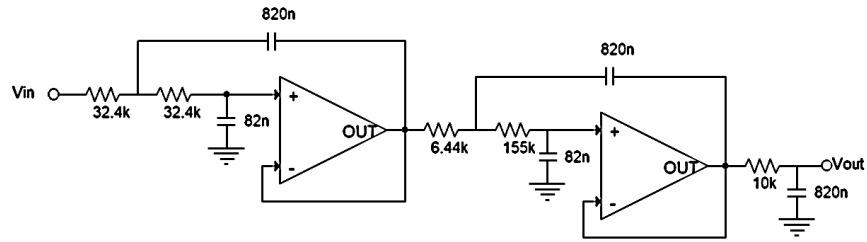


Figure 2.6: 5th order low-pass Butterworth filter circuit. Designed for a 20Hz cutoff frequency.

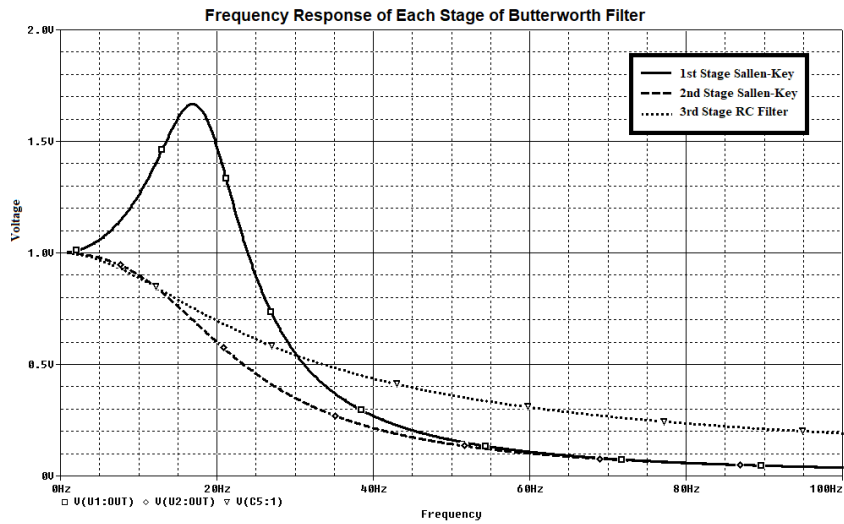


Figure 2.7: PSPice frequency response of each stage of filtering. Stages shown in Figures 2.5 and 2.6. Note that the 3 dB cutoff frequency design goal of 20 Hz was achieved when all three stages were simulated in cascade.

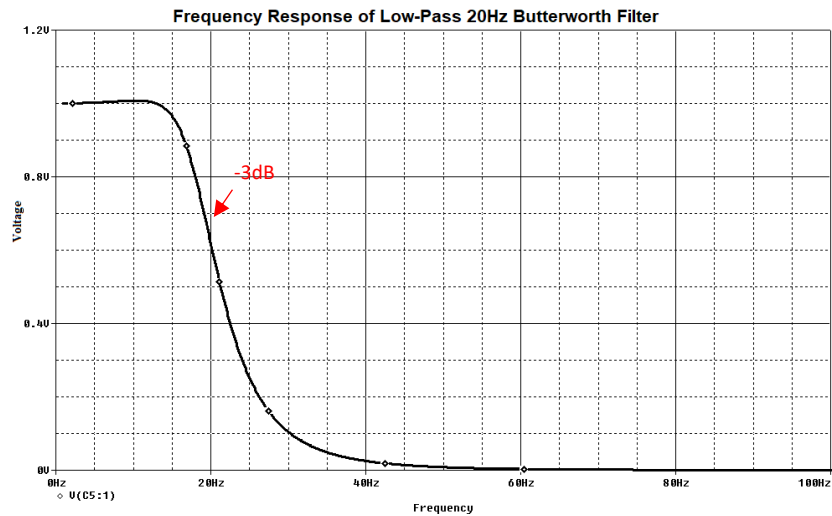


Figure 2.8: PSPice frequency response of overall filter. Low-pass Butterworth filter with -3 dB point at 20 Hz.

5. *Unity Gain Buffer* – Due to the high output impedance of the RC filter, a unity gain buffer was needed to maintain the signal integrity. The impedance transformation provides a stronger signal to the next stage and prevents overloading of the filter by the load.
6. *Inverting/Non-inverting Amplifiers* – In order to collect both the positive and negative signals in the ADC, which can only read values greater than the 0V reference, the signal was divided into 2 channels and further amplified with a non-inverting and inverting amplifier. This stage provided an additional 4.2 V/V gain.
7. *Precision Half-Wave Rectifier* – Due to the $\sim 0.7V$ voltage drop from a series diode, a precision half-wave configuration was used for rectification instead. By using this configuration on both the inverted and non-inverted channels, the negative components of each signal can be cut off for safe sampling by the ADC. These two channels are later summed in the microcontroller program. This stage provided an additional 2 V/V gain.

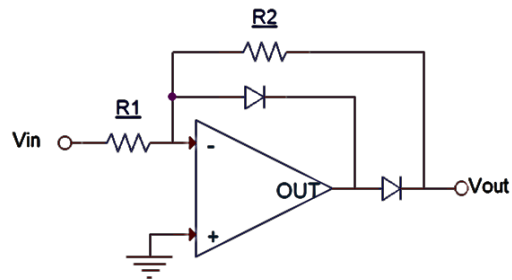


Figure 2.9: Precision half-wave rectifier. Used to ensure proper signal processing in the on-board microcontroller circuit.

8. *Right-Leg Driver* – Since the body also acts as an antenna for signals in the surrounding environment, such as 60Hz noise from the power mains in a room, a common-mode rejection circuit was used. By collecting the common mode signals from the center-tap of R_{gain} on each instrumentation amplifier, these signals can be

inverted, amplified and driven into the reference electrode to actively cancel these signals.

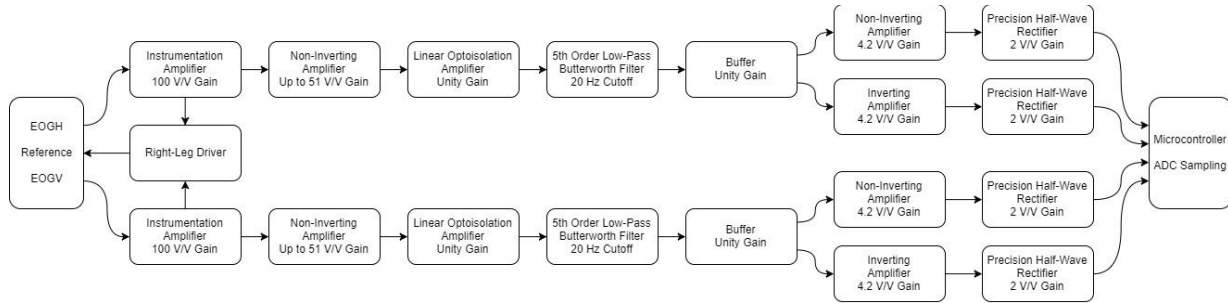


Figure 2.10: Block diagram of initial eye tracker system. Diagram shows EOG signal inputs (horizontal and vertical, left of figure) and final processed signals feeding the microcontroller after ADC sampling (right of figure).

2.2.2 Breadboard Prototype

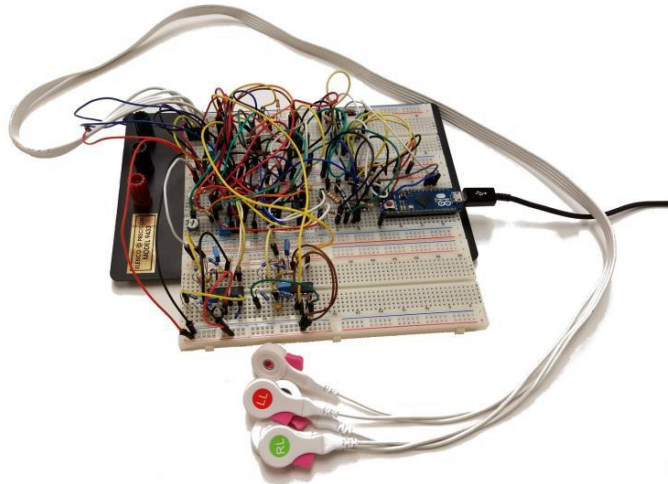


Figure 2.11: Digital photograph of initial design breadboard circuit. Includes leads to EOG electrodes.

In order to verify operation of this configuration, the circuit was first assembled on a breadboard using the components as outlined in table 2.1. As use of the linear isolation amplifier doesn't provide amplification or filtering in this circuit, in addition to consideration of the added cost, this component was omitted from this iteration of the design. To provide additional safety to the user, a 50mA current limit was set on the $\pm 12V$ power supply. Using a 50k Ω

potentiometer as the feedback resistor and a 5k Ω potentiometer between the offset pins, gain and offset could be manually adjusted on the variable amplifier stage. The ADC port of an Arduino Micro was used to collect and combine the data and transmitted through a UART COM port to a computer. This Arduino was later replaced by an MSP430F5529 TI Launchpad in order to allow for simple translation to the PCB microcontroller. The full schematic of this iteration can be found in Appendix A.

Table 2.1: Eye tracker initial design breadboard components

Stage	Component	Quantity	Model	Comments
Instrumentation Amplifier	Instrumentation Amplifier	2	INA128P	Used to avoid impedance matching resistor values
Variable Amplifier	General Purpose Op-Amp	2	LM741CN	OFFSET NULL pins used for variable offset, feedback resistor for variable gain
Linear Isolation	Linear Optoisolation Amplifier	0	-	Emitted from this iteration
Butterworth Filter	General Quad Op-Amp	2	LM324AN	-
Unity Gain Buffer	General Quad Op-Amp	-	LM324AN	Included in Butterworth filter component
Inverting and Non-Inverting Amplifier	General Purpose Op-Amp	4	LM741CN	-
Rectifier	General Quad Op-Amp	2	LT1079CN	Used for single-supply operation capability
Right Leg Driver	General Purpose Op-Amp	3	LM741CN	-
ADC Sampling	Microcontroller	1	Arduino Micro	Sampled on four ADC channels. Negative channels negated and added to positive channel

2.2.3 PCB Design

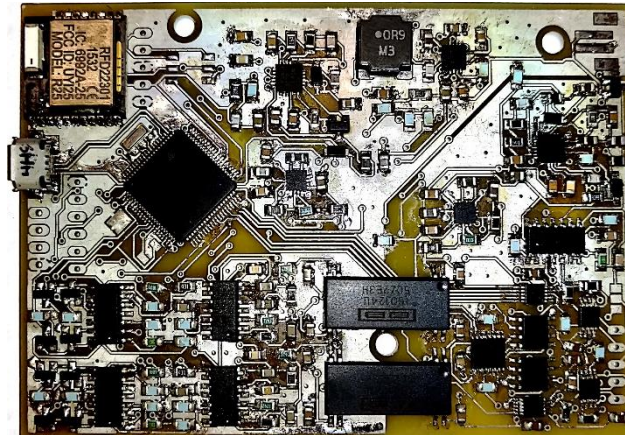


Figure 2.12: Digital photograph of the initial design printed circuit board (no solder mask). The board was fabricated by Advanced Circuits and populated in-house. Board dimension 9 by 6.5 cm.

In the second iteration of this design, the verified breadboard circuit was designed into an embedded system on a printed circuit board (PCB) with dimensions of 9 x 6.5cm. This PCB was designed and assembled using the components as outlined in Table 2.2. Powered by a single 3.7V 2000mAh lithium-ion battery, this circuit includes boost/buck DC-DC converters to provide the required $\pm 3.3V$ and $\pm 5V$ supplies. In addition, a second set of power supplies with an isolated ground plane was included in the design to provide additional noise isolation and safety to the user. This ground plane (ISO_GND) is connected to the primary ground (GND) through ferrites and isolates the instrumentation amplifiers, first stage of amplification, the right leg driver and the input of the isolation amplifier. In order to provide programmatic control of the gain and offset, digital potentiometers were used with SPI and I²C interfaces to the microcontroller.

The microcontroller included in this design, a TI MSP430F5529, was selected due to its low power capabilities and 12-bit ADC. In addition, its high-speed USB interface allows for the device to be programmed as a standard human interface device (HID) to be seen by the computer as a mouse. This will allow for the program written for the computer interface to be

ultimately limited to only the calibration process, requiring less of the computer's resources for operation. Alternatively, a UART-interfaced RF Digital RFD22301 module was also included in this design to allow for wireless communication with a computer or smartphone. While not directly included on this board, a header with an I²C interface allows for a head-mounted IMU to be connected to the device for compensation in the calibration step. The full schematic of this design can be found in Appendix B.

Table 2.2: Eye tracker initial design PCB components

Stage	Component	Quantity	Model	Comments
Instrumentation Amplifier	Instrumentation Amplifier	2	INA333	Chosen due to its small package and 50 μ A current consumption
Variable Amplifier	General Purpose Op-Amp	2	TLE2141	OFFSET pins used for variable offset, feedback resistor for variable gain
Gain Potentiometer	50k Ω Digital Potentiometer	2	AD5270	Used as feedback resistor for variable amplifier. Controlled by MCU
Offset Potentiometer	5k Ω Dual-Channel Digital Potentiometer	1	AD5258	Selected due to dual channel capability. Controlled by MCU
Linear Isolation	Linear Optoisolation Amplifier	2	ISO124U	-
Butterworth Filter	General Quad Op-Amp	2	AD8608	-
Unity Gain Buffer	General Quad Op-Amp	-	AD8608	Included in Butterworth filter component
Inverting and Non-Inverting Amplifier	General Quad Op-Amp	2	AD8608	-
Rectifier	General Quad Op-Amp	-	AD8608	Included in inverting/non-inverting amplifier component
Right Leg Driver	General Quad Op-Amp	1	AD8608	-
ADC Sampling	Microcontroller (12-bit ADC)	1	MSP430F5529	2 ADC channels per EOG channel. Negative channel negated and added to positive channel

Table 2.2: (continued)

Stage	Component	Quantity	Model	Comments
Wireless Communication	Bluetooth 4.0/BLE Module	1	RFD22301	Communicates with microcontroller through UART
Charging	Battery Charger	1	BQ24155	Communicates with microcontroller through I ² C
Power Supply	±3.3V Buck/Boost DC-DC Converter	2	TPS65130	Additional supply for ISO_GND
Power Supply	±5V Buck/Boost DC-DC Converter	2	TPS65135	Additional supply for ISO_GND
Battery	Li-ion 2000mAh 3.7V Battery	1	GSP 585460	-

2.2.4 Device Programming and Algorithms

To enable a smooth transition between the breadboard and PCB, in addition to simplicity of troubleshooting hardware interfaces, the Texas Instrument's Energia IDE was used for programming this iteration of the design. In addition, a C# program was written using Microsoft Visual Studio for assisting in the calibration process and converting COM data to cursor movement.

This program begins its setup() function by initiating the communication interfaces with the computer COM port and digital potentiometers and proceeds to halt the program until the COM port is paired with the computer.

2.2.4.1 Calibration Sequence

In order to translate the ADC information into cursor coordinates, the microcontroller must first collect information on the dimensions of the screen and record information on the user's gaze. The program first waits for a string of data in the format: "@x-dim,y-dim" (i.e. @1024,0768). The "@" symbol is used to verify that the received data is intended for the device

and the “,” is used as a delimiter between dimensions. Once the required string is received, a “!” is sent back to the computer to indicate successful pairing.

The calibration() function then proceeds to wait for incrementing values from 0 to 6, each indicating the user is ready for each step of the calibration sequence. Each time a channel value is sampled, the negative channel value is negated and added to the positive channel to obtain the full signal.

- *Step 0 - Center:* The vertical and horizontal gains are first set to 1024, which correspond to their lowest resistance value. With the user looking at the center of the screen and the offsets set to their maximum value, the vertical offset is then decremented until the recorded signal is within a ± 10 , or $\pm 32.2\text{mV}$, tolerance range. This process is then repeated for the horizontal offset. If a value within the tolerance is not recorded after sweeping the entire range of the potentiometers, a “0” is sent to the computer to inform the user of a failed calibration. If successful, a “1” is sent to continue into step 1.
- *Step 1 - Top:* Now focusing the user’s gaze on the top of their screen, the vertical gain is incremented until a value is received within the tolerance range from 3V. This allows for a baseline maximum value for the range. If a value within the tolerance is not recorded after sweeping the entire range of the potentiometer, a “0” is sent to the computer to inform the user of a failed calibration. If successful, a “1” is sent to continue into step 2.
- *Step 2 - Bottom:* The user then focuses on the bottom edge of the screen and the vertical offset is decremented until a value within the tolerance range from -3V. The total number of decrement iterations is then divided by 2 and added back to the offset potentiometer to place the center position back at 0. If a value within the tolerance is not recorded after sweeping the entire range of the potentiometer, a “0” is sent to the computer to inform the user of a failed calibration. If successful, a “1” is sent to continue into step 3.

- *Step 3 - Left:* Step 1 is repeated for the horizontal gain with the user's gaze on the left edge of the screen.
- *Step 4 - Right:* Step 2 is repeated for the horizontal offset adjustment with the user's gaze on the right edge of the screen.
- *Step 5 – Top Left:* The user then looks at the top-left corner of their screen. In this step, the minimum x and y values are recorded for signal-to-cursor coordinate calculation.
- *Step 6 – Bottom Right:* The user then looks at the bottom-right corner of their screen. In this step, the maximum x and y values are recorded for signal-to-cursor coordinate calculation.

Using formulas (2.6) and (2.7), these maximum and minimum values are used to calculate a multiplier that can later be used to convert raw signals directly to pixel coordinates, where X_{mult} and Y_{mult} are the multipliers, X_{max} and Y_{max} are the maximum values, X_{min} and Y_{min} are the minimum values, and X_{dim} and Y_{dim} are the screen dimensions.

$$X_{mult} = \frac{X_{max} - X_{min}}{X_{dim}} \quad (2.6)$$

$$Y_{mult} = \frac{Y_{max} - Y_{min}}{Y_{dim}} \quad (2.7)$$

2.2.4.2 Main Loop

After a successful calibration, the program proceeds to an infinite loop that continuously reads ADC values and translates them to coordinates for the screen using equations (2.8) and (2.9). Since a screen's y-direction coordinate system is inverted in direction, equation (2.10) is also used to account for this.

$$X = \frac{X - X_{min}}{X_{mult}} \quad (2.8)$$

$$Y = \frac{Y - Y_{min}}{Y_{mult}} \quad (2.9)$$

$$Y = Y_{dim} - Y \quad (2.10)$$

In order to allow the user to left-click an item on their screen, the difference between each value is calculated. If three sequential coordinates have a y value of at least 32.2 mV greater than the last, a blink is detected. If two blinks are detected within a 1 second period, with at least 50ms between them, a click is triggered. In order to prevent the cursor from moving during this event, the coordinate from 1 second prior is stored and maintained as the coordinate until the click is complete.

These coordinates are assigned to a string and sent through the COM port in the format: "x-coordinate,y-coordinate,clickbit" (i.e. 874,237,0). This string is received by the C# program and uses the "," delimiters for breaking up the string and assigning a new cursor coordinate. If the click bit is read as a "1," a left click is triggered by the program.

2.2.5 Device Testing

In implementing the proposed design, the frequency response of the Butterworth filter was first measured. As shown in Figure 2.13, the measured response matched the simulated filter response and both measured a -3dB point of approximately 19.01Hz. While the filter was originally designed for 20Hz, rounding off component values to standard values caused a 0.99Hz deviation. However, this proved to be acceptable in this application due to the DC nature of the signal.

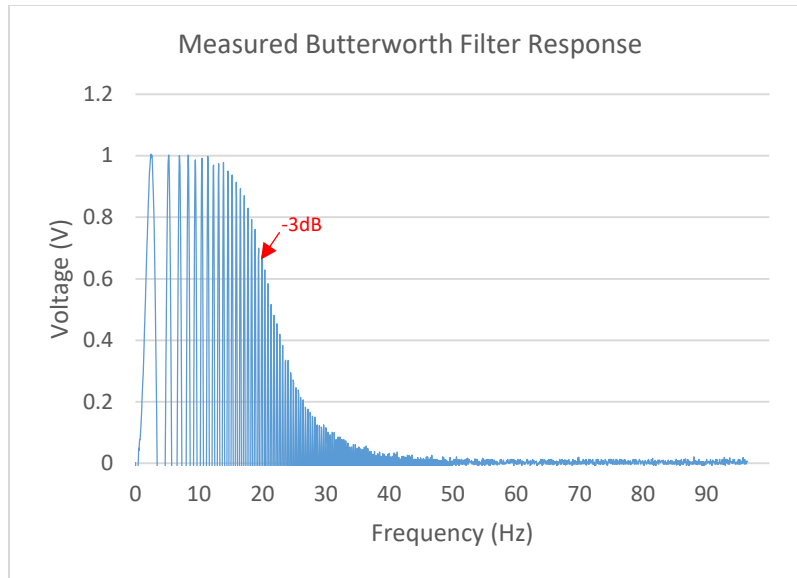


Figure 2.13: Measured Butterworth filter response. -3dB point at 19.01Hz, closely matching the simulated response from PSpice in Fig. 2.8.

In the initial observation of the output signals of the device, significant noise was still noted. To address this, a right-leg driver circuit was added which allowed for a signal-to-noise ratio of 41.27 to be obtained for the horizontal channel and 7.34 for the vertical channel. However, due to the drift of the signal, calibration was difficult and often times could not effectively be completed due to saturation of the signal. A third design was initiated to remove this challenge.

2.3 Miniaturized Wearable Device Design

In order to address the signal clipping issue found in the initial design, alternative methods of amplification and filtering were researched. As the previous PCB was too large for a head-mounted device, an alternative method was selected that would allow for a smaller footprint to reduce the overall cost of the system.

2.3.1 Integrated System Solution

As the ADC of the MSP430F5529 has a 12-bit resolution, a resolution of approximately 3.2mV can be obtained with its 3.3V supply. While the range of the ADC was doubled due to the two-channel sampling method used in the initial design, it proved to be ineffective at providing data clean enough for cursor control. Instead, an integrated solution was used.

This design iteration uses an ADS1299 analog front end from Texas Instruments for signal amplification, filtering and sampling. This component uses a 24-bit delta-sigma ADC architecture that can provide a resolution of approximately 0.3 μ V with its 5V supply. In addition, the ADS1299 has a programmable gain amplifier (PGA) on each channel for up to 24 V/V of gain and can be configured for common mode rejection in a right leg driver [13].

2.3.2 PCB Design



Figure 2.14: Digital photograph of miniaturized wearable design printed circuit board. Third iteration. The board was fabricated by PCBway and populated in-house. Board dimension 7.2 by 3.6 cm.

In order to provide the user with an additional level of safety, 2.2k Ω resistors were used in series with each electrode to limit current and 4.7nF capacitors were placed between each pair of inputs to filter the higher frequencies from the signal. In addition, ferrites were used to create an isolated ground for the analog connections on the AFE and the 5V supply powering it.

Since the size of this board is practical for mounting on the user's head, an IMU was also included in the design. In addition, LEDs and a button were included to allow for status

indication and reinitiating the calibration sequence when needed. The full schematic of the miniaturized wearable eye-tracker can be found in Appendix C.

Table 2.3: Wearable wireless eye-tracker part list

Stage	Component	Model	Comments
ADC Sampling	Analog Front End (24-bit ADC)	ADS1299	Two channels used with 24V/V gain and bias enabled for 1p and 2p. Remaining channels pulled high
Signal Processing and Data Transfer	Microcontroller (12-bit ADC)	MSP430F5529	-
Wireless Communication	Bluetooth 4.0/BLE Module	RFD22301	Communicates with microcontroller through UART
Head Movement Tracking	IMU	FXOS8700	Communicates with microcontroller through I ² C
Charging	Battery Charger	BQ24155	Communicates with microcontroller through I ² C
Power Supply	3.3V Buck/Boost DC-DC Converter	TPS6300	-
Power Supply	5V Buck/Boost DC-DC Converter	TPS61222	Grounded through ferrite for AFE isolation
Battery	Li-ion 2000mAh 3.7V Battery	GSP 585460	-

2.3.3 Device Programming and Algorithms

In order to improve reliability and capability of the device, the microcontroller was programmed using Code Composer Studio for this iteration of the design. While many of the algorithms remained the same as the initial design, this program requires an initialization procedure with the AFE to provide the parameters required for operation. Upon boot, the microcontroller is initialized by disabling the watchdog timer and defining the clock rate to 20MHz. After initializing UART and SPI interfaces, the register values in Table 2.4 are written to the AFE through SPI to configure channels 1 and 2 for reading the signals. The maximum gain

of 24 V/V is used and the bias is enabled for both positive electrodes. This configuration allows for the active cancellation of the common mode signals.

Table 2.4: AFE configuration register values

Register	Address	Value	Function
CONFIG1	01h	96h	Turns off daisy chain mode, disables clock output and sets sampling rate to 250 SPS
CONFIG2	02h	C0h	Configures test signal generation
CONFIG3	03h	ECh	Use 5V/2 as signal reference and enable bias for right-leg-driver
LOFF	04h	00h	Disable lead-off detection
CH[1:2]SET	05h	60h	Set up channels 1 & 2 as normal electrode inputs with 24 V/V of gain
CH[3:8]SET	07h	81h	Disable channels 3 – 8 by powering down and short their inputs for noise reduction
BIAS_SENSP	0Dh	03h	Enable bias sense for positive channels 1&2
BIAS_SENSN	0Eh	00h	Disable bias sense for all negative channels
LOFF_SENSP	0Fh	00h	Disable lead-off detection
LOFF_SENSN	10h	00h	Disable lead-off detection
LOFF_FLIP	11h	00h	Keep lead-off bias non-flipped
GPIO	14h	00h	Keep GPIO as output low
MISC1	15h	00h	Keep stimulus reference and bias switch open
CONFIG4	17h	00h	Continuous conversion mode with lead-off comparators off

After the initialization is complete, the program waits for an initial command from the computer. If a "@" is received, a calibration procedure is performed after receiving the screen resolution as outlined above. However, since this design reduces the likelihood of signal clipping, the calibration procedure doesn't require gain and offset control, allowing for equations (2.6) – (2.10) to be used for directly calculating the cursor coordinate. If a "!" is received, the data is streamed directly to the computer to allow for plotting of the signal or for offloading the calibration and tracking to a computer program. The process flow of this program can be found in Figure 2.15 and the full program can be found in Appendix D.

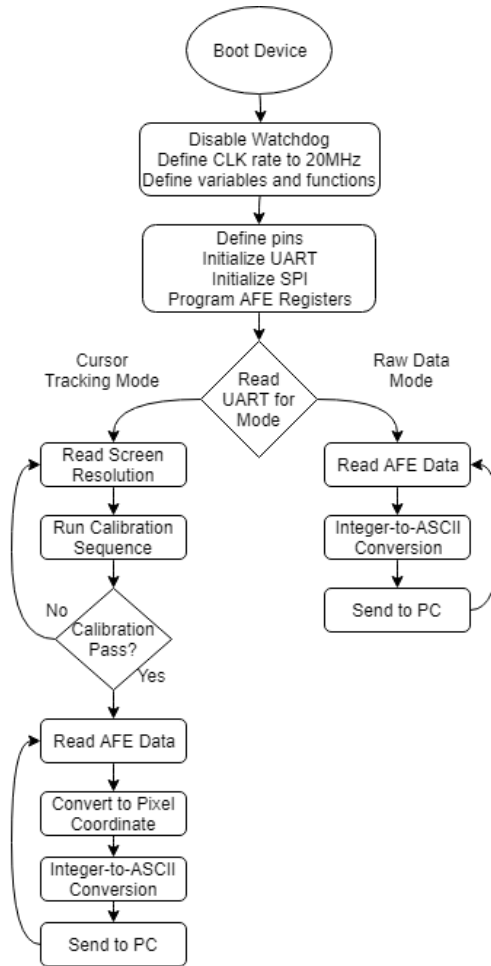


Figure 2.15: Wearable eye-tracker microcontroller program flow chart.

2.3.4 Device Testing

In implementing this improved design, the hardware problems noted in the initial design were found to have been resolved. However, additional filtering will be needed in order to enable use of this device for cursor control.

In order to compare the quality of the recorded signals, the signal-to-noise ratio (SNR) was calculated using equations (2.11) and (2.12). The root-mean-square (RMS) of the noise intensity was first calculated from data obtained while maintaining a stable gaze and displacing any offset in the signal. This was then repeated using the full range of signals to find V_{RMS} of the signal.

$$V_{RMS} = \sqrt{\frac{1}{n}(V_1^2 + V_2^2 + \dots + V_n^2)} \quad (2.11)$$

$$\frac{V_{RMS(Signal)}}{V_{RMS(Noise)}} \quad (2.12)$$

This process was repeated for both channels and compared during the use of the common-mode rejection driver and with a simple reference. As can be observed in Table 2.5, use of the right leg driver nearly doubled the SNR of the horizontal channel and improved the vertical channel by another two-thirds. As the vertical channel requires placement of an electrode on the forehead, this difference in noise rejection is due to the higher intensity of EEG signals collected by the vertical channel than the horizontal channel. In contrast to the initial design, implementation of the AFE in this design allowed for comparable SNRs to the ones previously found, before any filtering. Rather than using active filters as implemented in the initial design, a low-pass filter can be included in the microcontroller firmware to help prevent shakiness of the cursor.

Table 2.5: SNR of 2nd iteration eye-tracker signals.

Channel	Noise (V _{RMS})	Signal (V _{RMS})	Signal-to-noise Ratio
EOGH	4.99uV	118.32	23.71
EOGH with RLD	2.84uV	118.32	41.66
EOGV	6.26uV	46.1	7.36
EOGV with RLD	4.11uV	46.1	11.21

As tracking of eye rotation is used for controlling the eye cursor, accuracy was measured by tracking rotations in 5° increments. By sitting approximately 50cm from the screen, eye gaze was tracked starting from the center point with glances to the left and right in 4cm increments. This was repeated in the vertical direction for comparison of the channels. As can be observed in Figures 2.16 and 2.17, the horizontal direction was found to have moderate accuracy, with ~25µV steps between each increment. However, while general trends could be observed in the signal, the vertical direction was found to have significant drift. Within the sample plotted in

Figure 2.17, a $\sim 1\text{mV}$ drop in voltage was observed for the center point. A voltage drift of $\sim 100\mu\text{V}$ was also observed for the horizontal channel.

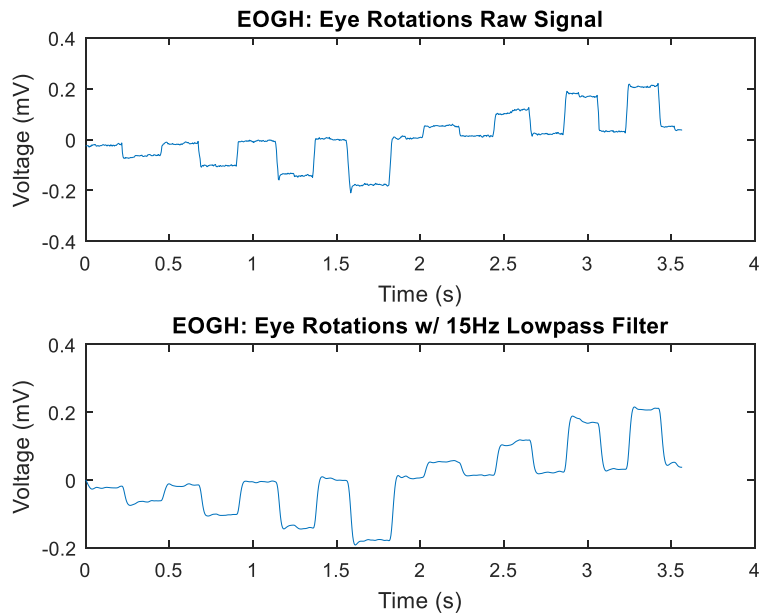


Figure 2.16: EOGH eye rotations in 5° increments. Raw signal in comparison to signal with 15Hz low-pass Butterworth filter, obtained through signal processing in MATLAB.

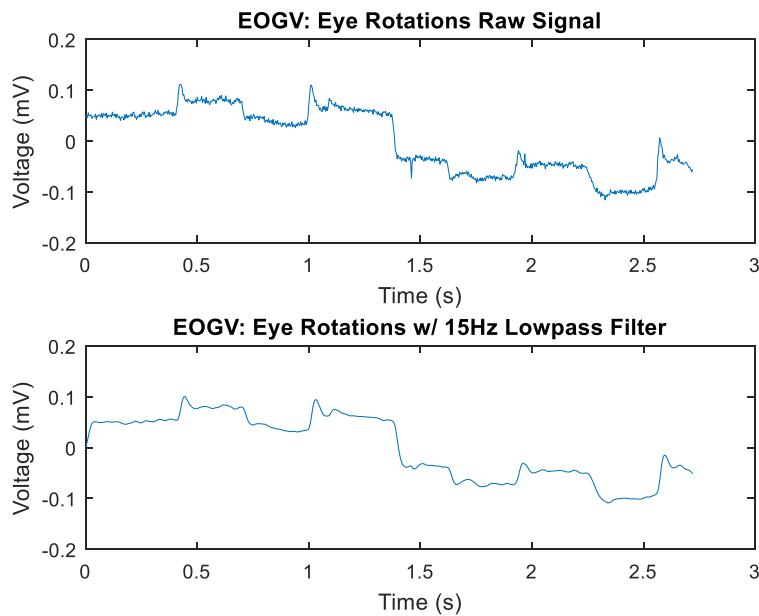


Figure 2.17: EOGV eye rotations in 5° increments. Raw signal in comparison to signal with 15Hz low-pass Butterworth filter, obtained through signal processing in MATLAB.

2.4 Summary

In this chapter, the design for an eye-gaze tracking computer cursor was proposed. Starting with a design of multiple stages of active filtering and amplification using discrete electrical components, this design was found to be too large for use in a wearable device. In addition, the required amplification frequently caused saturation of the signal, preventing reliable use of the device. An alternative was then proposed using an analog front end for miniaturization and simplification of the system. This design addressed the hardware issues found in the initial design, but will require additional development in signal processing in order to address the physical limitations of the use of electro-oculographic signals.

CHAPTER 3 : WIRELESS SYSTEM FOR IN-HOME SLEEP STUDIES

3.1 Device Concept Overview

In order to provide a more comfortable condition for patients undergoing sleep studies, an alternative device to the conventional polysomnography is proposed. As conventional sleep studies require dozens of leads connected to the patient, an hour is needed to setup the equipment and has to be disconnected and reconnected if the patient has to get out of bed (i.e. using the restroom). In addition, the patient is required to sleep on their back, making it difficult for many people to sleep. The goal of this device is to provide comparable data to the commonly used gold-standard devices in a low-profile device that can be mounted on the patient and transmit the collected data wirelessly.

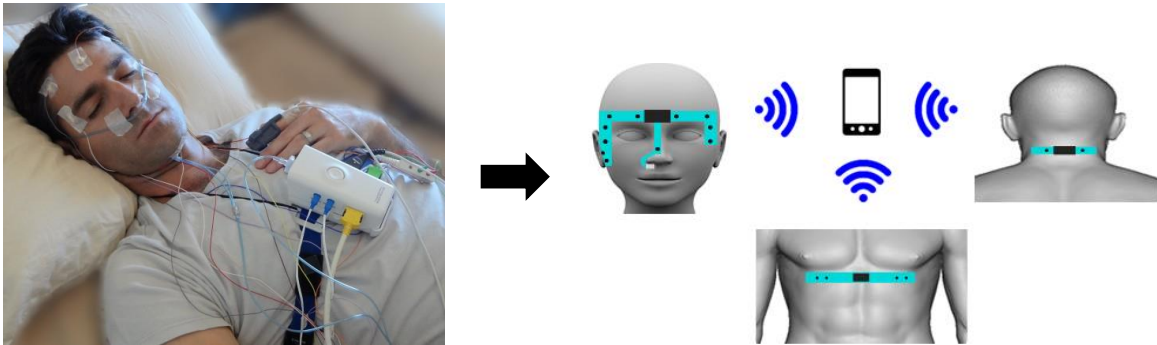


Figure 3.1: Comparison of conventional polysomnography (left) to proposed system (right). In the proposed system epidermal patches attach the sensors to the patient thus obviating the need for bulky cables and connections, greatly increasing patient comfort and allowing for in-home monitoring.

Due to the wide variety of physiological conditions required to be monitored during sleep studies, the sensors used in this system had to be divided into three separate subsystems (patches) located in the facial, nape (neck) and chest areas, respectively. Each of these devices

will have a miniaturized embedded system that receives, processes and stores its data until it can be wirelessly transmitted to a smartphone or computer.

3.1.1 Epidermal Patch Design

Through collaboration with Dr. Lara Wittine, a sleep study specialist with USF Health, relevant sensors were condensed into facial, chest and neck regions to design into the patches, keeping form-factor in mind to minimize obstruction to sleep habits. Using an elastomer backing that can provide high flexibility and stretchability, Ag/AgCl electrodes with an electrolytic hydrogel will be fabricated into the three adhesive patches as shown in Figure 3.2. Using this configuration, the electrodes would be able to retain a high signal-to-noise ratio and can be accurately placed into the required locations by either the sleep technician or patient. By designing a clamp into the electronics enclosure, this device will allow for a disposable patch with reusable electronics, minimizing costs of additional equipment between studies.

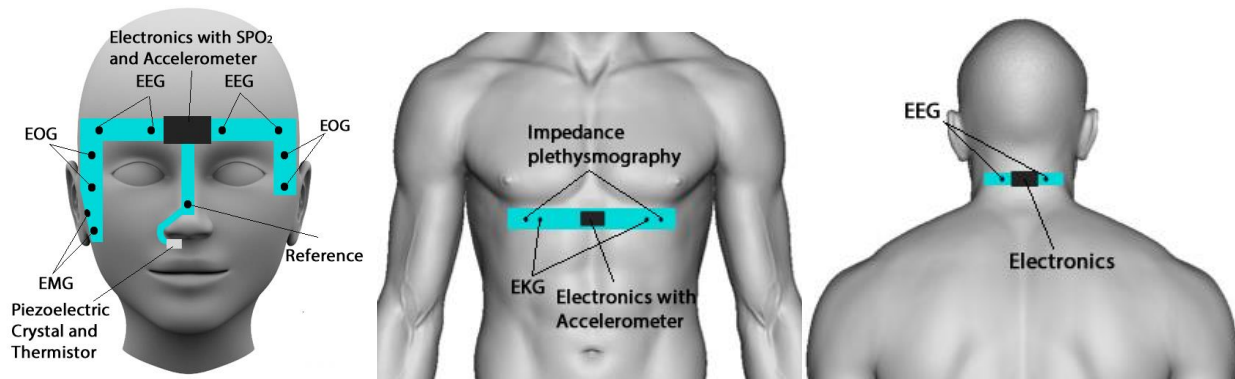


Figure 3.2: Elastomer patch locations and sensors. 3 patches placed on the face, chest and nape.

- *EEG (facial)* – 2 channels of electroencephalography (EEG) for reading neural activity. Electrodes will be placed in the F7, Fp1, Fp2 and F8 locations, based on the international 10-20 system.
- *EEG (occipital)* – 1 channel of EEG recorded from 2 electrodes placed symmetrically along nape of the neck for detection of alpha waves. As alpha waves are the strongest

signals of the brain, it is still possible to record these signals from just below the hairline, allowing for the elimination of electrodes in the hair [14].

- *EOG* – 2 channels of electrooculography from 4 electrodes placed symmetrically along the lateral edges of the eyes. This measures eye movement associated with sleep patterns.
- *EMG* – 1 channel of electromyography (EMG) from 2 electrodes placed along the masseter muscle for measuring its activity. This can be used to detect the onset of sleep.
- *EKG* – 1 channel of electrocardiography (EKG) from 2 electrodes placed on the chest for measuring electrical activity of the heart.
- *Nasal Pressure*: Piezoelectric crystal placed just below the nostril to detect respiratory flow through changes in pressure. As this method is sensitive to changes in airflow, it is frequently used for detection of hypopnea [15].
- *Nasal Thermistor*: Thermistor placed just below the nostril to detect respiratory flow through changes in temperature. As this method is roughly quantitative of air flow, it's often used for detection of apneas [15].
- *Position Sensor (head)*: Accelerometer included in the facial electronics module to track orientation of the head.
- *Position Sensor (torso)*: Accelerometer included in the chest electronics module to track orientation of the torso.
- *O₂ Saturation (SPO₂)*: Reflective photoplethysmography using red and infrared LEDs with photodiode to determine the blood oxygen saturation level in the forehead.
- *Respiratory Effort*: Impedance plethysmography using changes in chest resistance can be used to track chest expansion for determining respiratory effort.

- *Reference:* Due to its low electrophysiological activity, an electrode on the nose is used as the reference and right leg driver for the EEG, EOG, EMG and EKG signals.

3.1.2 Electronics Module Design

In order to provide the required functionality in a low-profile package, the electronics will be designed into modules with interconnecting headers to allow stacking. This stack will include a small rechargeable lithium ion battery at the bottom, placed just above the SPO₂ sensor in the facial electronics module, and PCBs stacked above it for providing the signal amplification, filtering, data storage and transmission. To provide additional noise isolation, these electronics will be enclosed by an aluminum chassis to provide a faraday cage for the device. A rubber silicone layer will be placed on top of this to improve comfort to the patient and to electrically insulate the chassis.

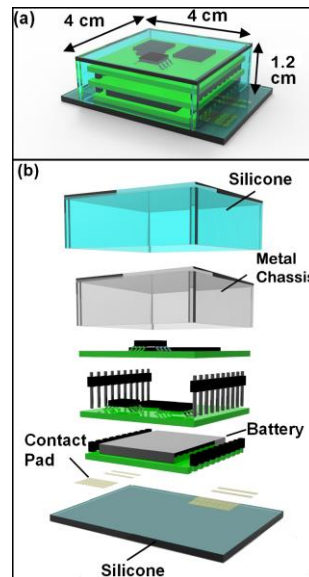


Figure 3.3: Proposed wearable sleep sensor device electronics module. Shown in (a) assembled view and (b) exploded view.

Each of these modules will contain a dedicated embedded system working independently of each other and will be Bluetooth capable and self-powered. Rather than using discrete cascaded op-amps for amplification and filtering of the electrophysiological signals as

initially used in the eye-tracker design, an analog front end (AFE) will be included in each device to simplify the data acquisition process and save real estate on the boards. This data will be received by a digital signal processor (DSP) for signal conditioning and stored in an embedded Multi-Media Controller (eMMC). In order to optimize power consumption while allowing sleep technicians to monitor the patient's condition, the Bluetooth is enabled every 2 hours for data transmission to a cell phone or computer.

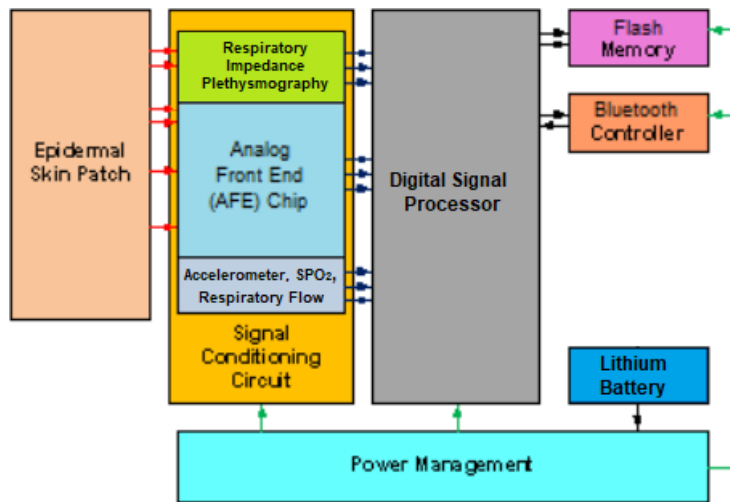


Figure 3.4: Block diagram of proposed wireless sleep sensor electronics module.

Once received by the cell phone or computer, an algorithmic will be able to analyze the data for indications of sleep stages and variables to help improve the accuracy and speed of sleep disorder diagnostics by the sleep physicians.

In this chapter, a non-miniaturized prototype system is presented for verification of sensor design and validating signal processing techniques. While not yet performed as of this writing, clinical trials will be conducted using this device to validate its operation in comparison to the gold-standard polysomnography (PSG) devices.

3.2 Device Design

In this first stage of the design, a prototype system was designed and tested using similar components to the ones required in the patch modules. Using a centralized electronics module mounted in the supraclavicular region, wires were used to connect the sensors, as shown in Figure 3.5. In order to provide access to the SPO₂, respiratory rate and position sensors, two small additional PCBs were designed for mounting in the required locations on the face.

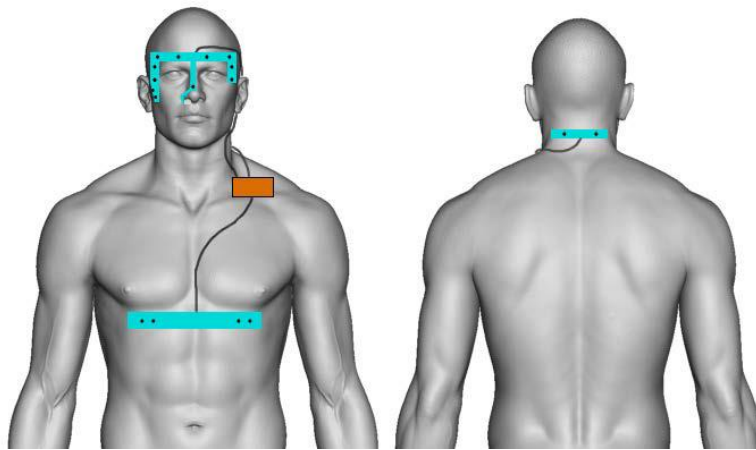


Figure 3.5: Sleep sensor prototype testing configuration. Includes wires running from facial, chest, and nape patches to a centralized electronics module mounted in the supraclavicular region.

The primary circuit was designed into a 10 x 6.5 cm PCB as shown in Figure 3.6. This device uses a TMS320C6745 DSP chip and stores the data to an eMMC. The right side of the board includes the AFE and impedance plethysmograph and the bottom edge has header connections for the external sensor interfaces. The top-left corner includes the battery charger and DC-DC converters for operation of the device.

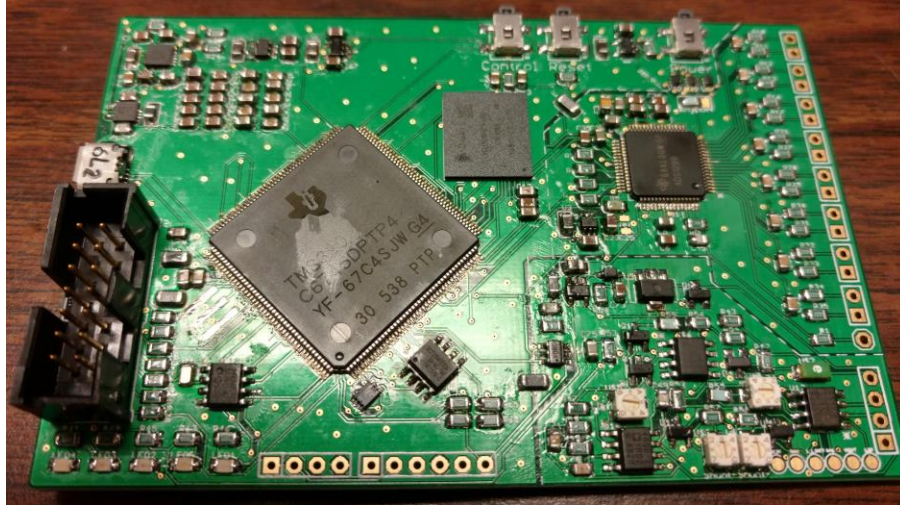


Figure 3.6: Digital photograph of wireless wearable sleep sensor PCB. The board was fabricated by Advanced Circuits and populated in-house. Board dimension 10 by 6.5 cm.

In order to allow for a full power-down of the device to maximize battery lifetime when not in use, an additional power supply was added for control of a digital logic circuit that can be controlled by three inputs: power button state, current device state and an input from the DSP. Together, these can be used to enable or disable the primary power supplies. This logic circuit can be described using the equation shown in the truth table in Figure 3.7 where the output is the enable pin of the supplies, A is the button input, B is the current power state of the device and C is an input GPIO (general purpose input-output) from the DSP. This will allow the device to turn off only when all three inputs are high and turn back on only when the button is pulled high. The DSP will include an interrupt for the power button to safely stop any processes before allowing the device to shut down.

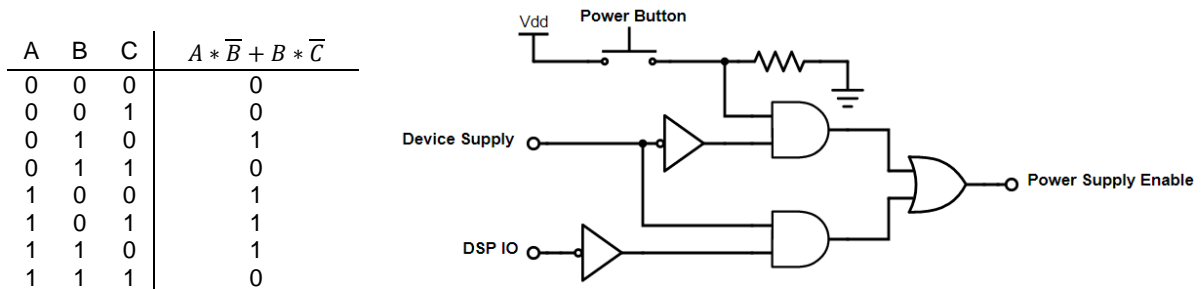


Figure 3.7: Device power logic truth table (left) and circuit (right). A is the button input, B is the device power state, and C is an IO from the DSP. This is used to power down the power supplies to conserve battery power when in standby mode.

3.2.1 Analog Front End

Using the same solution as the eye-tracker described in Chapter 2, an ADS1299 was also selected for the prototype board due to its 8 channels of input and high ADC resolution. However, in order to maximize eMMC storage while providing the required standard of data, a sampling rate of 500 SPS was used for the EEG, EMG, EOG and EKG as recommended by the American Association of Sleep Technologists (AAST) [15]. Each channel was given the maximum gain of 24 V/V and the right leg driver enabled for the positive channels. As each patch ultimately will not need 8 channels of input for the AFE (analog front end), alternative components such as the ADS1299-4 or ADS1294 can be used to reduce cost and power requirements. Configuration register values can be found in Table 3.1.

Table 3.1: AFE configuration register values for wearable sleep sensor.

Register	Address	Value	Function
CONFIG1	01h	95h	Turns off daisy chain mode, disables clock output and sets sampling rate to 500 SPS
CONFIG2	02h	C0h	Configures test signal generation
CONFIG3	03h	ECh	Use 5V/2 as signal reference and enable bias for right-leg-driver
LOFF	04h	00h	Disable lead-off detection
CH[1:7]SET	05h	60h	Set up channels 1 -7 as normal electrode inputs with 24 V/V of gain
CH8SET	07h	81h	Disable channel 8 by powering down
BIAS_SENSP	0Dh	7Fh	Enable bias sense for positive channels 1-7
BIAS_SENSN	0Eh	00h	Disable bias sense for all negative channels
LOFF_SENSP	0Fh	00h	Disable lead-off detection
LOFF_SENSN	10h	00h	Disable lead-off detection
LOFF_FLIP	11h	00h	Keep lead-off bias non-flipped
GPIO	14h	00h	Keep GPIO as output low
MISC1	15h	00h	Keep stimulus reference and bias switch open
CONFIG4	17h	00h	Continuous conversion mode with lead-off comparators off

3.2.2 Respiratory Impedance Plethysmography

In patients undergoing sleep studies, both respiratory flow and effort are tracked to help distinguish between obstructions in the airway with pauses in breathing efforts. In conventional polysomnography devices, respiratory effort is typically measured using inductive or piezoelectric sensors in belts wrapped around the chest or abdomen. However, since these sensors require straps wrapped around the entire torso, an alternative method was selected.

Instead, by injecting a harmless amount of alternating current through the chest, measured changes in thoracic impedance can be used to track the expansion of the chest [16]. This method, called impedance plethysmography, can use either two or four electrodes placed on the chest. By driving an AC signal with a constant current amplitude of up to 1mA at 100 kHz, measured voltage changes across the electrodes can be used to track the changes in resistance. This current was selected due to its amplitude remaining well below the threshold of human perception and any physiological harm [17]. This method allows for long-term respiratory monitoring and requires minimal adjustments with changes in body position.

In order to implement this sensor, a 100 kHz frequency was first generated. By using a Wein bridge oscillator as shown in Figure 3.6, the values of the resistors and capacitors can be used to tune the circuit for the required frequency using equation (3.1). As the complex impedance of the RC circuits provide a $0.33V_{out}$ positive feedback, this must also be matched by a voltage divider for the negative feedback using equation (3.2) to allow for the resonance. As this resistor ratio must be within a very tight range, a trimmer potentiometer was used for R_1 to allow for tuning of this circuit.

$$f = \frac{1}{2\pi RC} \quad (3.1)$$

$$R_2 = \frac{R_1}{2} \quad (3.2)$$

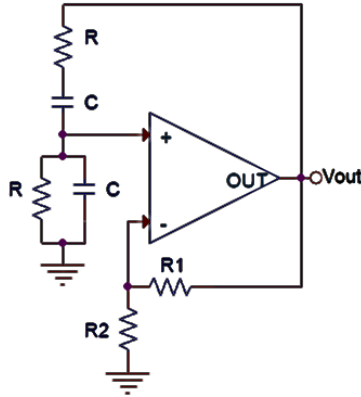


Figure 3.8: Wein bridge oscillator configuration. This was used to generate a 100 kHz signal for the system.

Using values of $R = 1.6\text{k}\Omega$, $C = 1\text{nF}$, $R_1=5\text{k}\Omega$ and $R_2=2.5\text{k}\Omega$, this circuit was simulated using PSpice. Since this oscillation is normally initiated by noise from the surrounding environment, an initial condition of 1V on the parallel capacitor was used to initiate the oscillation. By gradually increasing the R_1 value, a 100 kHz frequency was measured as shown in Figure 3.9.

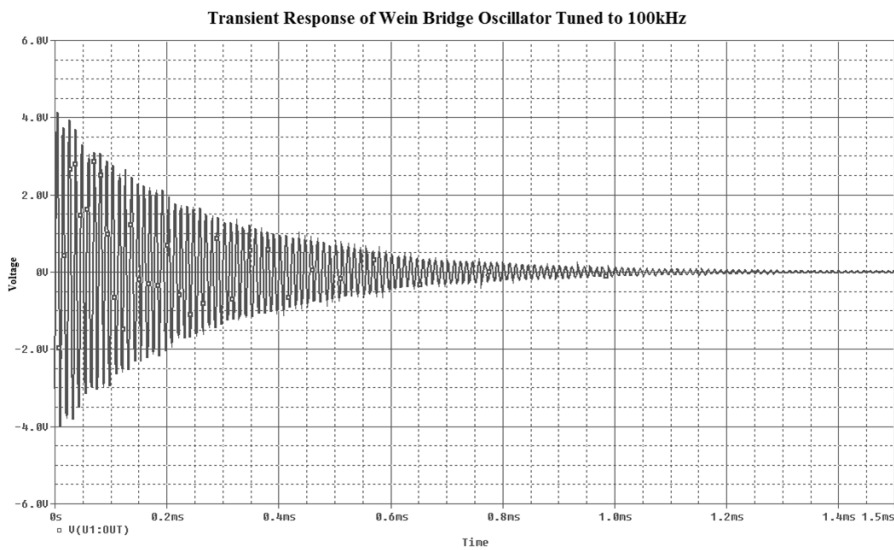


Figure 3.9: PSpice transient response of Wein bridge oscillator. Tuned to 100 kHz with $R_1=5.03\text{k}\Omega$ and $R_2=5\text{k}\Omega$ (R_1 just above point of being undamped ($\zeta=0$)).

This simulation was then repeated for the frequency response using a 1mV AC source placed in the positive feedback loop. By sweeping its frequency from 1Hz to 1MHz, the response of the circuit could be observed.

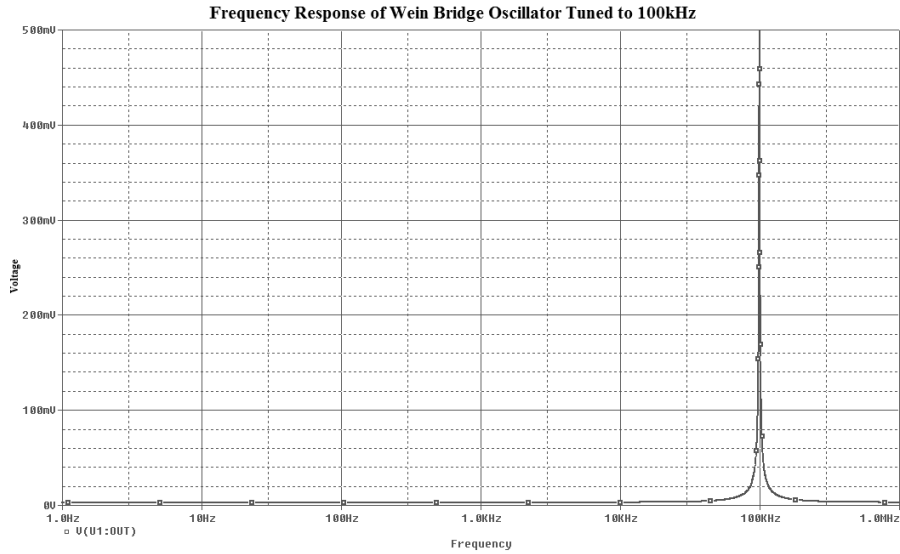


Figure 3.10: PSpice frequency response of Wein bridge oscillator. Tuned to 100 kHz with $R_1=5.03k\Omega$ and $R_2=5k\Omega$.

In order for this signal to be used in impedance plethysmography, this AC signal of constant voltage amplitude needed to be converted to a constant current amplitude. This was done using the voltage-to-current circuit as shown in Figure 3.11. By using equations (3.3) and (3.4), the current could be configured to provide an amplitude of 1 mA through the chest (R_L) using a V_{in} amplitude of 12 V and an R_2 value of 10 k Ω .

$$V_{out} = \frac{R_L + R_2}{R_2} V_{in} \quad (3.3)$$

$$I_{out} = \frac{V_{out}}{R_L + R_2} = \frac{V_{in}}{R_2} \quad (3.4)$$

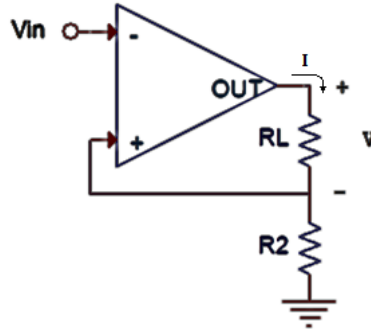


Figure 3.11: Voltage-to-current converter circuit. Provides constant current passing through R_L .

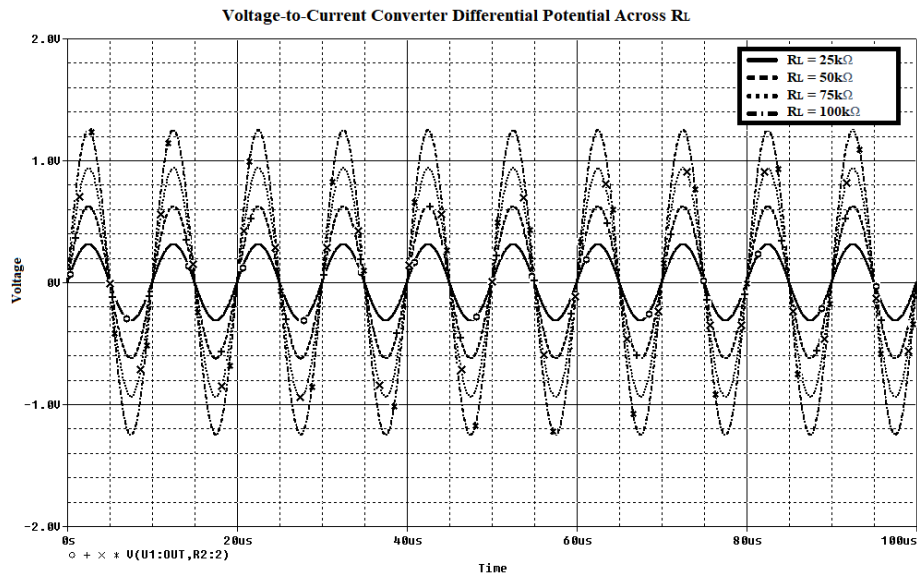


Figure 3.12: PSpice voltage-to-current converter voltage simulation through R_L . $R_L = 25k\Omega$, $50k\Omega$, $75k\Omega$ and $100k\Omega$ shown. Note the increase in voltage with higher resistance

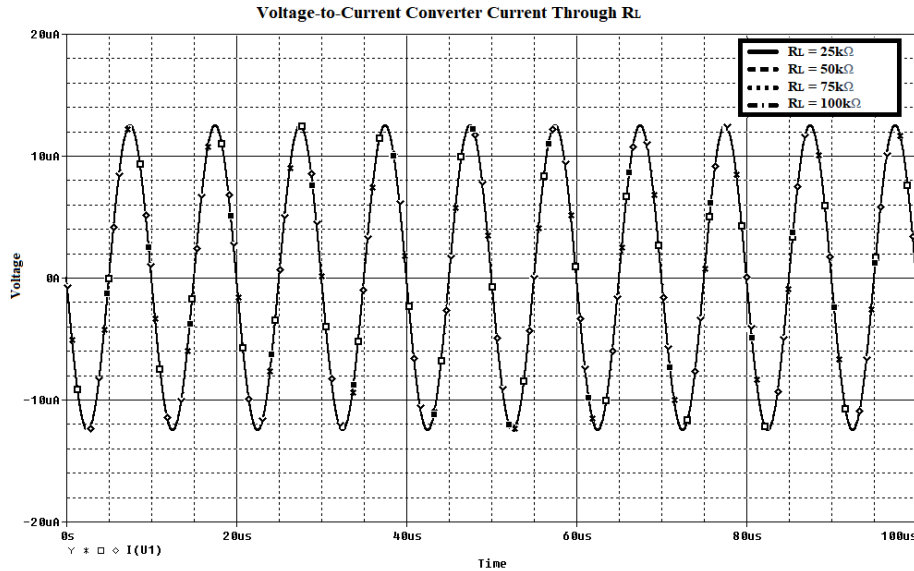


Figure 3.13: PSpice voltage-to-current converter current simulation through R_L . $R_L = 25k\Omega$, $50k\Omega$, $75k\Omega$ and $100k\Omega$ shown. As all resistor values provide identical current responses, all curves are superimposed on each other.

Using the instrumentation amplifier configuration as shown in Chapter 2, the differential voltage across R_L was measured to determine the respiratory rate. Since the resistance of the chest is changing as it expands, voltage amplitudes of the 100 kHz signal also change. To remove this carrier frequency (f_c) from the signal, a precision envelope detector was used as a peak detector. Using the precision half wave rectifier configuration, a capacitor and resistor are placed between the output and ground for filtering the signal as shown in Figure 3.12. This circuit removes the negative components from the signal and connects the signal peaks, as shown in the PSpice simulation results in Figure 3.15. Since the RC time constant ($\tau=RC$) needs to be between $\frac{1}{f_c}$ and $\frac{1}{f_m}$, where f_m is the highest modulation frequency, a time constant of 10ms was used. The resulting output signal can be acquired by an ADC for further processing.

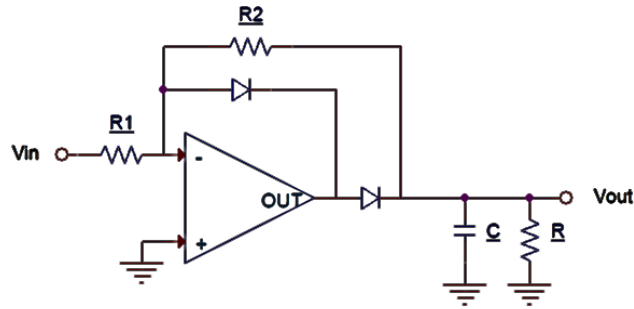


Figure 3.14: Precision envelope detector circuit configuration. This was used to strip off the 100 kHz oscillator frequency from the signal.

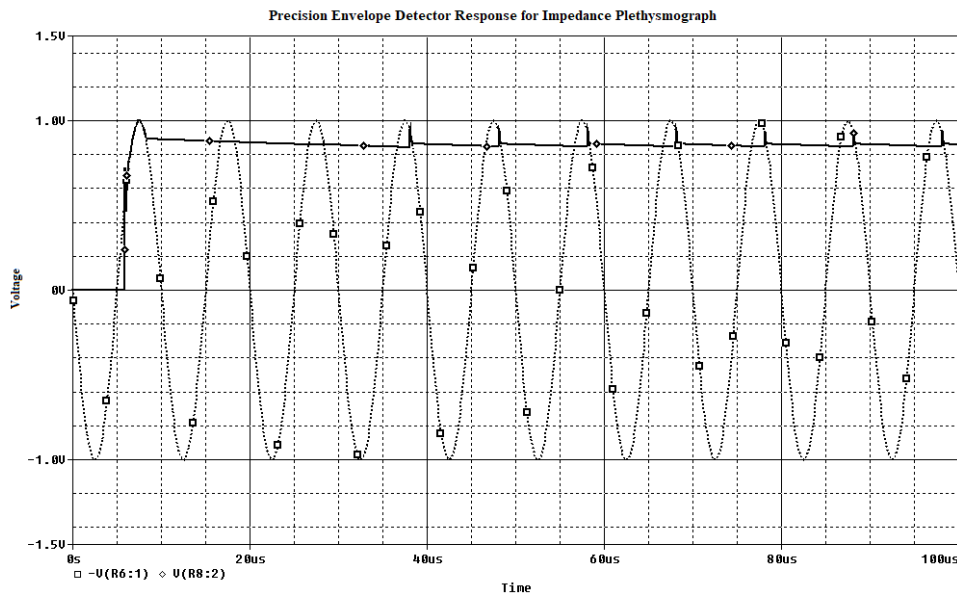


Figure 3.15: Transient response PSpice simulation of precision envelope detector. Shows response when $f_c = 100$ kHz and $\tau = 10$ ms. The dotted line is the carrier frequency and the solid line is the output of the peak detector.

Due to the varying conductive properties of the body, a ± 12 V supply was used to allow for operation with high-impedances through the chest. However, additional precautions were taken to ensure excessive currents cannot be driven through the patient. Adjustable Zener shunts were placed on the output of the voltage-to-current converter to allow for configurable clipping of the signal at voltages lower than 12V. In addition, a 50mA fuse was placed in the line to break the circuit should there be an electrical fault in the driver and a ferrite-isolated ground was shared with the AFE.

3.2.3 Piezoelectric Transducer with Temperature Detection

In contrast to respiratory effort with the detection of chest expansion, respiratory flow is also needed to determine how much air is making its way into the lungs. This is detected using a piezoelectric crystal placed just beneath the nostril. By allowing the patient to breathe onto the device, the increase in pressure and/or temperature change can be correlated to a respiratory flow rate. This PCB measures 0.9 x 0.85 cm and includes a TE MS5805 sensor that can measure pressure and temperature using a dedicated 24-bit ADC with an I²C interface.

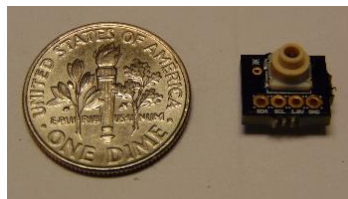


Figure 3.16: Digital photograph of respiratory flow sensor board. Includes pressure and temperature detection. The board was fabricated by OSH PARK and populated in-house. Board dimension 0.9 by 0.85 cm

3.2.4 Pulse Oximeter and Inertial Measurement Unit

An additional 1.6 x 1.1 cm board was also developed for the detection of head movement and saturation of peripheral oxygen levels. This board includes a MAX30102 SPO₂ sensor that detects the reflectance of red and infrared light through the forehead to calculate the blood oxygen level. On the reverse, a Bosch BMI160 IMU was included for the detection of head position. Both of these devices communicate with the DSP using the I²C interface.

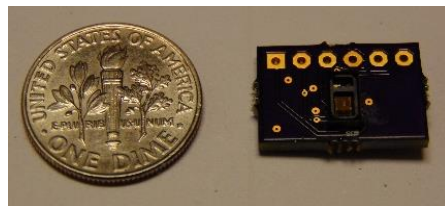


Figure 3.17: Digital photograph of facial sensor. Includes SPO₂ sensor and IMU. PCB was fabricated by OSH PARK and populated in-house. Board dimension 1.6 by 1.1 cm.

3.2.5 Digital Signal Processing and Data Storage

A TMS320C6745 DSP chip was selected for the system due to its 30 MHz clock, floating-point operation ability and low power consumption. In addition, while the footprint of this DSP was too large for the patch electronic modules, an alternative 1.5 x 1.5 cm BGA package is available which can allow for simple transition of the system into a miniaturized package. This device was configured to boot from an SPI-interfaced Microchip SST25VF512 flash chip to allow for an additional 512 kB of program space and an 8 GB Toshiba NAND eMMC chip for data storage. In addition, a Microchip MCP7940N real-time clock (RTC) chip was included in the design to allow for a real-time clock value to be included in the data logs.

Due to the large variety of sensors and additional frequency components found in each of the signals, the DSP was programmed to use infinite impulse response (IIR) filters for isolating the required bandwidth. Using the SPTool included in MATLAB, the parameters found in Table 3.2 can be used to generate the required coefficients for the digital filtering. These filter and sampling parameters are based on the AAST's polysomnography standards [15].

Table 3.2: Wearable sleep sensor filtering and sampling requirements

Channel(s)	Low Frequency Cutoff	High Frequency Cutoff	Sampling Rate
Facial EEG	0.3 Hz	35 Hz	500 SPS
Occipital EEG	0.3 Hz	35 Hz	500 SPS
EOG	0.3 Hz	35 Hz	500 SPS
Masseter EMG	0.3 Hz	10 Hz	500 SPS
EKG	0.3 Hz	70 Hz	500 SPS
Impedance Plethysmography	0.1 Hz	15 Hz	100 SPS
Nasal Piezoelectric	0.1 Hz	15 Hz	100 SPS
Nasal Thermistor	0.1 Hz	15 Hz	100 SPS
Facial Position	-	-	1 SPS
Torso Position	-	-	1 SPS
SPO ₂	-	-	25 SPS

While the wireless Bluetooth interface was not included in this iteration of the design, data will be accessible through the USB 2.0 interface. As each 12-hour session will require

about 250 MB of data to be stored, the 8 GB of eMMC non-volatile memory will allow for up to twenty-four sessions before memory clearing is required.

3.3 Device Testing

In testing this device, with only the use of the right leg driver for filtering, the EOG and EKG channels were found to have an SNR of approximately 20, EMG channel at 12, and EEG channels at approximately 2. While the IIR filters are not yet implemented in the DSP program as of this writing, the recorded data was filtered using the parameters outlined in Table 3.2 through signal processing functions in MATLAB. The resulting signals can be found in Figures 3.18 through 3.22.

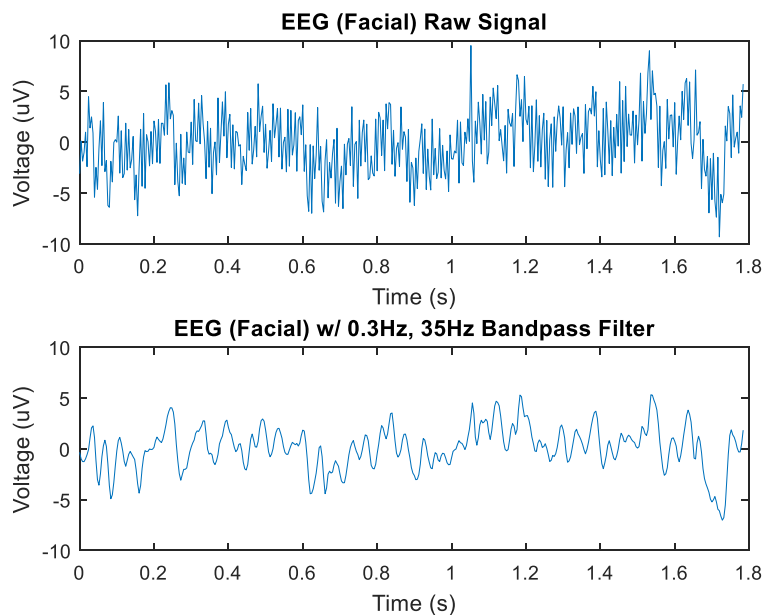


Figure 3.18: EEG signal obtained by electrodes placed in the F7 and Fp1 locations. Raw signal (top) in comparison to signal with 0.3Hz - 35Hz band-pass Butterworth filter (bottom), obtained through signal processing in MATLAB.

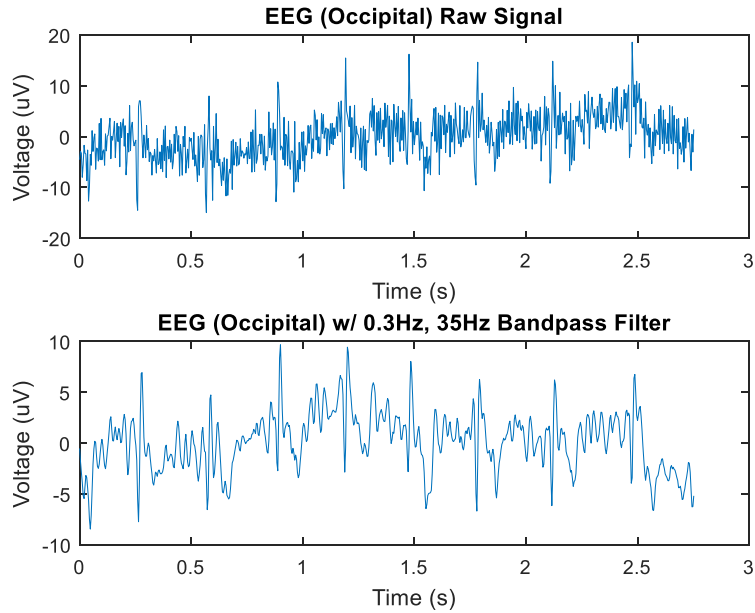


Figure 3.19: EEG signal obtained by electrodes placed on the nape of the neck. Raw signal (top) in comparison to signal with 0.3Hz - 35Hz band-pass Butterworth filter (bottom), obtained through signal processing in MATLAB.

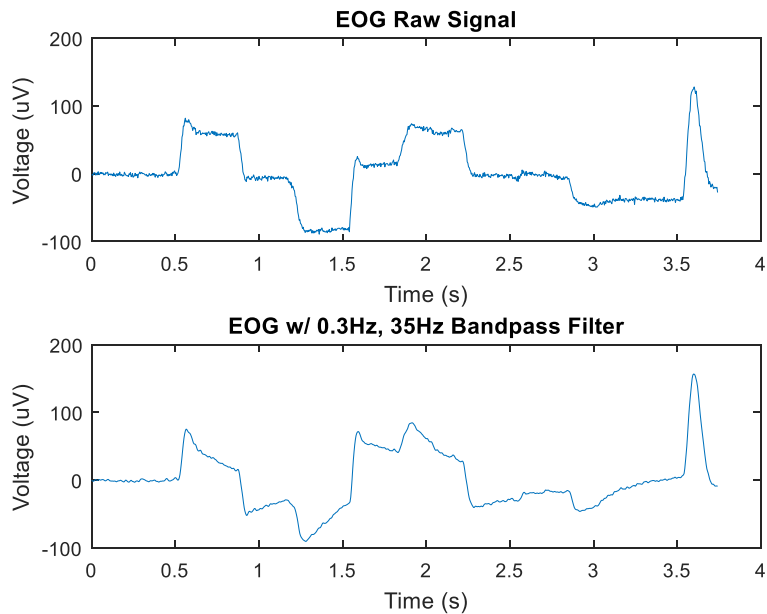


Figure 3.20: EOG signal obtained by electrodes placed along the lateral edge of the eye. Raw signal (top) in comparison to signal with 0.3Hz - 35Hz band-pass Butterworth filter (bottom), obtained through signal processing in MATLAB.

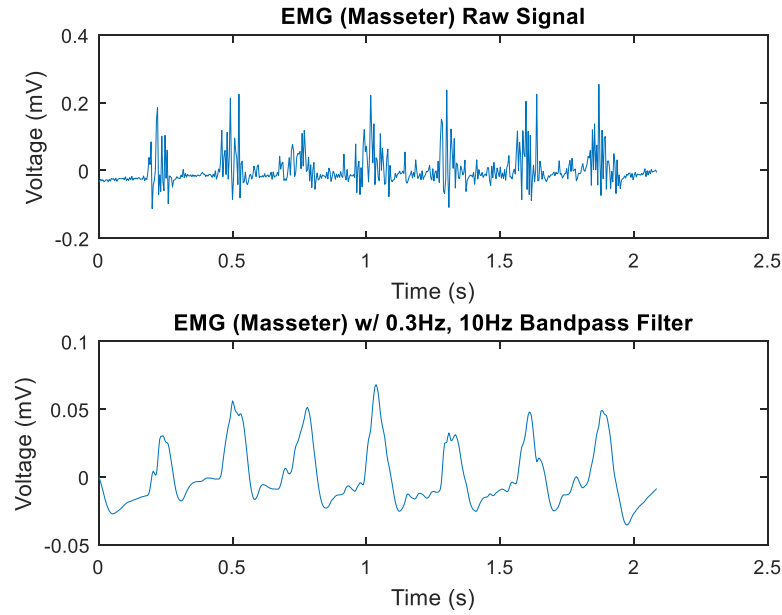


Figure 3.21: EMG signal obtained by electrodes placed on the masseter muscle of the face. Raw signal (top) in comparison to signal with 0.3Hz - 10Hz band-pass Butterworth filter (bottom), obtained through signal processing in MATLAB.

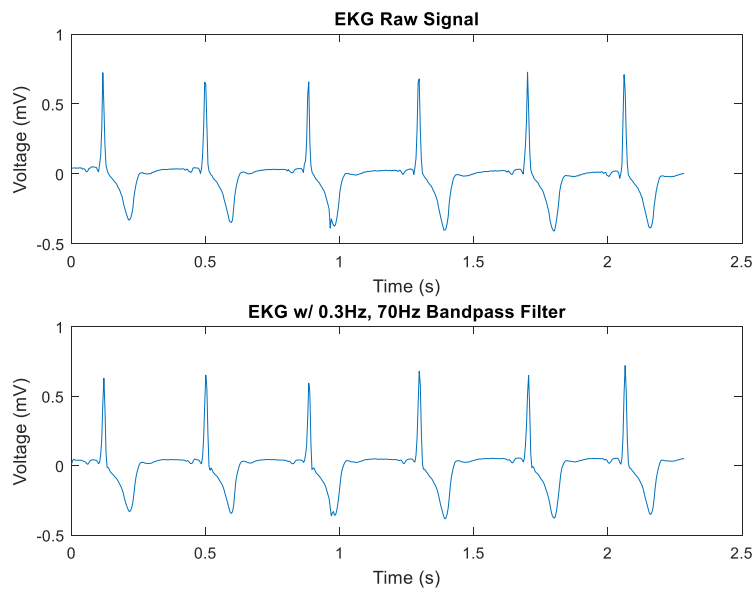


Figure 3.22: EKG signal obtained by electrodes placed on the chest. Raw signal (top) in comparison to signal with 0.3Hz - 70Hz band-pass Butterworth filter (bottom), obtained through signal processing in MATLAB.

In addition, the peripheral devices for measurement of respiratory flow, blood oxygen saturation and position were found to perform to the device standards outlined in their respective datasheets [18, 19]. As can be observed in Figures 3.23 and 3.24, changes in pressure and temperature could be correlated to respiration as the user inhaled and exhaled on the transducer. Also providing an SNR of approximately 20, these signals were filtered in MATLAB to produce the signals shown. While the temperature sensor was prone to rapid changes in temperature due to the small size of the device, the DC rejection of the bandpass filter was able to extract the required data.

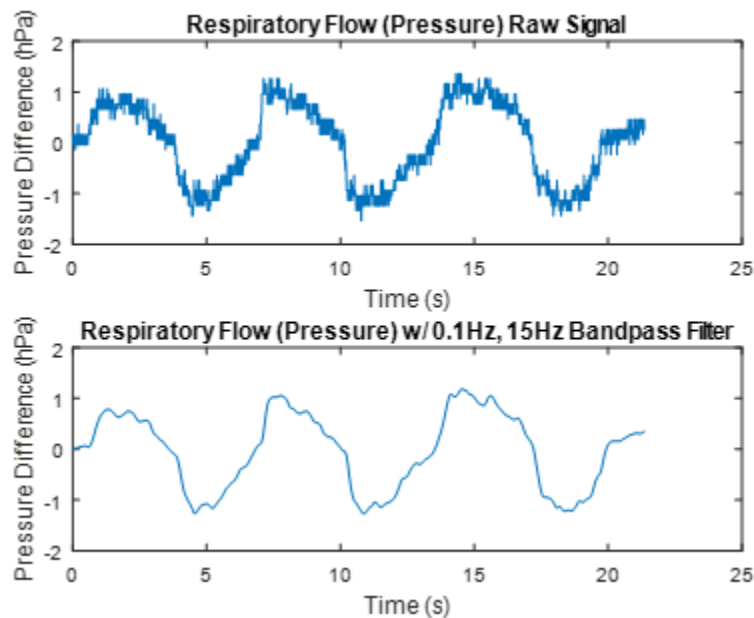


Figure 3.23: Respiratory flow signal obtained through piezoelectric pressure transducer. Placed just beneath the nostril. Raw signal (top) in comparison to signal with 0.1Hz - 15Hz band-pass Butterworth filter (bottom), obtained through signal processing in MATLAB.

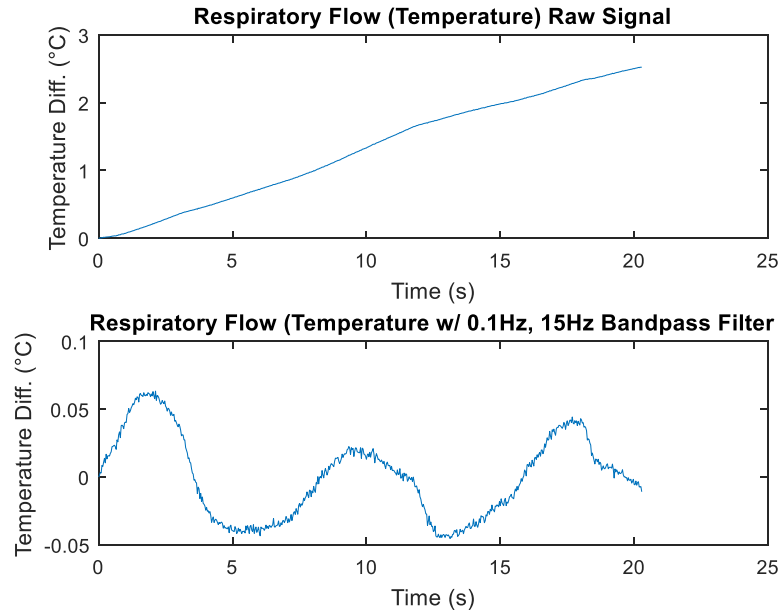


Figure 3.24: Respiratory flow signal obtained through thermistor. Placed just beneath the nostril (embedded into piezoelectric transducer). Raw signal (top) in comparison to signal with 0.1Hz - 15Hz band-pass Butterworth filter (bottom), obtained through signal processing in MATLAB.

When the impedance plethysmograph was tested, the Wein bridge oscillator was found to oscillate at approximately 75 kHz. While likely caused by the resistor and capacitor tolerance values, this frequency was still within the acceptable 20-100 kHz range for this application [11]. However, as the chest impedance measured to be a higher value than originally anticipated, a greater supply voltage and lower R_2 value (Figure 3.11) will be needed to provide the required measurements.

3.4 Summary

In this chapter, an initial prototype of a wireless sleep study system was proposed. Since conventional polysomnography (PSG) devices require dozens of electrodes placed around the body, patients are often uncomfortable due to their limited movement, preventing them from falling asleep. By integrating miniaturized wireless electronic modules onto elastomer patches mounted on the forehead, chest and nape, similar sensors can be used to provide a more

comfortable experience for patients undergoing sleep studies, resulting in higher quality data. As this first stage of development involved the design of a single device for comparison to gold standard devices used in clinical trials, the initially recorded raw data proved to retain a high signal-to-noise ratio. By performing additional filtering of the signals in the DSP, in addition to the reduced length of electrode leads for less noise, we believe that this device will be able to provide data comparable to conventional polysomnography devices. This hypothesis will be tested in the near future and is discussed in Chapter 5 in the future work section.

CHAPTER 4 : TRAUMA-DETECTING PERSONAL LOCATOR BEACON

4.1 Device Overview

With the increasing use of personal locator beacon (PLB) devices around the world, thousands of people have been able to receive emergency care despite being in remote areas without any local emergency services available. However, many hikers have lost their lives due to trauma preventing them from activating their devices.

While many forms of trauma could result in life-threatening conditions, this device would monitor physiological conditions that would estimate when conditions most likely exist that could indicate major trauma or other medical emergencies. In this concept design, sensors will be used for detecting blood oxygen concentration, pulse rate, sweat, skin temperature and blunt impact. When trauma is determined to have occurred, the device first activates a quiet alarm that is audible to the person wearing the device and people within the immediate vicinity. After 2 minutes, this alarm increases in volume for another 8 minutes to alert anyone that may be nearby before activating the distress signal. This 10 minutes of alarms allows for the user or bystanders to cancel the activation in case of a false positive or if the severity of the incident doesn't require intervention of emergency services.

4.2 Device Design

In normal use, PLB devices used by hikers are often carried either on the strap of their backpack or in a quick-access pocket. However, due to the sensors required in this device, skin contact must also be maintained. In order to remain unobtrusive to the user's activities while providing quick access, this device will be mounted on the wrist. As these users often go on multi-day hikes without access to electricity, this device must also be able to operate for the

duration of their hike and not prevent access to emergency services should the battery die. In addition, the rough terrain of the hikes will require ruggedization of the device and resistance to moisture and temperature.

In order for this to be accomplished, the system will use two batteries: one rechargeable battery for operation of the biophysiological sensors and one non-rechargeable battery dedicated to operation of the PLB. Two buttons will also be included in the design: an easy-access button for disabling the alarm and a protected button for manual activation of the PLB. Each of the sensors will transmit their data to a low-power microcontroller. A block diagram of the system can be found in Figure 4.1.

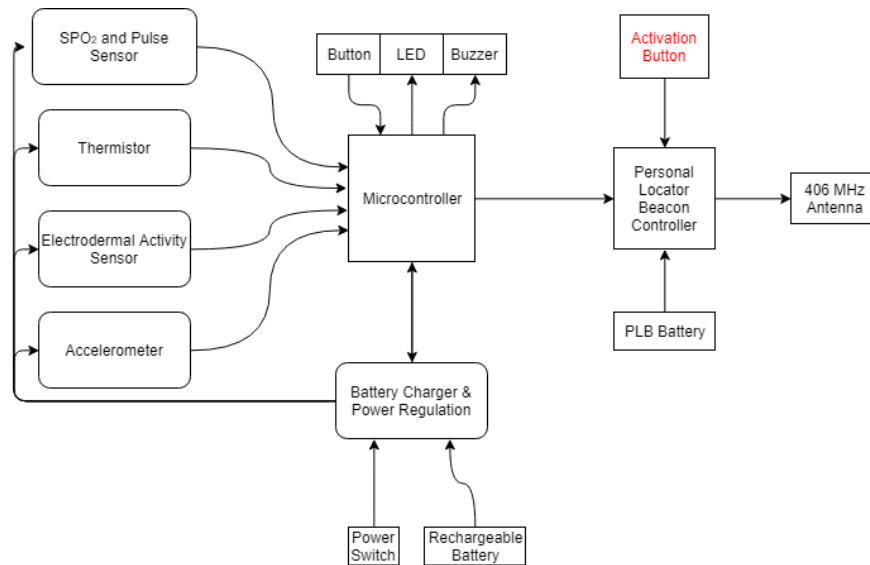


Figure 4.1: Trauma-activated PLB block diagram. Design includes dedicated battery to power the PLB to ensure sufficient power to drive the 406 MHz distress signal antenna.

4.2.1 SPO₂ and Pulse

As common physiological indicators of unconsciousness often includes lowered heart rate or hypoxia, an SPO₂ sensor can be used for detecting blood oxygen saturation levels and pulse rate. By measuring the blood's absorption of red and infrared light at around 660 nm and 940 nm, respectively, deoxygenated hemoglobin is found to absorb more infrared than red light and vice-versa for oxygenated hemoglobin. This trend can be used to calculate the approximate

blood oxygen saturation level and fluctuations in the signal can be used to calculate pulse rate. However, conventional transcutaneous transmittance oximeters require placement on thin portions of the body such as the fingers, toes, earlobes or nose to work effectively, which is not compatible with this device. Alternatively, reflectance oximeters are often used in locations such as the forehead by using optical reflection from the cranium and subcutaneous tissue, but are not normally effective for use on the wrist due to its sensitivity to slight movement and small area of effective use just above the radial artery [20].

In order to allow the reflectance method to work for this wristband, a soft concave support structure can be used to hold the sensor in place along the contour of the inner side of the arm as demonstrated by Pang and Ma [21]. By embedding infrared and red LEDs into this contour structure and aiming the light through the skin towards a photodiode, the resulting signal can be used to determine oxygen saturation of the blood. Rather than using discrete components for amplification and filtering of the signal, a dedicated AFE (e.g. TI AFE4400) can be used and enabled only when a measurement is required.

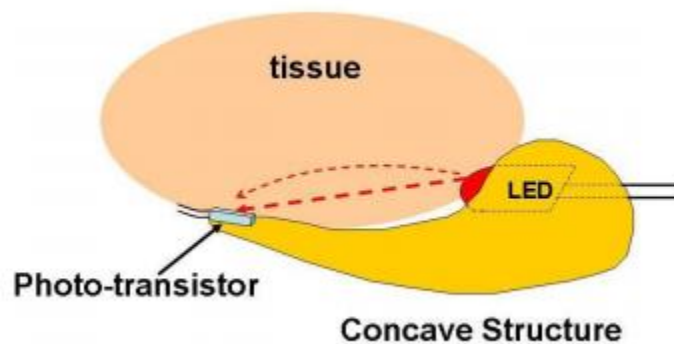


Figure 4.2: Pulse oximeter concave support structure. Used for placement of LEDs and photodiode along contour of wrist [21] © 2014 IEEE.

4.2.2 Electrodermal Activity

While sweating itself does not indicate medical emergency, measurement of the electrodermal activity can be used with other sensors to help indicate conditions such as severe

dehydration and can help indicate signs of distress. Since electrodermal activity can be measured by finding the conductance of the skin, a Wheatstone bridge configuration can be used to accurately convert the varying resistance into a measurable voltage. This can be done by placing two Ag/AgCl electrodes approximately 3 cm apart and connecting them in place of R_x , as shown in Figure 4.3. Using formula (4.1), the voltage found across V_G can be used to calculate the resistance of the skin. This voltage is measured using an instrumentation amplifier as outlined in chapter 2.

$$V_G = \left(\frac{R_2}{R_1 + R_2} - \frac{R_x}{R_x + R_3} \right) \quad (4.1)$$

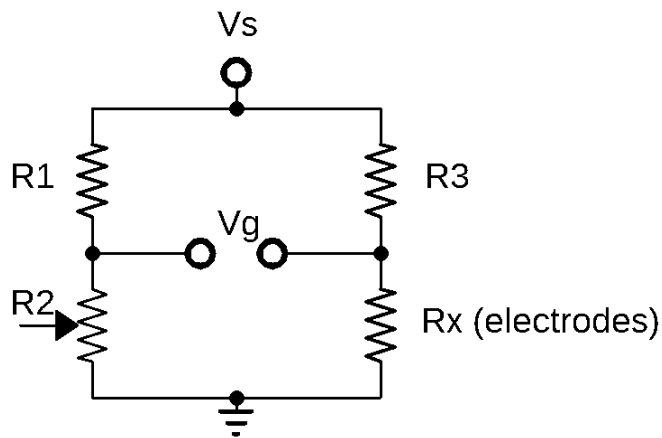


Figure 4.3: Wheatstone bridge configuration. R_x is the Ag/AgCl electrodes connected to the skin. R_2 is used to adjust the starting voltage measured across V_G .

4.2.3 Freelfall and Impact Detection

As one of the more common causes of injury causing search and rescue efforts, falls are a common concern when exploring remote areas with large drop-offs [22]. In order to detect these events, an accelerometer can be used to detect free-falls and impacts. Since this sensor must be run continuously during use, ultra-low power consumption is required for this component. Using components such as the ST LIS3DH, free-fall detection interrupts can be used to wake the microcontroller when G-forces have fallen below a threshold value. If followed by a sudden spike of G-forces, the PLB alarm sequence can be initiated.

4.2.4 Skin Temperature

Whether due to falling through the ice or overexerting yourself on a hot day, significant deviations of body temperature can be disorienting and dangerous if not treated quickly. In order to help detect this, a thermistor can be placed against the skin to estimate core body temperature. By connecting the thermistor to a Wheatstone bridge and differential op-amp as described previously, fluctuations in temperature can be converted into a voltage value that can be read by an ADC.

4.2.5 Microcontroller and Algorithms

Controlling the interface between the sensors and PLB, an ultra-low power microcontroller (e.g. PIC16LF1509) can be used to periodically check sensor conditions against a list of criteria for hazardous conditions.

Remaining in sleep mode between sampling sessions, the microcontroller will read data for five seconds before analyzing it, comparing it to trends over the last 30 minutes, and going back to sleep for another minute. This will allow for optimal power consumption while actively monitoring the user for hazardous conditions. If the analysis determines the user may be in distress, an alarm sequence is initialized. As the quiet alarm is triggered, sensor sampling is increased to once every 15 seconds to check if conditions have returned to normal. To save power, this is reduced to once every 30 seconds if the loud alarm is initiated. A basic flow chart of program flow can be found in Figure 4.4.

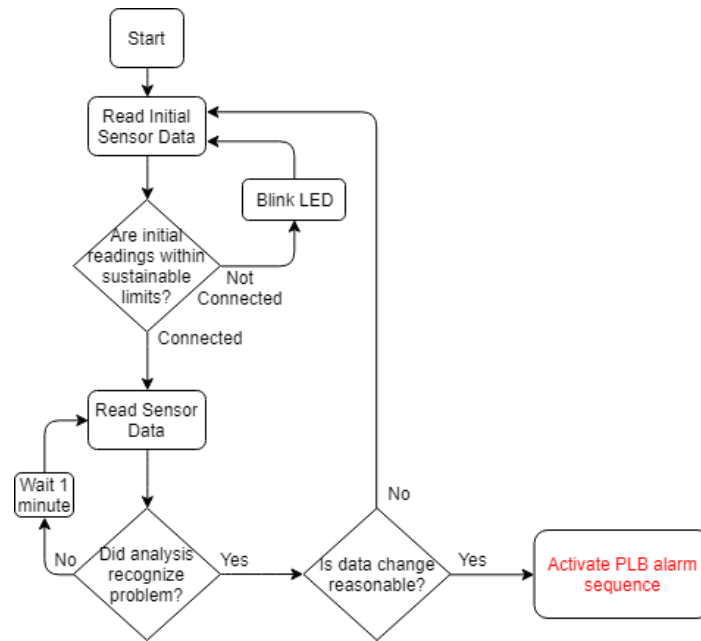


Figure 4.4: Flow diagram of trauma-detecting PLB microcontroller program.

As the user may not always be wearing the device when powered on, the program must also be able to distinguish between practical and impractical changes in physiological conditions. While the electrodermal sensor itself may be able to detect when the device is removed, this device must be able to factor in the rate of change in vital signs. Example emergency conditions and criteria can be found in Table 4.1.

Table 4.1: Examples of emergency conditions and sensor criteria for alarm activation

Emergency Condition	Blood Oxygen Sat.	Pulse	Electrodermal Activity	G-Forces	Skin Temperature
Fall	-	-	-	Freefall followed by high-G spike	-
Heat Stoke	-	High	Very Low	-	Very High
Hypothermia	-	Low, Weak	-	-	Very Low
Hypoxia	Very Low	High	High	-	-
Bradycardia	-	Very Low	-	-	-
Cardiac Emergency	-	Sudden change or irregular	Sudden Increase	-	-

4.2.6 PLB Transmitter

As the highest priority section of the device, the PLB transmitter would contain the standard circuits of a PLB for a distress beacon operating at 406 MHz. As the antenna is often bulky due to the wavelength used, a collapsible antenna could be designed into the wristband that could automatically deploy when needed. Using the flexible antenna design commonly found on devices such as the McMurdo FastFind Max or the OceanSignal RescueME, the antenna could be wrapped around the device and a mechanical actuator would release the antenna when triggered.

4.3 Summary

In summary, this chapter outlined a concept for a novel device to provide incapacitated hikers with emergency care. Using a pulse oximeter, accelerometer, electrodermal activity sensor and thermistor embedded into a wristband, the user's vitals can be monitored to determine if the user is in critical condition. If detected, an alarm sequence can be initiated that ultimately triggers a 406 MHz distress signal for emergency services. By miniaturizing this device into a wristband, this device could provide an additional level of safety to those exploring the outdoors.

CHAPTER 5 : CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

In this thesis, three bioelectronic systems were proposed: an assistive technology computer mouse using electrooculographic (EOG) signals, a wireless sleep study system to provide higher-quality data in sleep studies, and a trauma-detecting personal locator beacon to provide emergency services to incapacitated hikers due to injury.

The eye-gaze tracking device was first designed, simulated and built on a breadboard as a proof-of-concept, later being translated into an embedded system on a PCB. Due to the large size of the PCB and observed signal saturation problems, the system was redesigned using an AFE (analog front end) for its amplification and data acquisition functions. This solution allowed for miniaturization of the system and resolved the saturation problems. While the horizontal channel was able to provide moderately accurate readings, accuracy in the vertical channel proved to be difficult, particularly due to signal drift.

Using the same solution as the eye-tracker, the wearable sleep study system was also designed around an AFE for data acquisition of the bioelectric signals. An impedance plethysmograph was designed and simulated before also being integrating into the system for providing measurement of respiratory effort. This prototype system included separate modules for detection of SPO_2 , respiratory flow and body position. While still in development, this design has shown promising signal results with signal quality comparable to that of a gold-standard polysomnography device, such as the Cadwell Easy III or Philips Alice 6 LDx.

The personal beacon locator (PLB) was designed based on lessons learned from the first two systems and parts identified to implement the design.

5.2 Future Work

While the prototype system functioned well enough to prove out the hardware design strategy implemented during this thesis research, the eye-tracking system needs development in its signal processing algorithm. First, a low-pass Butterworth can be included in the microcontroller firmware for reduction of noise in the system, allowing for the removal of observed jitter in cursor position. In order to address the issue of EOG drift, an automated calibration system may be implemented using an artificial neural network, as reported by Coughlin, Cutmore and Hine, whereby implementing this type of algorithm increased the eye tracking accuracy to approximately 1.09° of visual angle [23]. In addition, further work is needed for developing an algorithm to factor in head movement in the calibration sequence. Ultimately, glasses also need to be designed to house the needed surface electrodes to the user and allow simple and repeatable placement of the electrodes.

Whereas the wearable sleep sensor design is still a work in progress, further development of the DSP program and filter implementation is needed for accurate signal measurements. After an IRB is approved, clinical trials may be conducted using the developed device to compare and contrast data to gold standard polysomnography (PSG) devices. The resulting data will be used to correct any faults in the device before proceeding with miniaturization and patch design. In addition advanced algorithms may then be tested to further increase the utility of the system and drive an additional design optimization loop to reduce electronics part count and complexity, if possible, while maintaining system functionality.

The trauma-detecting PLB device was proposed in this thesis as a concept design and should be reduced to practice in a prototype design. The parts have already been identified and the same procedure as the eye tracker and sleep sensor will be used to prove out the design (breadboard) and provide a user-friendly PLB system (PCB design and firmware implementation).

REFERENCES

- [1] A. Lymberis, "Smart wearables for remote health monitoring, from prevention to rehabilitation: current R&D, future challenges," in *4th International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine, 2003.*, 2003, pp. 272-275.
- [2] "Complete Public Version of the 2016 Annual Statistical Report for the Spinal Cord Injury Model Systems," National Spinal Cord Injury Statistical Center 2016.
- [3] A. Singh, L. Tetreault, S. Kalsi-Ryan, A. Nouri, and M. G. Fehlings, "Global prevalence and incidence of traumatic spinal cord injury," *Clinical Epidemiology*, vol. 6, pp. 309-331, 09/23 2014.
- [4] R. S. Snell, *Clinical Neuroanatomy*, 7 ed.: Lippincott Williams & Wilkins, 2010.
- [5] W. W. Flemons, N. J. Douglas, S. T. Kuna, D. O. Rodenstein, and J. Wheatley, "Access to diagnosis and treatment of patients with suspected sleep apnea," *Am J Respir Crit Care Med*, vol. 169, pp. 668-72, Mar 15 2004.
- [6] L. Celmer, "Demand for treatment of sleep illness is up as drowsy Americans seek help for potentially dangerous conditions," ed: American Academy of Sleep Medicine, 2012.
- [7] E. Herbst, T. J. Metzler, M. Lenoci, S. E. McCaslin, S. Inslicht, C. R. Marmar, *et al.*, "Adaptation effects to sleep studies in participants with and without chronic posttraumatic stress disorder," *Psychophysiology*, vol. 47, pp. 1127-33, Nov 2010.
- [8] D. A. Robinson, "A Method of Measuring Eye Movement Using a Scieral Search Coil in a Magnetic Field," *IEEE Transactions on Bio-medical Electronics*, vol. 10, pp. 137-145, 1963.
- [9] H. D. Crane and C. M. Steele, "Generation-V dual-Purkinje-image eyetracker," *Applied Optics*, vol. 24, pp. 527-537, 1985/02/15 1985.
- [10] B. Estrany, P. Fuster, A. Garcia, and Y. Luo, "Human computer interface by EOG tracking," presented at the Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments, Athens, Greece, 2008.
- [11] J. Malmivuo and R. Plonsey, *Bioelectromagnetism*, 1995.
- [12] "ISO124 Precision Lowest-Cost Isolation Amplifier," *Texas Instruments*, 2016.
- [13] "ADS1299-x Low-Noise, 4-, 6-, 8-Channel, 24-Bit, Analog-to-Digital Converter for EEG and Biopotential Measurements," *Texas Instruments*, 2017.

- [14] I. S. Isa, B. S. Zainuddin, Z. Hussain, and S. N. Sulaiman, "Preliminary Study on Analyzing EEG Alpha Brainwave Signal Activities Based on Visual Stimulation," *Procedia Computer Science*, vol. 42, pp. 85-92, 2014/01/01/ 2014.
- [15] "Sleep Technology: Technical Guideline - Standard Polysomnography," ed: American Association of Sleep Technologists, 2012.
- [16] F.-T. Wang, H.-L. Chan, C.-L. Wang, H.-M. Jian, and S.-H. Lin, "Instantaneous Respiratory Estimation from Thoracic Impedance by Empirical Mode Decomposition," *Sensors (Basel, Switzerland)*, vol. 15, pp. 16372-16387, 2015.
- [17] R. M. Fish and L. A. Geddes, "Conduction of Electrical Current to and Through the Human Body: A Review," *Eplasty*, vol. 9, p. e44, 10/12 2009.
- [18] "BMI160 - Small, low power inertial measurement unit," *Bosch*, 2015.
- [19] "MAX30102 - High-Sensitivity Pulse Oximeter and Heart-Rate Sensor for Wearable Health," *Maxim Integrated*, 2015.
- [20] H. Lee, H. Ko, and J. Lee, "Reflectance pulse oximetry: Practical issues and limitations," *ICT Express*, vol. 2, pp. 195-198, 2016/12/01/ 2016.
- [21] G. Pang and C. Ma, "A Neo-Reflective Wrist Pulse Oximeter," *IEEE Access*, vol. 2, pp. 1562-1567, 2014.
- [22] T. W. Heggie and M. E. Amundson, "Dead Men Walking: Search and Rescue in US National Parks," *Wilderness & Environmental Medicine*, vol. 20, pp. 244-249, 2009/09/01/ 2009.
- [23] M. J. Coughlin, T. R. H. Cutmore, and T. J. Hine, "Automated eye tracking system calibration using artificial neural networks," *Computer Methods and Programs in Biomedicine*, vol. 76, pp. 207-220, 2004/12/01/ 2004.

APPENDICES

Appendix A: Initial Eye-Tracker Schematics

A.1 Breadboard

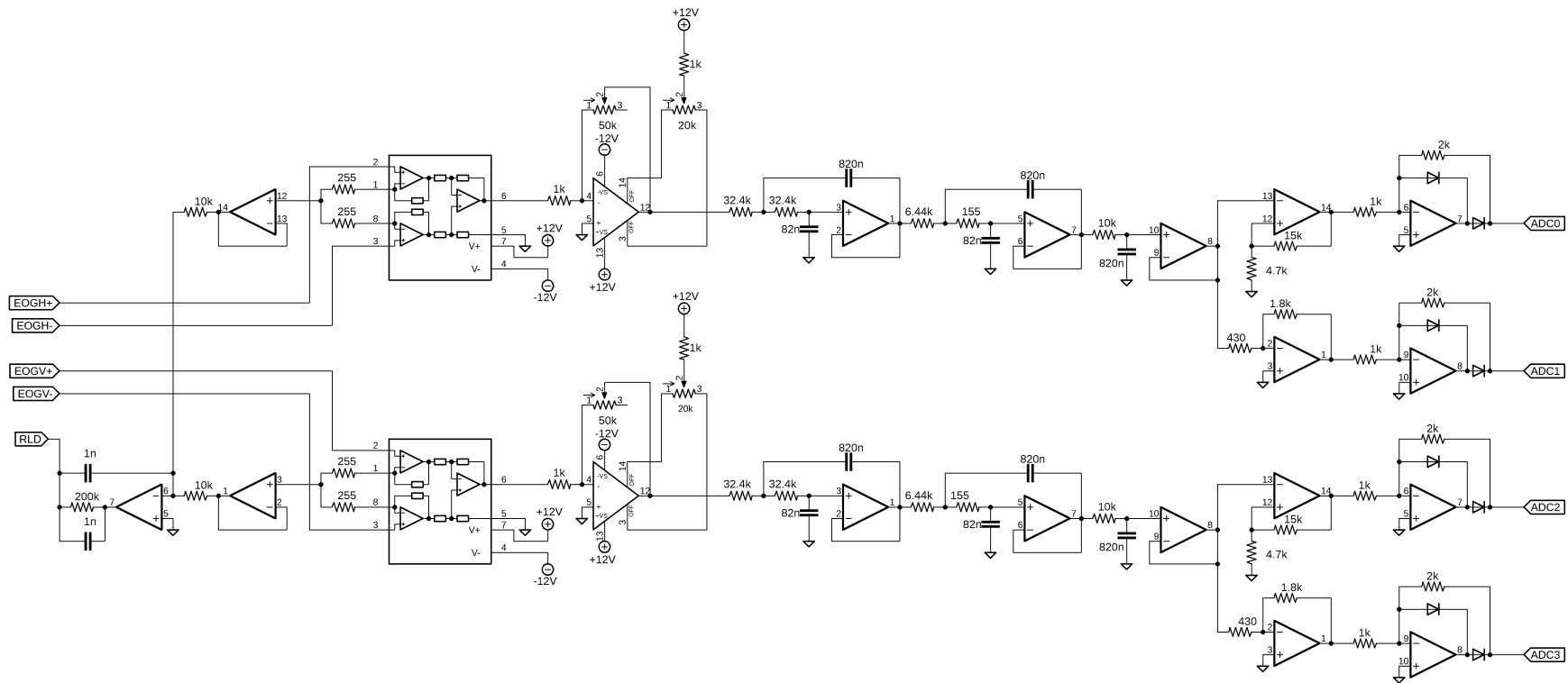


Figure A.1: Breadboard schematic of eye-tracker

Appendix A: (continued)

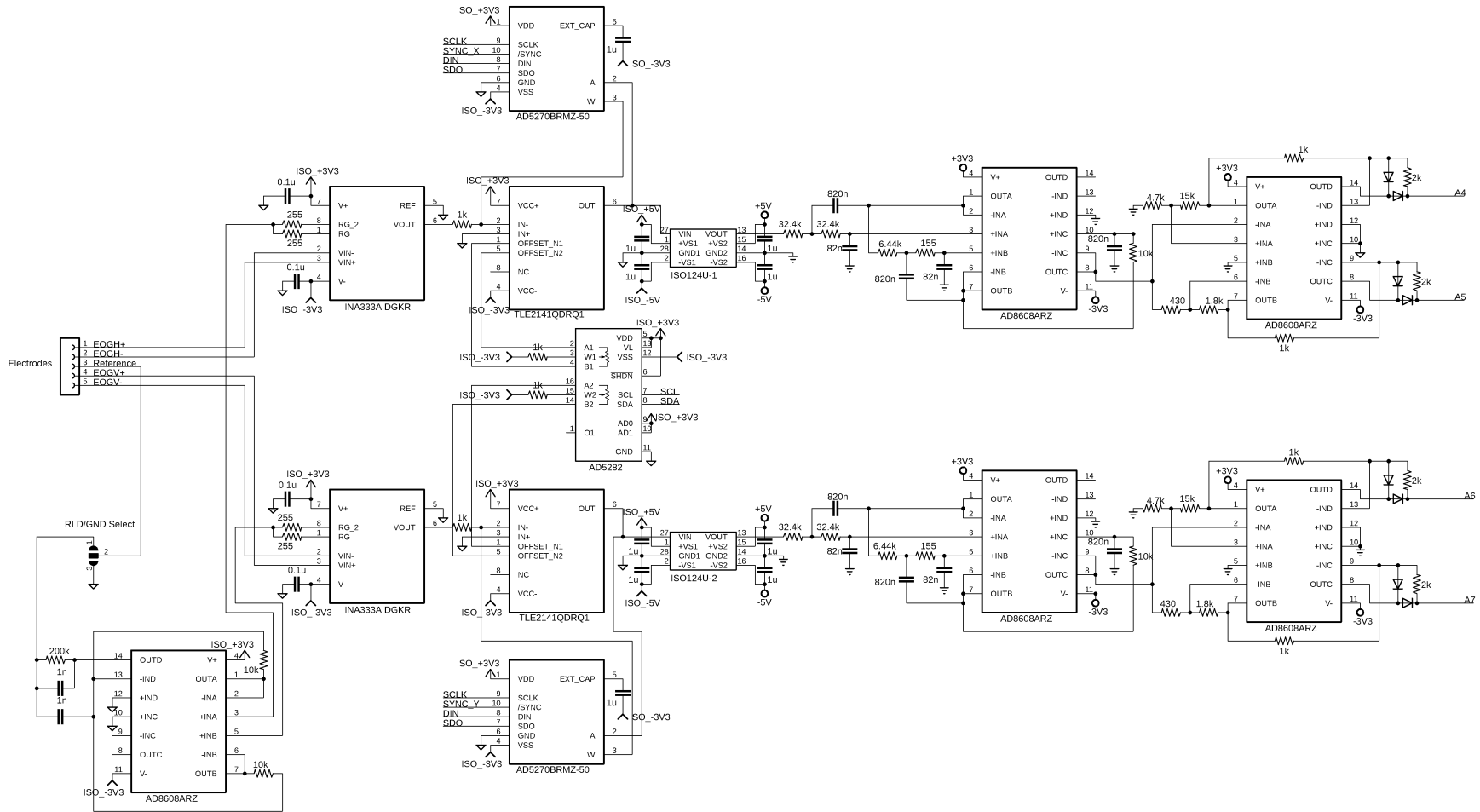


Figure A.4: Initial eye-tracker PCB signal conditioning

Appendix B: Initial Eye-Tracker Program

```
#include <Wire.h>
#include <SPI.h>

char stage; // Stage of calibration
String resolution; // Received resolution
int horizontal; // Horizontal resolution (after parse)
int vertical; // Vertical Resolution (after parse)
bool eyetrack; // Calibration success/fail

// Analog input pins
int horizplus = A5;
int horizminus = A4;
int vertplus = A7;
int vertminus = A6;

// SPI chip select pins
int csHorizGain = P2_2;
int csVertGain = P2_6;

// Coordinate variables
float horizcoordinate;
float vertcoordinate;
int horizcoordinateint;
int vertcoordinateint;
bool clickbit;
int prevcoordinate;
int difference;

// Calibration parameters
int offsettolerance = 10; // (+/-)
int gaintolerance = 10; // (+/-)
int clickthreshold = 10; // difference between samples to detect blink

void setup() {

    // Initiate USB and SPI communication
    Serial.begin(115200);

    // Initiate SPI
    SPI.begin();
    pinMode(csHorizGain, OUTPUT);
    pinMode(csVertGain, OUTPUT);
    digitalWrite(csHorizGain, LOW);
    digitalWrite(csVertGain, LOW);

    // Initiate I2C
    Wire.begin();

    // Wait until USB connection detected
```

Appendix B: (continued)

```
while (!Serial) {}

// Read initial string when available
while(!Serial.available()){
  resolution = Serial.readString();
  if (resolution.substring(0, 1) == "@"){ // Verify that the first character is "@"
    for (int i = 1; i < resolution.length(); i++) {
      if (resolution.substring(i, i+1) == ",") { // Use "," as delimiter
        horizontal = resolution.substring(1, i).toInt(); // Set value before "," as horizontal
        resolution
          vertical = resolution.substring(i+1).toInt(); // Set value after "," as vertical resolution
          break;
        }
      }
    }
  Serial.write("!");
}

void loop() {
  // Run calibration
  eyetrack = Calibration();
  String packet;
  clickbit = false;
  while(eyetrack == true){ // track if calibration successful
    // Calculate coordinates
    horizcoordinate = (2000 * analogRead(horizplus) - 2000 * analogRead(horizminus) +
8000000) * horizontal;
    // (((horizplus-horizminus) + ADC resolution)/2) / ADC resolution) * horizontal
    resolution
    vertcoordinate = (-2000 * analogRead(vertplus) + 2000 * analogRead(vertminus) +
8000000) * vertical;
    // (((-(vertplus-vertminus) + ADC resolution)/2) / ADC resolution) * vertical resolution
    horizcoordinateint = horizcoordinate; // float to int
    vertcoordinateint = vertcoordinate; // float to int
    difference = vertcoordinate - prevcoordinate;
    if(difference > clickthreshold){
      clickbit = clickdetect(vertcoordinateint);
      if (clickbit == true){
        vertcoordinateint = prevcoordinate;
      }
    }
  }

  packet = horizcoordinateint + "," + vertcoordinateint + "," + clickbit;
  Serial.write(packet);

  prevcoordinate = vertcoordinateint;

  // If "1" received, end eye tracking and recalibrate
  if (Serial.available()){
```

Appendix B: (continued)

```
        if (Serial.read() == 1){
            eyetrack = false;
        }
    }
}
```

```
bool Calibration(){ // bool for success/fail
```

```
    // Set initial gain in both channels to minimum
    digitalWrite(csHorizGain, HIGH);
    SPI.transfer(1024); //0x00010000000000 or R = 0
    digitalWrite(csHorizGain, LOW);
    digitalWrite(csVertGain, HIGH);
    SPI.transfer(1024); //0x00010000000000 or R = 0
    digitalWrite(csVertGain, LOW);
```

```
    // Look center (offset adjustment)
    while(!Serial.available()){}
    stage = Serial.read();
    if (stage == '0'){
        // Vertical offset
        int offset = 100; // Arbitrary non-zero value outside of tolerance
        for(int i = 0; i < 256; i++){
            // I2C to offset pot
            Wire.beginTransmission(94); // device address as specified in datasheet
            Wire.write(0); // sends instruction byte, for channel 1
            Wire.write(i); // sends potentiometer value byte
            Wire.endTransmission(); // stop transmitting

            // Check offset
            offset = analogRead(vertplus) - analogRead(vertminus);

            // If within tolerance, break for loop
            if((offsettolerance * -1) < offset < offsettolerance){
                break;
            }
            // If max value reached, terminate calibration
            if(offset == 255){
                Serial.write("0"); // Fail!!!
                return false;
            }
        }
    }
```

```
    // Horizontal offset
    offset = 100; // Arbitrary non-zero value
    for(int i = 0; i < 256; i++){
        // I2C to offset pot
        Wire.beginTransmission(94); // device address is specified in datasheet
```

Appendix B: (continued)

```
Wire.write(128); // sends instruction byte, for channel 2
Wire.write(i); // sends potentiometer value byte
Wire.endTransmission(); // stop transmitting

// Check offset
offset = analogRead(horizplus) - analogRead(horizminus);

// If within tolerance, break for loop
if((offsettolerance * -1) < offset < offsettolerance){
    Serial.write("1"); // Success!!!
    break; // Break for loop
}
// If max value reached, terminate calibration
if(offset == 255){
    Serial.write("0"); // Fail!!!
    return false;
}
}
}

// Look Up (vertical gain adjustment)
while(!Serial.available()){
stage = Serial.read();
int vert = 100; // Arbitrary non-zero value
if (stage == '1'){
    for(int i = 0; i < 1024; i++){
        digitalWrite(csVertGain, HIGH);
        SPI.transfer(1024 + i); // command 1 (1024) + wiper position
        digitalWrite(csVertGain, LOW);

        vert = analogRead(vertplus);

        if((gaintolerance * -1) < vert < gaintolerance){
            Serial.write("1"); // Success!!!
            break;
        }
        if(vert == 1023){
            Serial.write("0"); // Fail!!!
            return false;
        }
    }
}
}

// Look Down (verify vertical gain & offset calibration)
while(!Serial.available()){
stage = Serial.read();
if (stage == '2'){
    vert = analogRead(vertminus);
    if((gaintolerance * -2) < vert < (gaintolerance * 2)){
```


Appendix B: (continued)

```
        Serial.write("1"); // Success!!!
    }
    else{
        Serial.write("0"); // Fail!!!
        return false;
    }
}

// Look Left (horizontal gain adjustment)
while(!Serial.available()){
stage = Serial.read();
int horiz = 100; // Arbitrary non-zero value
if (stage == '3'){
    for(int i = 0; i < 1024; i++){
        digitalWrite(csHorizGain, HIGH);
        SPI.transfer(1024 + i); // command 1 (1024) + wiper position
        digitalWrite(csHorizGain, LOW);

        horiz = analogRead(horizminus); // minus since left is assumed negative

        if((gaintolerance * -1) < horiz < gaintolerance){
            Serial.write("1"); // Success!!!
            break;
        }
        if(vert == 1023){
            Serial.write("0"); // Fail!!!
            return false;
        }
    }
}
}

// Look Right (verify horizontal gain & offset calibration)
while(!Serial.available()){
stage = Serial.read();
if (stage == '4'){
    horiz = analogRead(horizplus);
    if((gaintolerance * -2) < horiz < (gaintolerance * 2)){
        Serial.write("1"); // Success!!!
    }
    else{
        Serial.write("0"); // Fail!!!
        return false;
    }
}
}
}

bool clickdetect(int vertcoordinate){
}
```

Appendix C: Wearable Wireless Eye-Tracker Schematics

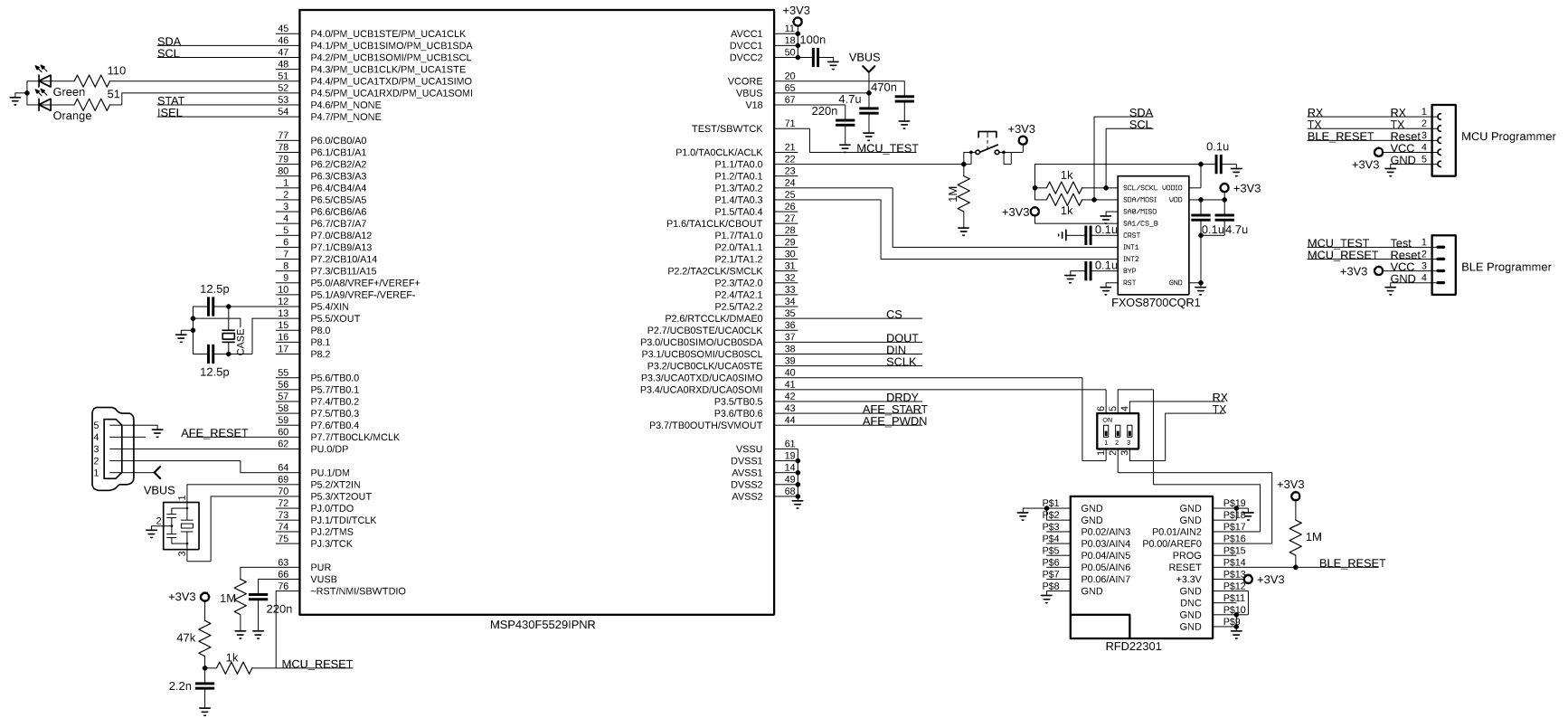


Figure C.1: Wearable wireless eye-tracker microcontroller and peripherals interface

Appendix C: (continued)

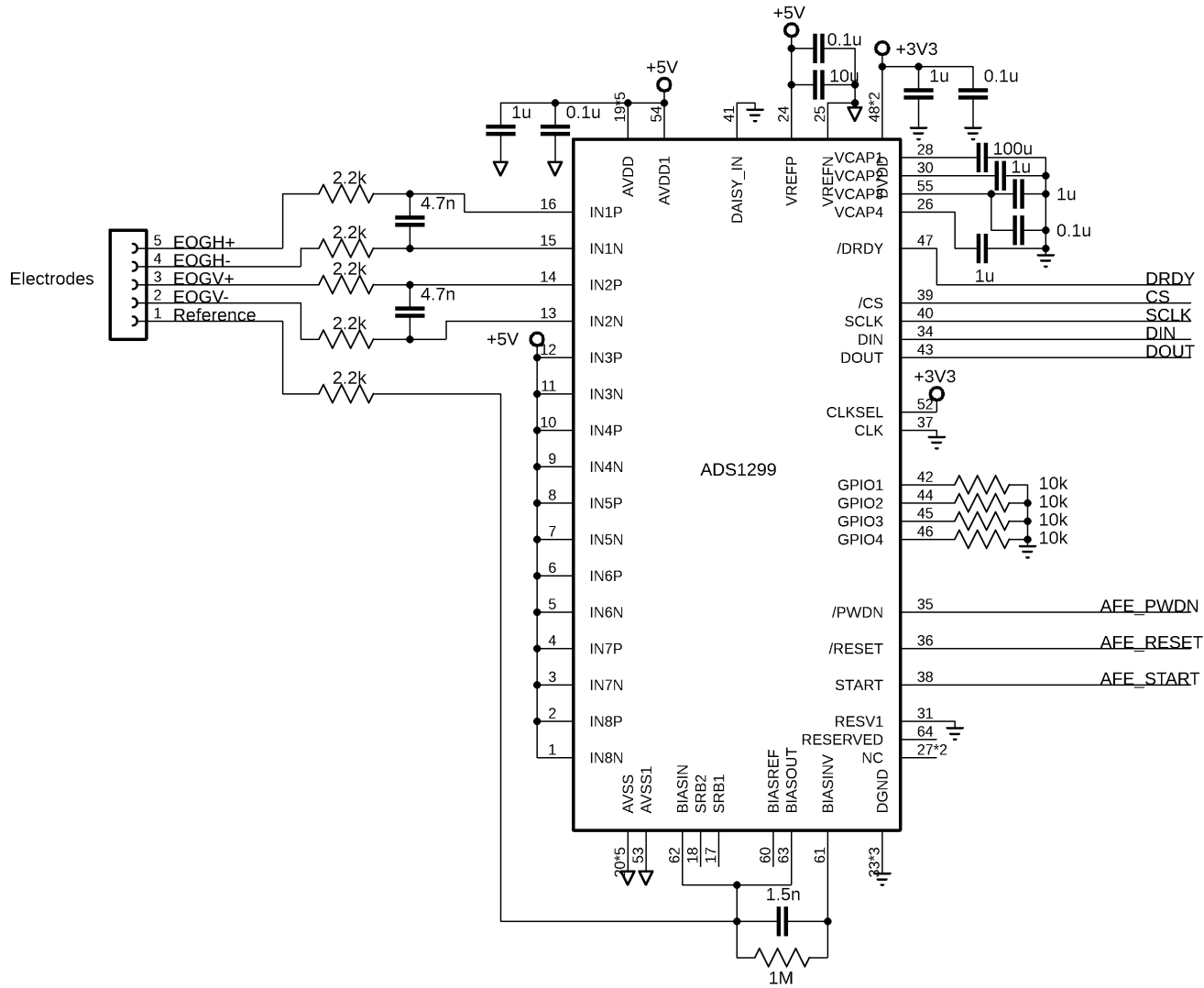


Figure C.3: Wearable wireless eye-tracker analog front end interface

Appendix D: Eye-Tracker Program for MSP430F5529

```
#include <msp430f5529.h>
#include <driverlib/MSP430F5xx_6xx/driverlib.h>
#include <stdlib.h>

// Pin register bits
#define POWER_LED          BIT4           //green LED (port 4)
#define RUN_LED           BIT5           //orange LED (port 4)
#define BUTTON            BIT1           //button (port 1)
#define TXD               BIT3           //UART TX (port 3)
#define RXD               BIT4           //UART RX (port 3)
#define SOMI              BIT0           //SPI DOUT (port 3)
#define SIMO              BIT1           //SPI DIN (port 3)
#define SCLK              BIT2           //SPI CLK (port 3)
#define DRDY              BIT1           //AFE Data ready, active low (port 2)
#define CS                BIT6           //AFE Chip Select (port 2)
#define START_AFE        BIT6           //AFE Start (port 3)
#define PWDN              BIT7           //AFE Power Down (port 3)
#define RESET_AFE        BIT7           //AFE Reset (port 7)

// AFE commands (p.39 of ADS1299 datasheet)
#define WAKEUP            0x02
#define STANDBY           0x04
#define RESET             0x06
#define START             0x08
#define STOP              0x0A
#define RDATA_C           0x10
#define SDATA_C           0x11
#define RDATA             0x12

// Define AFE register values
#define CONFIG1           0b10010110    // 250SPS AFE sampling
#define CONFIG2           0b11000000    // Test signal definition
#define CONFIG3           0b11101100    // Use 5V/2 as sig ref, enable RLD
#define CONFIG4           0b00000000    // Disable lead-off comparators
#define LOFF              0b00000000    // Disable lead-off control
#define Ch1_2SET          0b01110000    // Enable channels 1 & 2 with 24x gain,
normal electrode input
#define Ch3_8SET          0b10000001    // Disable channels 3-8
#define LOFF_SENSP        0b00000000    // Disable all Lead-off P
#define LOFF_SENSN        0b00000000    // Disable all Lead-off N
#define BIAS_SENSP        0b00000011    // Enable RLD for channels 1P & 2P
#define BIAS_SENSN        0b00000000    // Disable all Bias-sense N

// Function variables
#define swap(x,y)         (x^=y, y^=x, x^=y)
#define SPICLK            2048000
char STAT_BUFFER[9];
char X_BUFFER[9];
char Y_BUFFER[9];
```

Appendix D: (continued)

```
uint32_t STAT;
int32_t X;
int32_t Y;
uint32_t X_Diff, Y_Diff;
uint32_t X_Min, X_Max;
uint32_t Y_Min, Y_Max;
uint32_t X_Mult, Y_Mult;
uint32_t X_Dim, Y_Dim;
bool MODE;

// Define functions
void UART_TX(char *tx_message); // UART transmission
char* itoa(long int num, char* str, int base); // Integer to ASCII conversion
void reverse(char str[], int length); // Reverse string
void initialize_UART(); // UART initialization procedure
void initialize_SPI(); // SPI initialization procedure
void initialize_AFE(); // AFE initialization procedure
void resolution(); // Extract screen resolution from UART
void calibration(); // Procedure for calibrating AFE
measurements with screen dimensions
void Read_Data(); // Procedure for Reading AFE data
through SPI
void Convert_to_Pixel(); // Convert AFE 24-bit value to pixel
coordinate

void main(void){

    // Initialize microcontroller
    static const uint32_t mclk = 20000000; // define microcontroller frequency (20 MHz)
    WDTCTL = WDTPW | WDTHOLD; // stop watchdog timer
    PMM_setVCore(PMM_CORE_LEVEL_3); // set vcore to level 3
    UCS_initFLLSettle(mclk/1000, mclk/32768); // set clk rate to mclk frequency

    // Initialize GPIO
    P4DIR |= POWER_LED + RUN_LED; // configure LEDs as output
    P1DIR |= ~BUTTON; // configure button as input
    P4OUT |= POWER_LED; // start with green LED on
    P4OUT &= ~RUN_LED; // start with orange LED off

    // initialize AFE pins
    P3DIR |= PWDN + START_AFE; // configure PWDN & START_AFE as
output
    P2DIR |= ~DRDY + ~SHORT; // configure DRDY (& SHORT pin) as
input
    P2DIR |= CS; // configure CS as output
    P7DIR |= RESET_AFE; // configure RESET as output
    P7OUT |= RESET_AFE; // Pull RESET_AFE high
    P3OUT |= PWDN; // Pull PWDN high
```

Appendix D: (continued)

```
    P3OUT &= ~START_AFE;           // Pull START pin low to initiate by
command                               // Pull CS pin high (active low)
    P2OUT |= CS;

    // Initialize Button interrupt
    P1IES &= ~BUTTON;             // Button interrupt edge select (low to
high)                                 // Button interrupt flag cleared
    P1IFG &= ~BUTTON;             // Button interrupt enabled
    P1IE |= BUTTON;

    // Initialize communications
    initialize_UART();
    initialize_SPI();
    initialize_AFE();

    // Enable interrupts
    __bis_SR_register(GIE);
    while(UCA0STAT & UCIBUSY);    // Wait until UART data received

    while(1){
        // If "@" received, enter Cursor Coordinate mode
        if(UCA0RXBUF == '@'){
            MODE = true;
            resolution();          // Read screen resolution from computer
            calibration();         // Perform calibration sequence
            break;
        }
        // If "!" received, enter Raw mode
        if(UCA0RXBUF == '!'){
            MODE = false;
            break;
        }
    }

    // Cursor Coordinate mode
    while(MODE == true){

        // Turn on Orange LED
        P4OUT |= RUN_LED;

        // Read AFE data
        Read_Data();

        // Convert value to pixel coordinate
        Convert_to_Pixel();

        // Convert coordinates to ASCII characters
        itoa(X, X_BUFFER, 10);
        itoa(Y, Y_BUFFER, 10);
```

Appendix D: (continued)

```
// Send X and Y coordinates to computer
UART_TX(X_BUFFER);
__delay_cycles(25);
UART_TX("\t");
__delay_cycles(25);
UART_TX(Y_BUFFER);
__delay_cycles(25);
UART_TX("\n");
__delay_cycles(25);
}

// Raw Data mode
while(MODE == false){

    // Turn on Orange LED
    P4OUT |= RUN_LED;

    // Read AFE data
    Read_Data();

    // Convert coordinates to ASCII characters
    itoa(X, X_BUFFER, 10);
    itoa(Y, Y_BUFFER, 10);

    // Send X and Y values to computer
    UART_TX(X_BUFFER);
    __delay_cycles(25);
    UART_TX("\t");
    __delay_cycles(25);
    UART_TX(Y_BUFFER);
    __delay_cycles(25);
    UART_TX("\n");
    __delay_cycles(25);
}

// Read AFE data
void Read_Data(void){

    // Wait until DRDY toggle, indicating device ready
    while((P2IN & DRDY) == DRDY);

    // Receive status bytes (24-bits, 3-bytes)
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL); // Transmit null value
through SPI
    STAT = USCI_B_SPI_receiveData(USCI_B0_BASE); // Assign received
byte to variable
    STAT = STAT * 0b100000000; // Shift 8 bits
}
```


Appendix D: (continued)

```
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);           // Transmit null value
through SPI
    STAT = STAT + USCI_B_SPI_receiveData(USCI_B0_BASE);    // Add old value to
new value and assign to variable
    STAT = STAT * 0b100000000;                             // Shift 8 bits
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);           // Transmit null value
through SPI
    STAT = STAT + USCI_B_SPI_receiveData(USCI_B0_BASE);    // Add old value to
new value and assign to variable
```

```
    // Receive x-direction data
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);
    X = USCI_B_SPI_receiveData(USCI_B0_BASE);
    X = X * 0b100000000;
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);
    X = X + USCI_B_SPI_receiveData(USCI_B0_BASE);
    X = X * 0b100000000;
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);
    X = X + USCI_B_SPI_receiveData(USCI_B0_BASE);
```

```
    // Receive y-direction data
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);
    Y = USCI_B_SPI_receiveData(USCI_B0_BASE);
    Y = Y * 0b100000000;
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);
    Y = Y + USCI_B_SPI_receiveData(USCI_B0_BASE);
    Y = Y * 0b100000000;
    USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);
    Y = Y + USCI_B_SPI_receiveData(USCI_B0_BASE);
```

```
}
```

```
void Convert_to_Pixel(void){
```

```
    // Convert ADC value to coordinate based on calibration data
    X = ((X - X_Min) / X_Mult);
    Y = ((Y - Y_Min) / Y_Mult);
```

```
    // Assign max and min value to X
```

```
    if(X > X_Dim){
        X = X_Dim;
    }
```

```
    if(X < 0){
        X = 0;
    }
```

```
    // Assign max and min value to Y
```

```
    if(Y > Y_Dim){
        Y = Y_Dim;
    }
```

```
}
```

Appendix D: (continued)

```
    if(Y < 0){
        Y = 0;
    }

    // Invert Y
    Y = Y_Dim - Y;
}

// Run initialization procedures for UART interface with computer
void initialize_UART(void){

    P3SEL |= TXD + RXD; // configure TXD and RXD pins as special
function (UART)
    UCA0CTL1 |= UCSWRST; // Put state machine in reset
    UCA0CTL1 |= UCSSEL_2; // SMCLK set to internal clk (20MHz)
    UCA0BR0 = 173; // 20MHz 115200 (see user's guide p.951)
    UCA0BR1 = 0; // 20MHz 115200
    UCA0MCTL |= UCBRS_1 + UCBRF_0; // Modulation UCBRSx=1, UCBRFx=0
    UCA0CTL1 &= ~UCSWRST; // Initialize USCI state machine
}

// Run initialization procedures for SPI interface with AFE
void initialize_SPI(void){
    uint8_t returnValue = 0x00;

    // Wait 0.5s to ensure proper power-up sequence for AFE
    __delay_cycles(10000000);

    // Toggle AFE reset pin
    P7OUT &= ~RESET_AFE;
    __delay_cycles(300);
    P7OUT |= RESET_AFE;

    //Assign SPI pins to SPI interface
    GPIO_setAsPeripheralModuleFunctionInputPin(GPIO_PORT_P3, SOMI + SIMO + SCLK);

    // Turn AFE on and wait for initialization
    P3OUT |= PWDN;
    __delay_cycles(300000);

    //Initialize Master
    USCI_B_SPI_initMasterParam param = {0};
    param.selectClockSource = USCI_B_SPI_CLOCKSOURCE_SMCLK;
    param.clockSourceFrequency = UCS_getSMCLK();
    param.desiredSpiClock = SPICLK;
    param.msbFirst = USCI_B_SPI_MSB_FIRST;
    param.clockPhase =
USCI_B_SPI_PHASE_DATA_CHANGED_ONFIRST_CAPTURED_ON_NEXT;
    param.clockPolarity = USCI_B_SPI_CLOCKPOLARITY_INACTIVITY_LOW;
```

Appendix D: (continued)

```
returnValue = USCI_B_SPI_initMaster(USCI_B0_BASE, &param);

// Break function if SPI initialization fails
if (STATUS_FAIL == returnValue){
    return;
}

//Enable SPI module
USCI_B_SPI_enable(USCI_B0_BASE);

//Wait for slave to initialize
__delay_cycles(100);

// Enable chip select
P2OUT &= ~CS;

//Transmit Data to slave
USCI_B_SPI_transmitData(USCI_B0_BASE, NULL);
}

// Define AFE registers
void initialize_AFE(void){
    USCI_B_SPI_transmitData(USCI_B0_BASE,SDATAC);           // Put AFE in
SDATAC mode
    USCI_B_SPI_transmitData(USCI_B0_BASE,0b01000001);     // WREG starting
at address 01h
    USCI_B_SPI_transmitData(USCI_B0_BASE,0x0F);           // Write to 16
registers
    USCI_B_SPI_transmitData(USCI_B0_BASE,CONFIG1);        // Configure AFE
- 1
    USCI_B_SPI_transmitData(USCI_B0_BASE,CONFIG2);        // Configure AFE
- 2
    USCI_B_SPI_transmitData(USCI_B0_BASE,CONFIG3);        // Configure AFE - 3
    USCI_B_SPI_transmitData(USCI_B0_BASE,LOFF);           // Lead-off
configuration
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch1_2SET);       // Channel 1
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch1_2SET);       // Channel 2
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch3_8SET);       // Channel 3
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch3_8SET);       // Channel 4
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch3_8SET);       // Channel 5
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch3_8SET);       // Channel 6
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch3_8SET);       // Channel 7
    USCI_B_SPI_transmitData(USCI_B0_BASE,Ch3_8SET);       // Channel 8
    USCI_B_SPI_transmitData(USCI_B0_BASE,BIAS_SENSP);     // RLD sense P
channels
    USCI_B_SPI_transmitData(USCI_B0_BASE,BIAS_SENSN);     // RLD sense N
channels
```

Appendix D: (continued)

```
    USCI_B_SPI_transmitData(USCI_B0_BASE,LOFF_SENSP);           // Lead-off P
channels
    USCI_B_SPI_transmitData(USCI_B0_BASE,LOFF_SENSN);           // Lead-off n
channels
    USCI_B_SPI_transmitData(USCI_B0_BASE,0b01010111);           // WREG starting at
17h
    USCI_B_SPI_transmitData(USCI_B0_BASE,0x00);                 // Write to 1 register
    USCI_B_SPI_transmitData(USCI_B0_BASE,CONFIG4);               // Configure AFE - 4
    USCI_B_SPI_transmitData(USCI_B0_BASE,START);                 // Start continuous
data sampling
    USCI_B_SPI_transmitData(USCI_B0_BASE,RDATAC);                 // Enable continuous
data sampling
}
```

// Send data to computer through UART

```
void UART_TX(char *tx_message)
{
    unsigned int i=0;                                           // Define end of string loop int
    char *message;                                             // Message variable
    unsigned int message_num;                                   // Define ascii int version variable
    message = tx_message;                                       // Move tx_message into message
    while(1)
    {
        if(message[i]==0)                                       // If end of input string is reached, break
loop.
        {break;}
        message_num = (int)message[i];                         // Cast string char into an int variable
        while (!(UCA0IFG & UCTXIFG));                           // USCI_A0 TX buffer ready?
        UCA0TXBUF = message_num;                                // Write int to TX buffer
        i++;                                                    // Increase string index
        __delay_cycles(10000);                                   // Transmission delay
        if(i>50){                                              // Prevent infinite transmit
            break;
        }
    }
}
```

// Read and assign screen resolution values to variables

```
void resolution(void){
    char buffer[1];
    while(UCA0RXBUF != '@');                                     // Wait until "@" is received
    while(UCA0RXBUF == '@');                                     // Wait until next value is received
    buffer[0] = UCA0RXBUF;                                       // Assign received to buffer
    X_Dim = atoi(buffer) * 1000;                                // Convert to integer and shift 4 places.
    while(UCA0STAT & UCBUSY);                                     // Wait until next value received
    buffer[0] = UCA0RXBUF;
    X_Dim = X_Dim + (atoi(buffer)*100);
    while(UCA0STAT & UCBUSY);
    buffer[0] = UCA0RXBUF;
```

Appendix D: (continued)

```
X_Dim = X_Dim + (atoi(buffer)*10);
while(UCA0STAT & UCBSY);
buffer[0] = UCA0RXBUF;
X_Dim = X_Dim + (atoi(buffer));
while(UCA0RXBUF != ',');
while(UCA0RXBUF == ',');
buffer[0] = UCA0RXBUF;
Y_Dim = atoi(buffer) * 1000;
while(UCA0STAT & UCBSY);
buffer[0] = UCA0RXBUF;
Y_Dim = Y_Dim + (atoi(buffer)*100);
while(UCA0STAT & UCBSY);
buffer[0] = UCA0RXBUF;
Y_Dim = Y_Dim + (atoi(buffer)*10);
while(UCA0STAT & UCBSY);
buffer[0] = UCA0RXBUF;
Y_Dim = Y_Dim + (atoi(buffer));
}
```

// Calibration sequence where 1 = X_Min, 2 = X_Max, 3 = Y_Max and 4 = Y_Min

```
void calibration(void){
```

```
    UART_TX("!");
```

```
    while(1){
```

```
        while(UCA0RXBUF != '1');
```

```
        Read_Data();
```

```
        X_Min = X;
```

```
        while(UCA0RXBUF != '2');
```

```
        Read_Data();
```

```
        X_Max = X;
```

```
        while(UCA0RXBUF != '3');
```

```
        Read_Data();
```

```
        Y_Max = Y;
```

```
        while(UCA0RXBUF != '4');
```

```
        Read_Data();
```

```
        Y_Min = Y;
```

```
        // Verify that values are valid
```

```
        if(X_Max > X_Min && Y_Max > Y_Min){
```

```
            break;
```

```
        }
```

```
        // If not valid, send back "1" to initiate recalibration
```

```
        UART_TX("1");
```

```
    }
```

```
    // Calculate multiplier for ADC - pixel coordinate conversion
```

```
    X_Mult = (X_Max - X_Min)/X_Dim;
```

Appendix D: (continued)

```
    Y_Mult = (Y_Max - Y_Min)/Y_Dim;
}

// Convert integer to ASCII character array
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);

    return str;
}

// A utility function to reverse a string
void reverse(char str[], int length)
{
```

Appendix D: (continued)

```
int start = 0;
int end = length -1;
while (start < end)
{
    swap(*(str+start), *(str+end));
    start++;
    end--;
}

// Reset button interrupt routine
#pragma vector=PORT1_VECTOR
__interrupt void Port_1(void){
    int count = 0;
    P4OUT &= ~RUN_LED;                // Turn off orange LED
    __delay_cycles(500);                // Delay 25ms for debounce
    while ((P1IN & BUTTON) > 0){
        count = count + 1;
        __delay_cycles(1000000);        // Delay 50ms

        // After 2 seconds, reset
        if (count > 40){
            P4OUT &= ~POWER_LED;        // Turn off green LED
            __delay_cycles(5000000);    // Delay 0.5s
            WDTCTL = WDTPW | ~WDTHOLD;  // Enable watchdog
            while(1);                    // Initiate infinite loop until watchdog resets
        }
    }
    P1IFG &= ~BUTTON;
}
```


Appendix E: (continued)

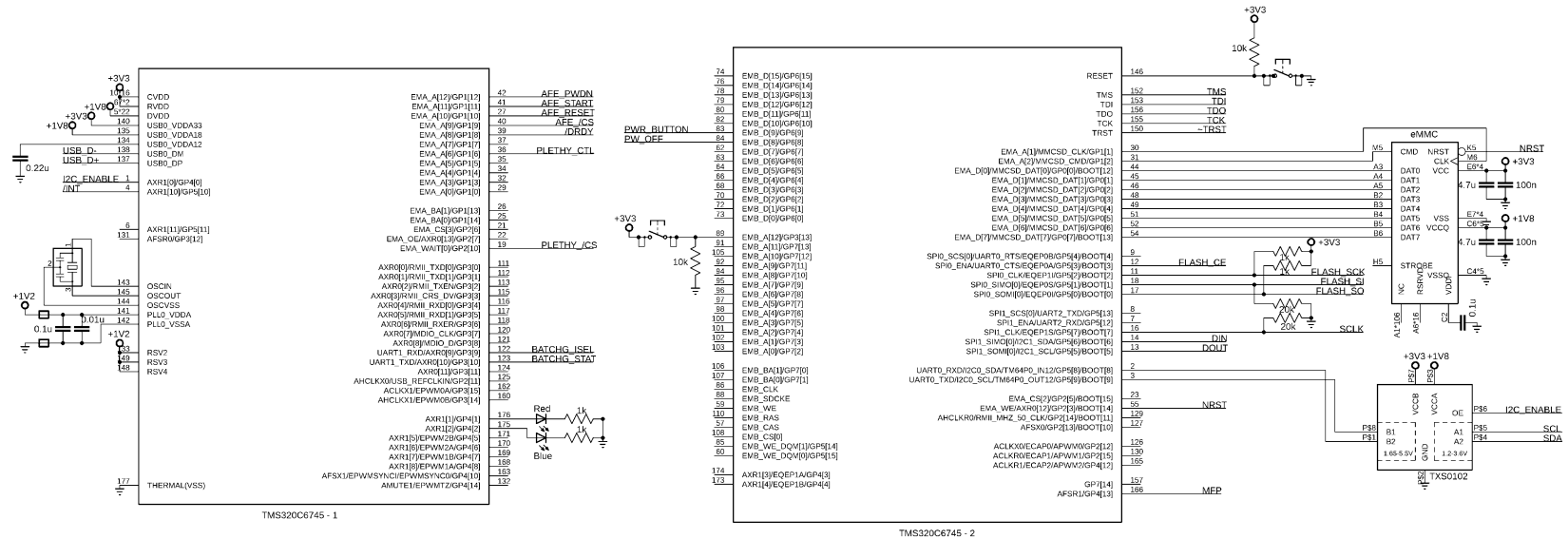


Figure E.2: Wearable wireless sleep study system DSP and eMMC interface

Appendix E: (continued)

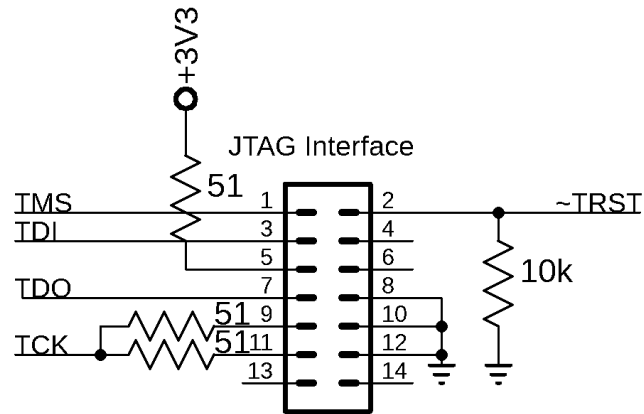
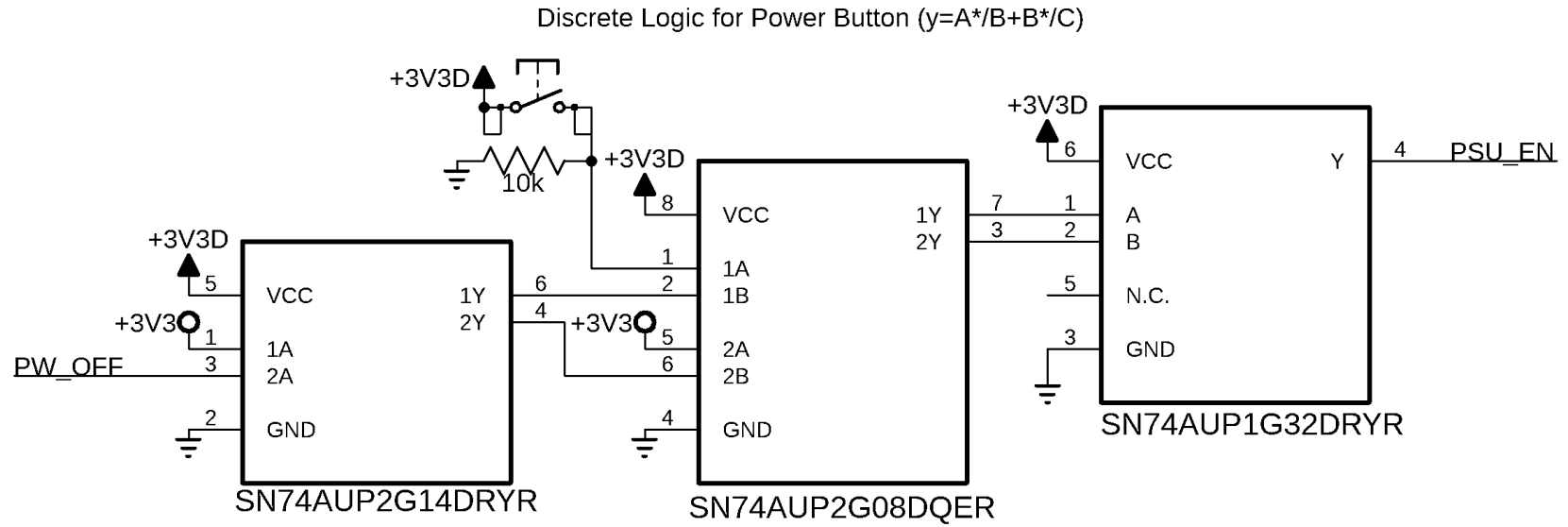


Figure E.3: Wearable wireless sleep study system power supply logic and JTAG header

Appendix E: (continued)

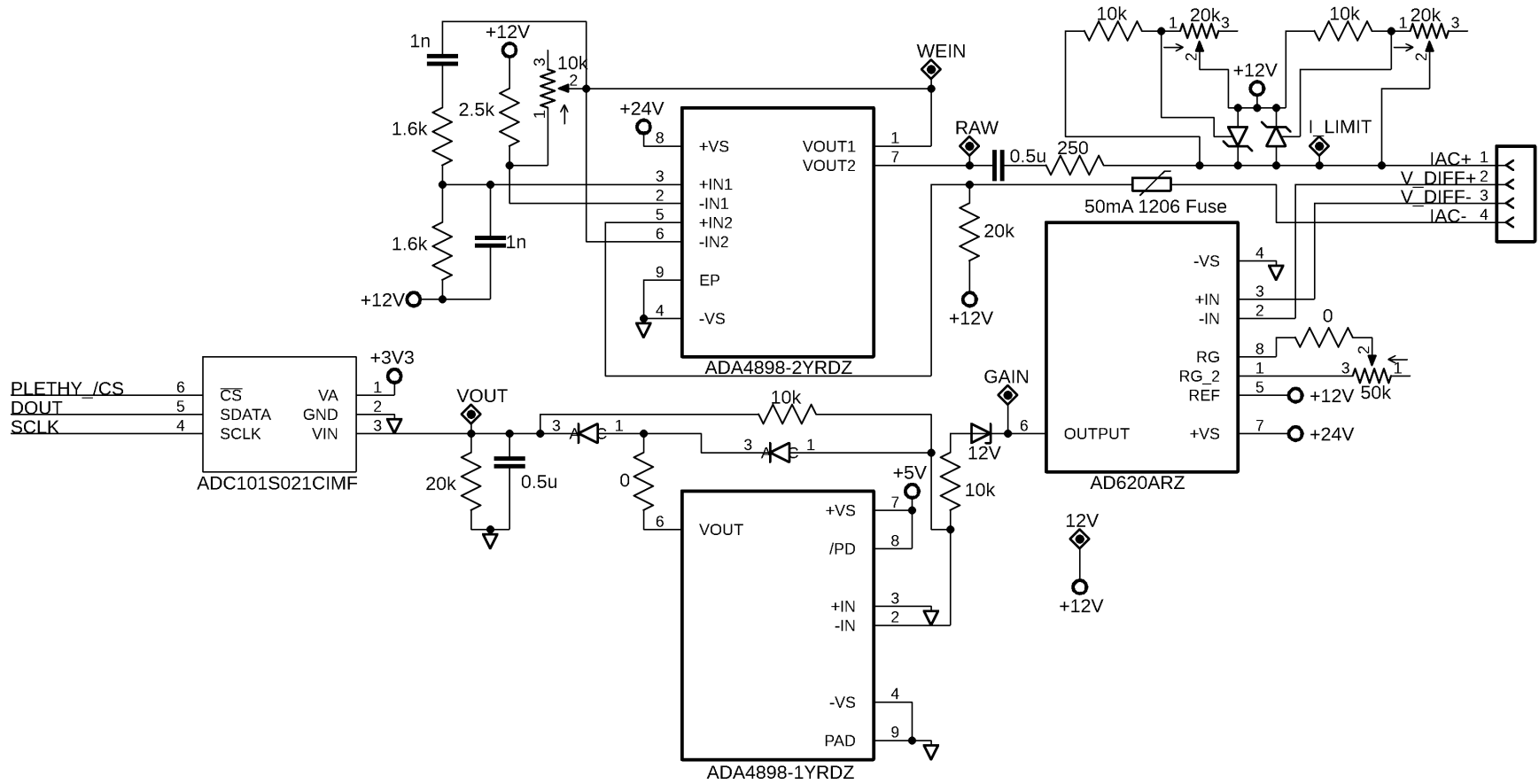


Figure E.5: Wearable wireless sleep study system respiratory impedance plethysmography

Appendix E: (continued)

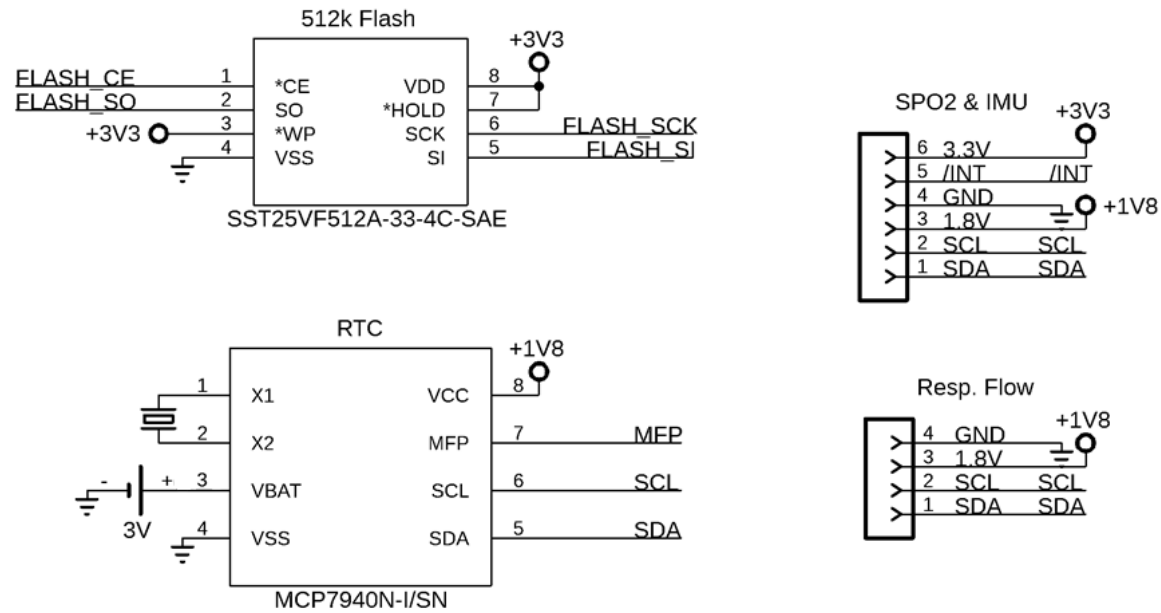


Figure E.6: Wearable wireless sleep study system flash, RTC and external board interface

Appendix E: (continued)

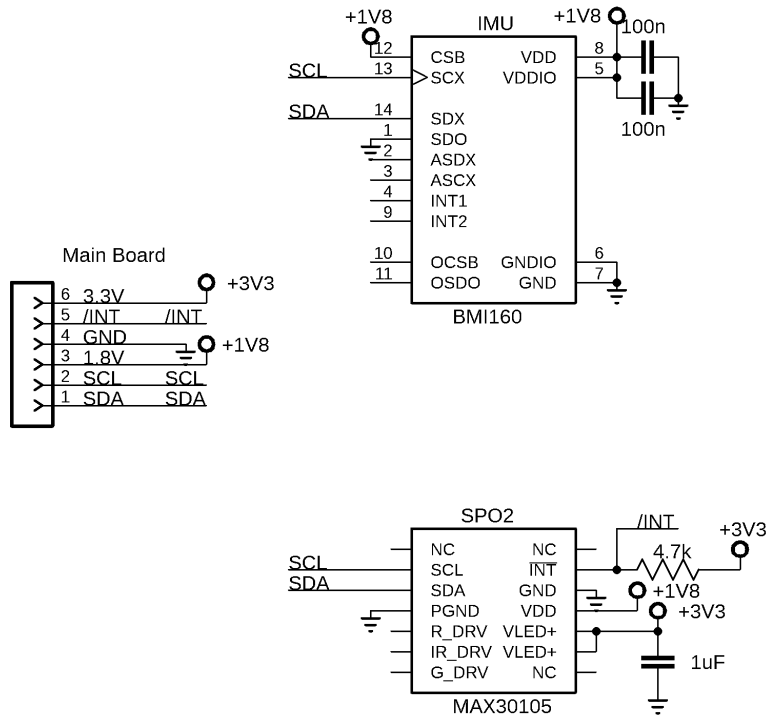


Figure E.7: Wearable wireless sleep study system external IMU and SPO₂ module

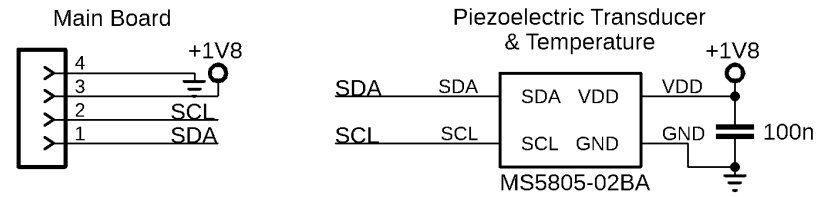
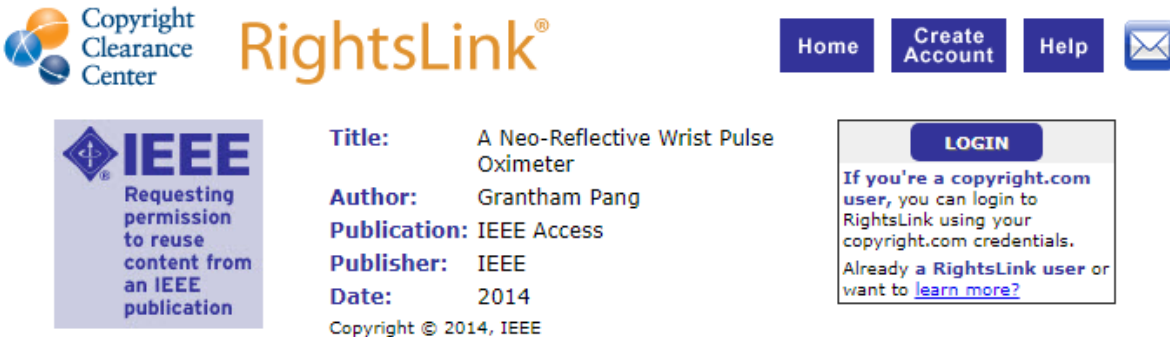


Figure E.8: Wearable wireless sleep study system external piezoelectric module

Appendix F: Copyright Permissions

The permission from IEEE to use Figure 4.2 is below.



The screenshot shows the IEEE RightsLink interface. At the top left is the Copyright Clearance Center logo. To its right is the RightsLink logo. Further right are navigation buttons for Home, Create Account, Help, and an email icon. Below the logo is a purple box with the IEEE logo and the text: "Requesting permission to reuse content from an IEEE publication". To the right of this box, the following information is displayed: Title: A Neo-Reflective Wrist Pulse Oximeter; Author: Grantham Pang; Publication: IEEE Access; Publisher: IEEE; Date: 2014. Below this information is the text "Copyright © 2014, IEEE". To the right of the information is a LOGIN button and a text box that says: "If you're a copyright.com user, you can login to RightsLink using your copyright.com credentials. Already a RightsLink user or want to learn more?"

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 WINDOW

Copyright © 2018 Copyright Clearance Center, Inc. All Rights Reserved. [Privacy statement](#). [Terms and Conditions](#).
Comments? We would like to hear from you. E-mail us at customer@copyright.com