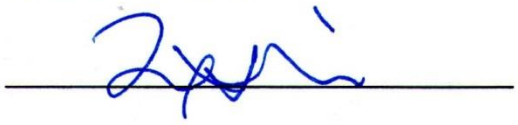
5-2014

Reading and Wirelessly Sending EEG Signals Using Arduinos and XBee Radios to Control a Robot

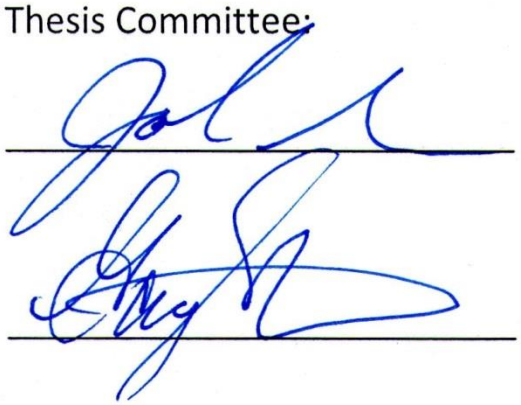
Andrew Paul Simms
University of Arkansas, Fayetteville

This thesis is approved.

Thesis Advisor:



Thesis Committee:



Reading and Wirelessly Sending EEG Signals Using Arduinos and XBee Radios to Control a Robot

An Undergraduate Honors College Thesis

in the

Department of Electrical Engineering
College of Engineering
University of Arkansas
Fayetteville, AR

by

Andrew Paul Simms

April 16, 2014

ABSTRACT

The objective of this project is to develop an EEG device that can read brainwaves from an individual, analyze the data, and use the result to send a wireless signal using Arduinos and XBee Radios to a Boe-bot to perform an action. One of the goals of this project is to read EEG data with a higher sampling frequency than a previously manufactured EEG device. The second part of the project used the device developed to differentiate an individual's thinking between right and left and then send a simple signal to a robot using an XBee radio to perform an action, such as making a motor turn.

The implementation of this project contained three parts. The first part of the project involved making the EEG Data readable by the Arduino by amplifying the signal using instrumentational amplifiers and operational amplifiers as well as a notch filter and a low-pass filter to make the data readable. The second part consisted of interpreting the data that was being received from the EEG probe by filtering it a third time using a band-pass filter from 1 to 31 kHz and plotting it in MATLAB. The software components were data reading, transmission, reception, and processing of the information from the source device to the destination device. The last part of the project was programming the software component of the Arduino such that the wireless devices were able to communicate the EEG information.

Furthermore, the application of this wireless network is far-reaching. Its practical uses cover a wide spectrum, from giving feedback to individuals based on their brainwaves, helping them improve their skilled performance in a game such as golf or archery, to increasing access for those with disabilities to allow individuals to interface with a gaming system using only their brainwaves.

ACKNOWLEDGEMENTS

This research was funded by a grant from the Honors College at the University of Arkansas. I would like to thank Dr. Jingxian Wu, a professor in the University of Arkansas Electrical Engineering Department, for being my mentor and advisor for this project. I would like to thank Dr. Baohua Li, who is also in the University of Arkansas Electrical Engineering Department, for mentoring me in this project and helping me move in the right direction with the research. I also want to thank Dr. Gay Stewart for being my physics advisor and mentor for this project. Lastly, I would like to thank Jayshree Desai, a Graduate student in Electrical Engineering at the University of Arkansas for being an excellent partner for this project.

DEDICATION

This Undergraduate Honors Senior Thesis is dedicated to all of the students and faculty who helped make this project possible.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

A. BACKGROUND AND MOTIVATIONS

Rather than sending signals through many long wires, transmitting a wireless signal is a much easier way to monitor brain activity because it ultimately gives the patient being monitored a full range of motion. Once EEG (Electroencephalography) measurements are made wirelessly, it would also open the door to many more applications related to brainwave monitoring. In a case study conducted by Andrew Callaway, a few novice and professional archers were studied. Their brain waves were measured with an EEG probe while they took a shot with a bow and arrow. The study showed that their skill level was correlated to the EEG patterns of their brains, which means that scientific monitoring of brain signals could help athletes improve their consistency and accuracy in a particular sport [1]. For sports that require a lot of free range of motion, a wireless way to transmit waves using Arduinos and XBee radios from an EEG probe would be the only way that these athletes could perform without any interference from wires connected to the EEG probes.

Interpreting brainwaves also has many more applications such as controlling a computer interface. Imagine if any household item could be controlled strictly from someone thinking a particular way. This would not only make life much easier for the average individual, but it could also give certain people with disabilities a better way to interact in this growing world of technology. Thus, EEG signals can be used for health monitoring and also as a way for humans to interact with computers.

B. OBJECTIVES

The main objective of this project was to control a robot wirelessly using EEG signals from the brain. To accomplish this task, it was necessary to build a circuit that could amplify and

filter an EEG signal, which could then be read by an Arduino microcontroller. Once the signal was read, then a command would be sent from one Arduino to another Arduino using XBee radios to make that robot turn right or left based on the individual's eye movement. A secondary objective of this project was to collect large samples of EEG signals for processing in a machine learning program that a graduate student would be using to determine the accuracy of these EEG measurements.

C. EEG (ELECTROENCEPHALOGRAPHY):

EEG is basically the recording of electrical activity in the brain. Each EEG probe that is used acts as a conductor and sends a current directly from the brain to the amplifier circuit for further modifications. This current can then be translated into something that a computer can read and quantify as a signal, but it must be sent through a very complicated circuit during this process. There are different types and varieties of these EEG probes, but the ones used for this experiment were small tin electrodes as seen in Fig. 1.

Fig. 1. Tin EEG Probes

All parts of the brain have a different function, and the part of the brain used for this experiment was the frontal lobe for its function of movement and location near the scalp. Fig. 2 shows the different sections of the brain and what each section does.

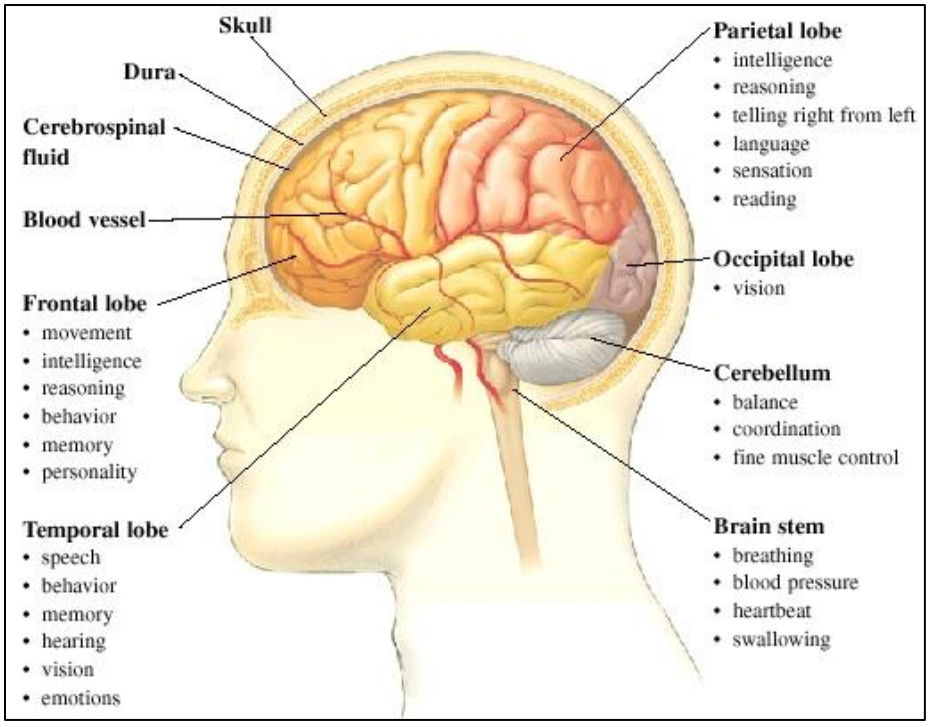


Fig. 2. Regions of the Brain [2]

In order to quantify the data, at least three EEG probes must be used to find a differential voltage across the brain. This is done with a negative EEG probe located on one side of the frontal lobe, a positive EEG probe located on the other side of the frontal lobe, and a ground probe, which is used to ground the system. Fig. 3 illustrates the EEG probe setup when trying to read a signal from the frontal lobe of the brain.

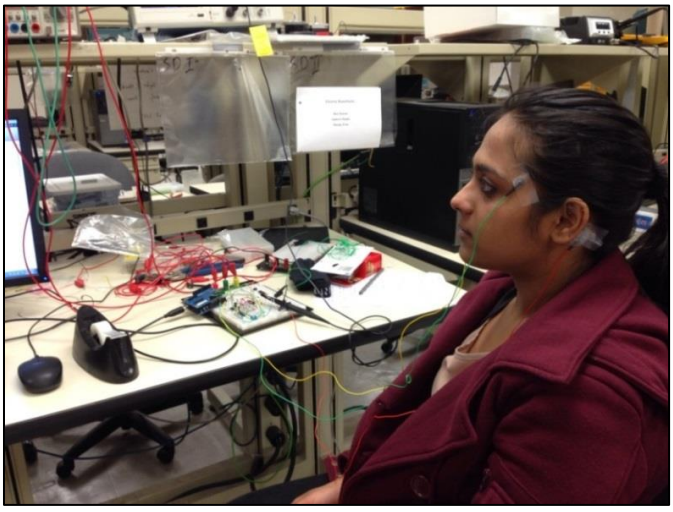


Fig. 3. EEG Probes on Graduate Researcher Jayshree Desai

Humans have only four different types of brainwaves. They can be divided into four categories: theta, delta, alpha, and beta. Each one represents a different state of mind that someone is experiencing at a particular time. Delta waves have a frequency of 3Hz or lower and have the highest amplitude around 50μV. They occur when individuals are in deep sleep. Theta waves have a frequency between 3.5 and 7.5Hz and are characterized by slow-paced activity. Alpha waves are brainwaves between 7.5 and 13Hz, which occurs during relaxation. The last type of waves are beta waves, which have the lowest amplitude and the highest frequency, 14Hz or higher, and are associated with normal waking consciousness. For this project, the waves that were seen the most were beta waves because subjects were awake [3]. Fig. 4 illustrates these brainwaves.

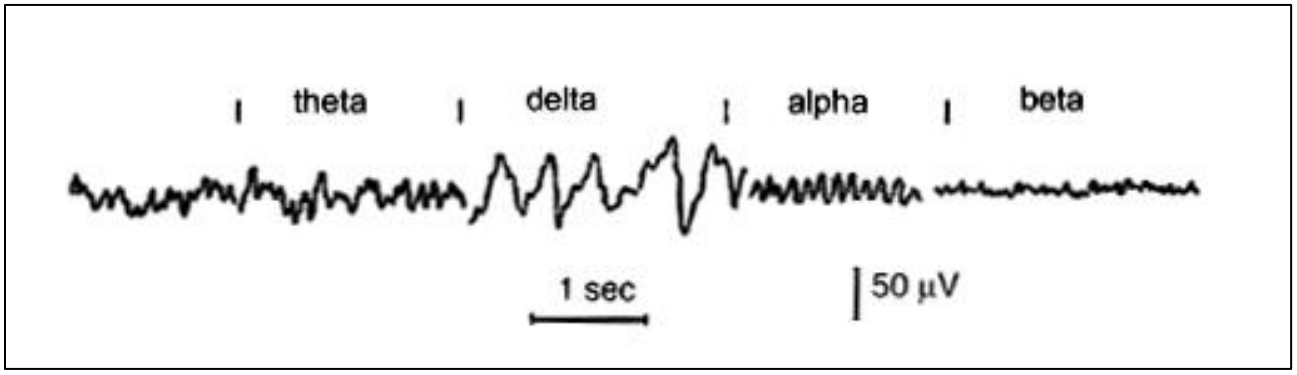


Fig. 4. Four Types of Brainwaves [3]

The three-probe setup treats the brain as an electrical circuit of its own. When the brain receives a stimulus from movement of some kind, such as eye or arm movement, then the signal will instantaneously change rapidly forming a pulse that is clearly visible when the data is graphed. Fig. 5 shows a graph of an EEG measurement with an individual's eye movement.

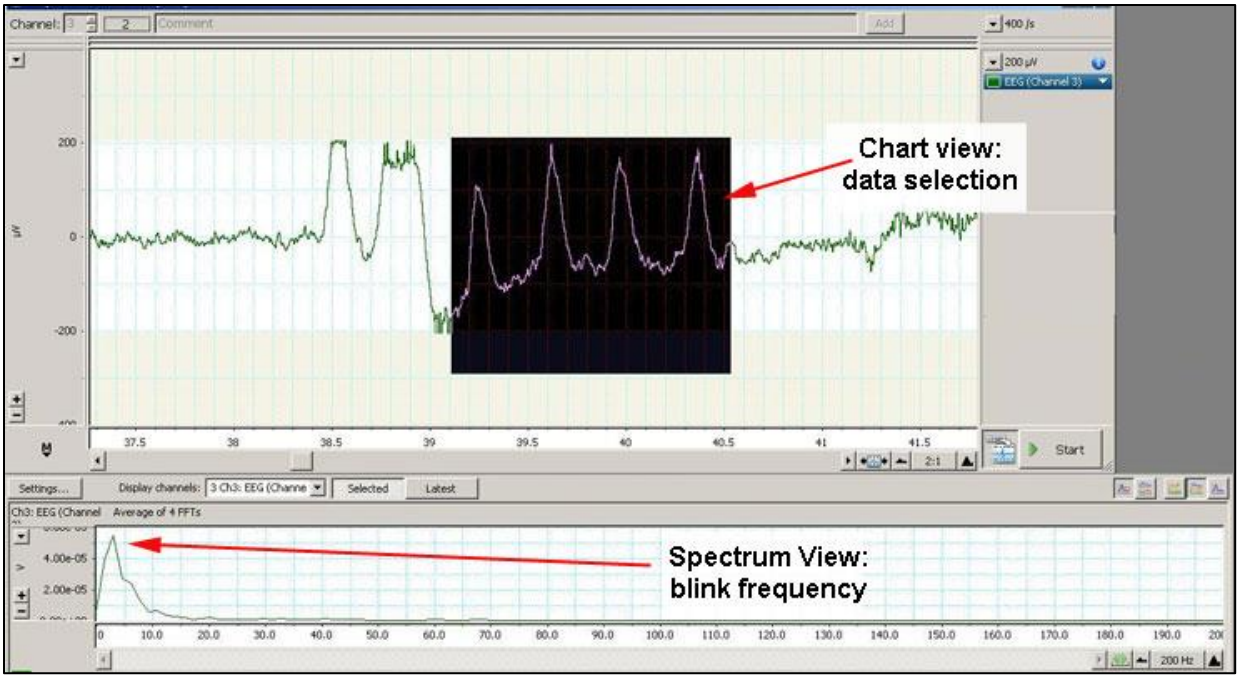


Fig. 5. EEG Signal with Eye Movement Pulses [3]

Typically, brainwave signals have a frequency around 1-20Hz. A signal that is centered at 60Hz is most certainly noise from the circuit, which must be filtered out. This process as well as the amplification of the signal from an individual's brain so that it can be read by a microprocessor requires many steps in the circuit. Once the data from an individual's brain has been analyzed properly, these pulses from the individual's brain can then be used to perform any sort of on/off task, such as telling a robot to move right or left or turning a light on or off.

D. ARDUINOS:

Arduino is a fairly new programming platform that was created in 2005 at the International Design Institute. The software uses a programming language that is very similar to the C programming language, and this hand sized electronic device can easily be programmed from the basic software available on the Arduino website (http://arduino.cc). Fig. 1 is a picture of an Arduino microcontroller that is used to perform certain tasks based on the given input.

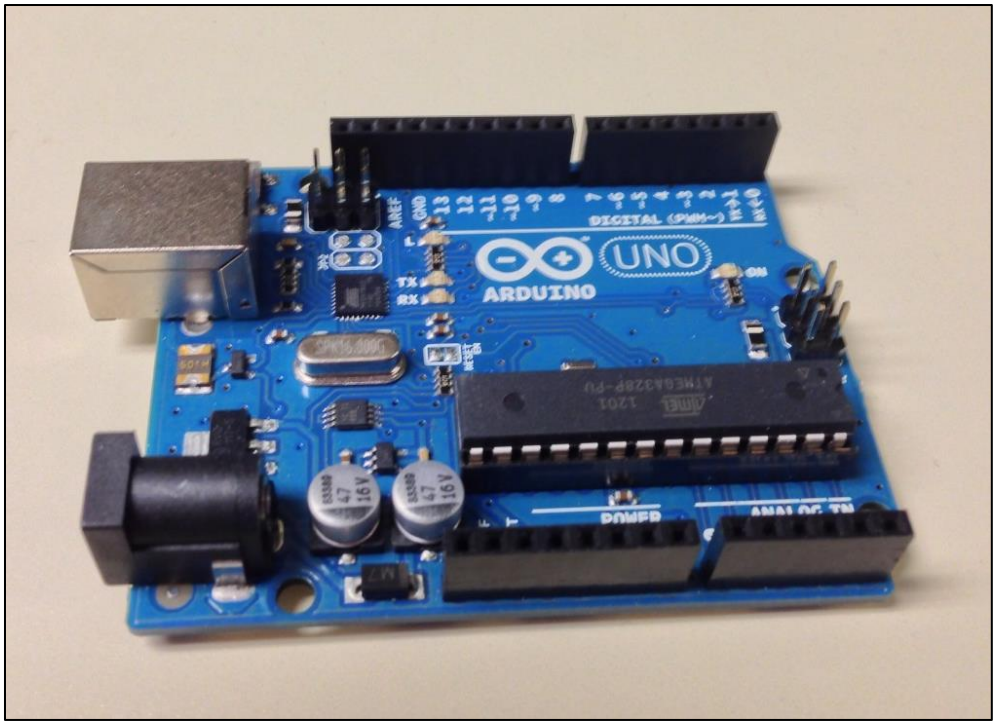


Fig. 6. Arduino Electronic Device

The creation of Arduino has begun a new age of electronics. What makes them unique is that they are affordable and simple to use for anyone with a basic programming and electronic knowledge. In fact, "Arduino has become the most influential open-source hardware movement of its time" [4].

Arduino can be used to perform tasks as easy as turning on and off a light to controlling many types of actuators. Once a program for a particular task is written it is uploaded to the Arduino hardware, which in turn performs the desired task. The tasks can be performed from a few feet away to all the way on the other side of the world via the internet. All of the software is free and requires a basic knowledge of the C programming language and some practice from the examples that are given on the Arduino website [5]. The Arduino hardware is very affordable and requires a basic knowledge of how to put it together from examples that are provided for free online; however, creating a network that performs a particular task can be very time-consuming due to the several components involved in finishing a project.

E. XBEE RADIOS:

XBee radios were used for this project to communicate from one Arduino microprocessor device to another. These XBee are perfect when communicating in a microprocessor environment and are easy to work with because they have commonly used pins, making it easy to use with a breadboard for wiring the required circuits [4]. Fig. 7 is a picture of what one of these XBee radios looks like, including its input and output pins.

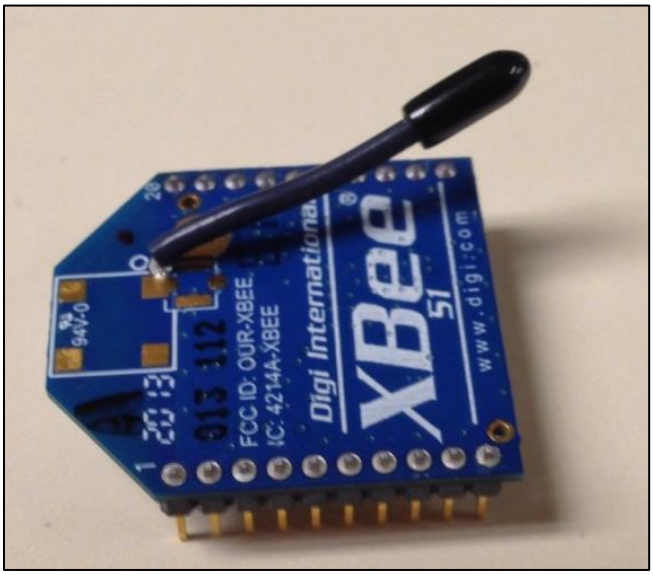


Fig. 7. XBee Radio

This minute radio can be used for a range of applications, which makes it an ideal device to use for this project because of its low cost and versatility. By attaching these XBee radios to each Arduino electronic device, it was possible for all of the devices to communicate with one another by generating and interpreting waves that are sent and received between multiple Arduino electronic devices. Once these radios are attached to each Arduino electronic device, they were used to communicate wirelessly with each other.

In order to use these XBee radios, they must be programmed independently. Two XBees must have the same Pan ID and destination address to send to each other. They must also be operating at the same baud rate, which was a constant 9600 bps. There are other characteristics

of the XBees such as the pins that are used for sending a signal, join notification, and sampling rate, which were all taken into account during the setup process. The main problem that jeopardized the objective of the project was the sampling rate of the XBee radio.

Different XBees have different sampling rates, which is something that was discovered early in the project. Basically, it was found that a series two XBee radio cannot send a signal as fast as a series one. A major goal of this project was to increase the resolution of the data that was being collected and then send it wirelessly to be analyzed at a sampling rate close to 1kHz. The fastest sampling rate capability of the XBee 802.15.4 is 1 kHz, which the series two XBee cannot achieve.

XBees and Arduinos are highly compatible devices, which is the primary reason why they were both chosen for this project. By adding a wireless shield to the Arduino, XBee radios can communicate with the Arduino Microprocessor without blocking any of the input or output pins of the Arduino in a fairly user-friendly setup. Fig. 8 illustrates what the Arduino and XBee setup looks like when they are connected with a wireless shield.

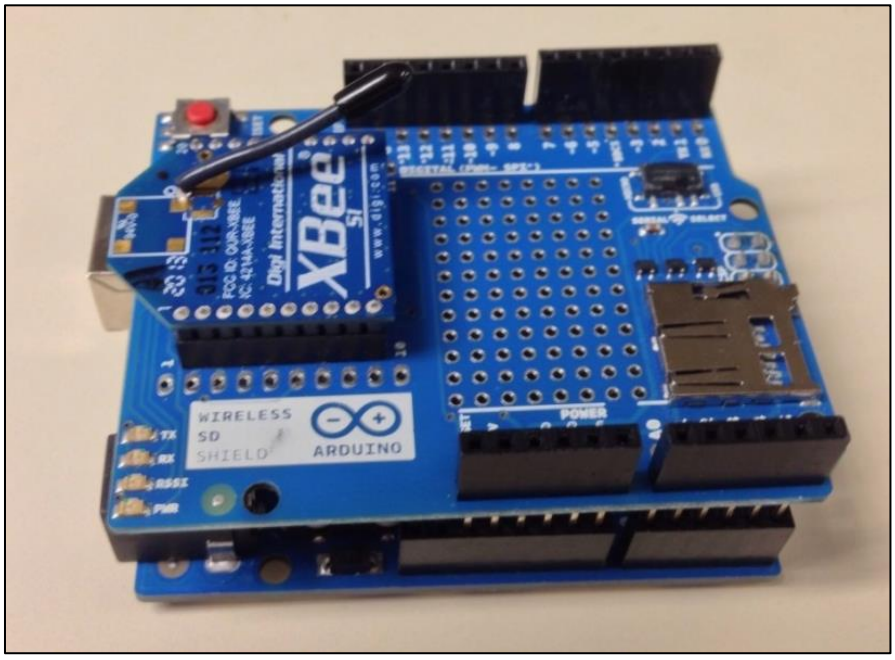


Fig. 8. XBee Radio connected to Arduino by a Wireless Shield

F. THE BOE-BOT:

The Boe-bot has two servo motors connected to a vehicle structure that carries the microcontroller. It is very versatile for all types of uses, but the main application is moving the robot in a desired direction. In most circumstances, the Boe-bot is controlled by the 8051 microcontroller, but for this project, the Boe-bot was controlled by the Arduino Uno microcontroller, which is much more compatible with the XBee radios. Fig. 9 illustrates what the Boe-bot looks like.

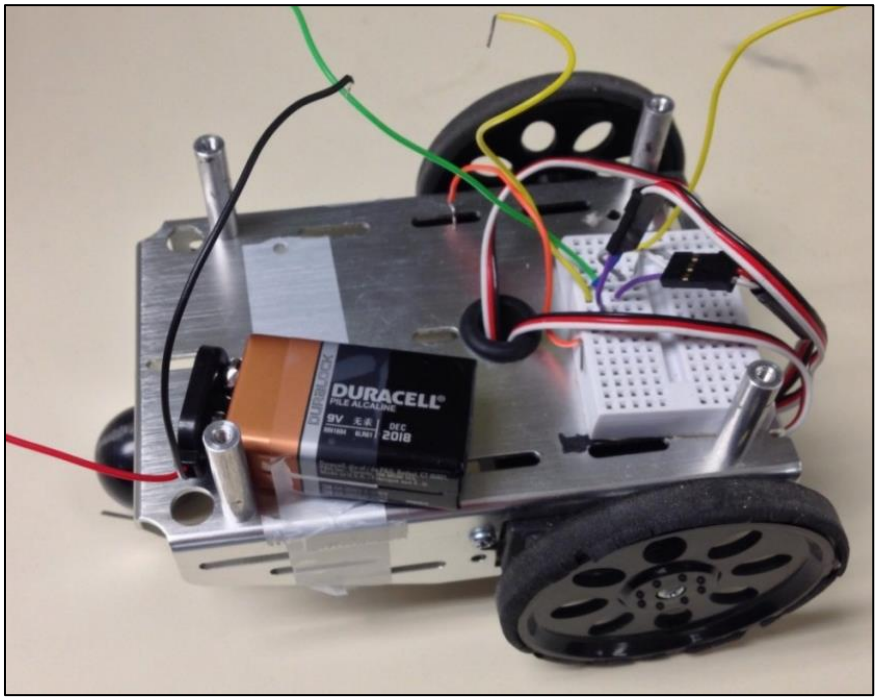


Fig. 9. Boe-bot

In order to control the Boe-bot that was used for this project, it was necessary to send a constant pulse with a specific time delay to the servo motor to tell it to move clockwise, counterclockwise, or stay still. These pulses were created using the Arduino's simple programming interface, and then administered based on the data that it received from the XBee radio transmitting the brainwave data. Basically, when eye movement was high it turned right and when it was low it turned left. If neither of these events occurred, then it would stay still.

II. EEG SENSING SYSTEM DEVELOPMENT

The methods used for this project required two separate parts. The two parts are the software and hardware implementation. For the hardware implementation, an amplifier circuit had to be built, which made data acquisition from the EEG probes possible since EEG signals occur at an extremely low voltage, 20μV, and low frequency, 20Hz. Typically, EEG values are unreadable by the Arduino Microprocessor and the Oscilloscope; however, after the EEG signal passed through the Amplifier Circuit, it could be read by both. Nevertheless, the data had to be amplified, filtered, and offset in order for the Arduino to make a reading.

For the software implementation, the data had to be filtered on MATLAB for analysis due to the large noise that was added by the breadboard. Once this was done, large samples of the data were collected for analysis, and the data also had to be averaged to be interpreted by the Arduino. After the data was averaged, it was analyzed to determine if there was a dramatic change in the data every hundred microseconds, and if so, a command would be sent via series 1 XBee radios to another Arduino. This Arduino would then perform an action based on the data that it received from the first Arduino.

A. MATERIALS

2 x Arduino Unos	1 x 9kΩ resistors
2 x XBee Series 1 Radios (802.15.4)	1 x 4.99kΩ resistors
3 x EEG Probes	1 x 100kΩ resistors
1 x AD620 Instrumentational Amplifier	1 x 10kΩ resistors
1 x UAF42 Notch Filter	1 x 69 nF capacitor
1x Boe-bot	1 x 32 nF capacitor
1 x TL084 Operational Amplifier	1 x 10 uF capacitor

3 x 2kΩ resistors	1 x 1 uF capacitor
2 x 1MΩ resistors	1 x D1N4454 diode
2 x 160kΩ resistors	2 x 2.65MΩ resistors
1 x 12.1kΩ resistors	1 x 12kΩ resistors
2 x 2.65MΩ resistors	1 x 11kΩ resistors

B. READING AN EEG SIGNAL

The first part of collecting the EEG data was building a circuit that included five major sections of components. Each section served an important purpose that helped make the EEG data readable by the Arduino, which has an analog to digital converter range of 0-5 Volts. This range of voltage corresponds to 0-1024, which is the data that is actually read by the Arduino. Each section of the circuit is important for its specialized purpose in filtering, amplifying and offsetting the AC signal that is read from the brain.

Instrumentational Amplifier

The instrumentational amplifier was the first part of the circuit configuration. It takes a differential AC signal, and amplifies it into something that is readable by the system. It is a circuit configuration that consists of three operational amplifiers, but it was decided early on that this circuit configuration would be too difficult to build, so we decided to purchase a pre-built instrumentational amplifier that only requires a resistor to be added to the circuit in order to generate the gain for the circuit. Fig. 10 illustrates the circuit ADC620 instrumentational amplifier that was chosen for this project.

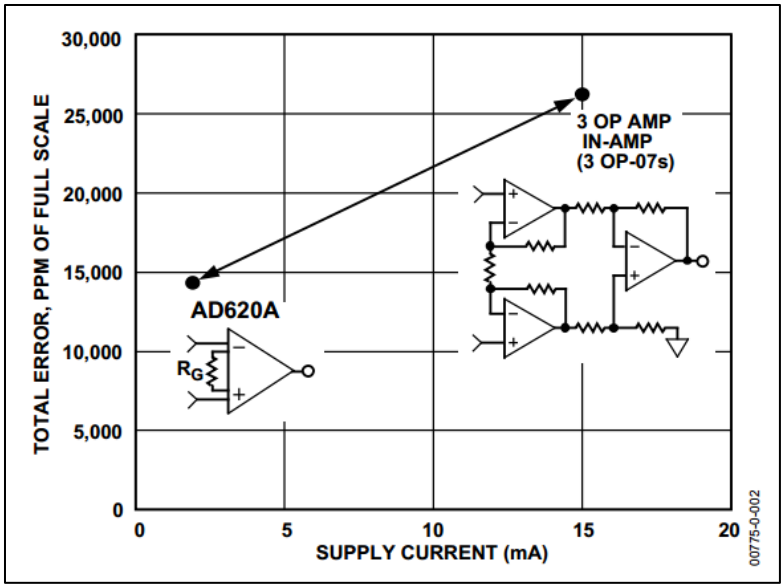


Fig. 10. AD620 – Instrumentational Amplifier [7]

The resistance, R_G, was continually adjusted in order to find the correct gain value that was needed for the circuit. Equation 1 was used to calculate the gain given the resistance.

$$G = \frac{49.4k\Omega}{R_G} + 1 \tag{1}$$

It was decided to use a resistance of 2kΩ, which yielded a gain of 25.7 V/V, precisely the amount of gain that was needed for this particular step. Fig. 11 shows the graph of the differential signal that was measured on the oscilloscope once the EEG signal passed through it.

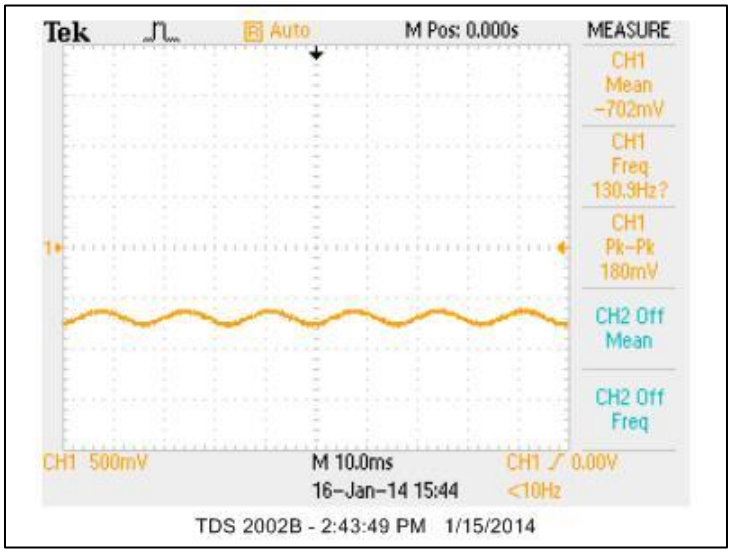


Fig. 11. Graph of EEG Signal after Instrumentational Amplifier

Fig. 11 is a graph of the EEG signal just after it has passed through the instrumentational amplifier. This graph illustrates the high frequency noise primarily from the circuit. The next section of the circuit helps eliminate some of this problem.

60 Hz Notch Filter

The 60 Hertz notch filter is once again a series of operational amplifiers that are used to eliminate any noise right at 60Hz. It was decided that building the 60Hz notch filter would be a tedious undertaking that has no relevance to the objective of the project aside from the reading the EEG signal on the Arduino. It was decided to purchase a UAF42 notch filter, which helped filter the signal from the EEG as soon as it was sent through this portion of the circuit. Fig. 12 illustrates what the notch filter looks like as well as which resistor values to use in order to filter the signal at 60 Hz.

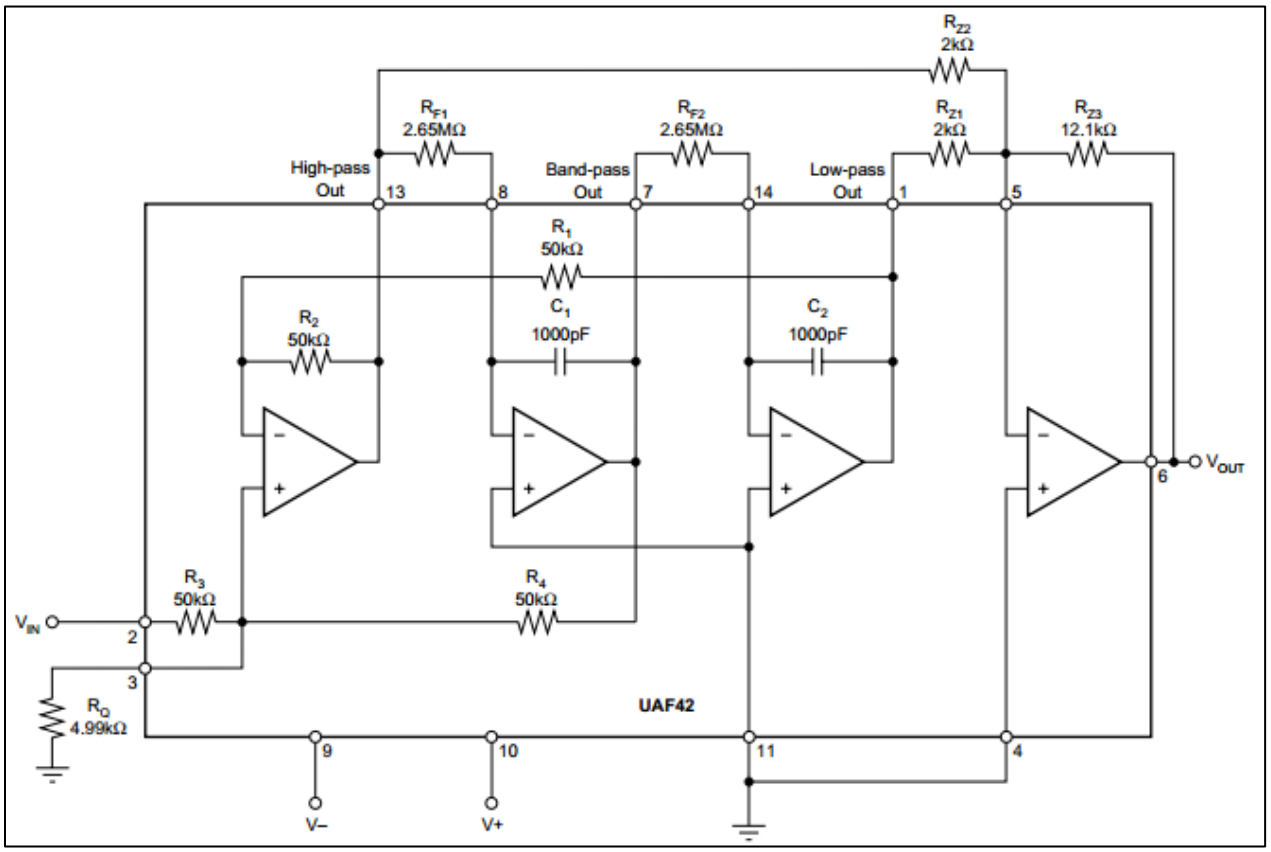


Fig. 12. UAF42 – 60 Hz Notch Filter Design [8]

Fig. 13 shows the response of the 60 Hz notch filter.

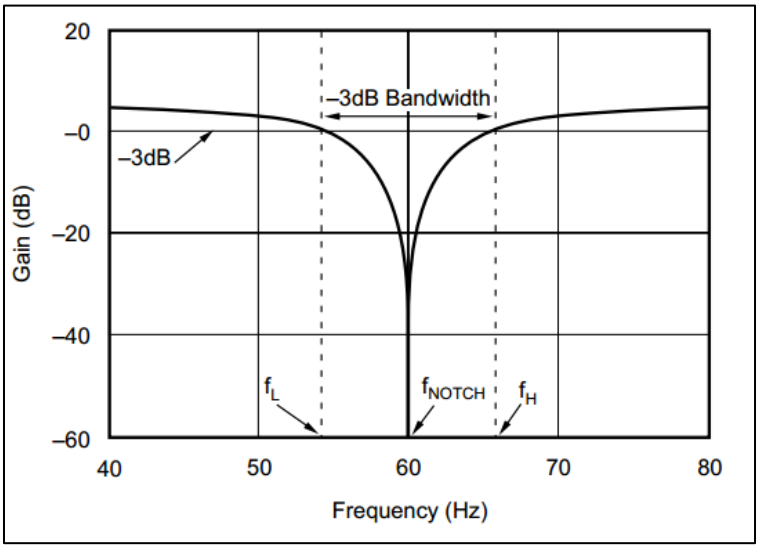


Fig. 13. Notch Filter Design [8]

Any signal passing through this notch filter will automatically lose any 60Hz noise that is generated by the circuit as soon as it passes through this section of the circuit. However, there is still another filter that is needed in order to reduce any high frequency noise. Fig. 14 shows the signal once it has passed through the instrumentational amplifier and the notch filter.

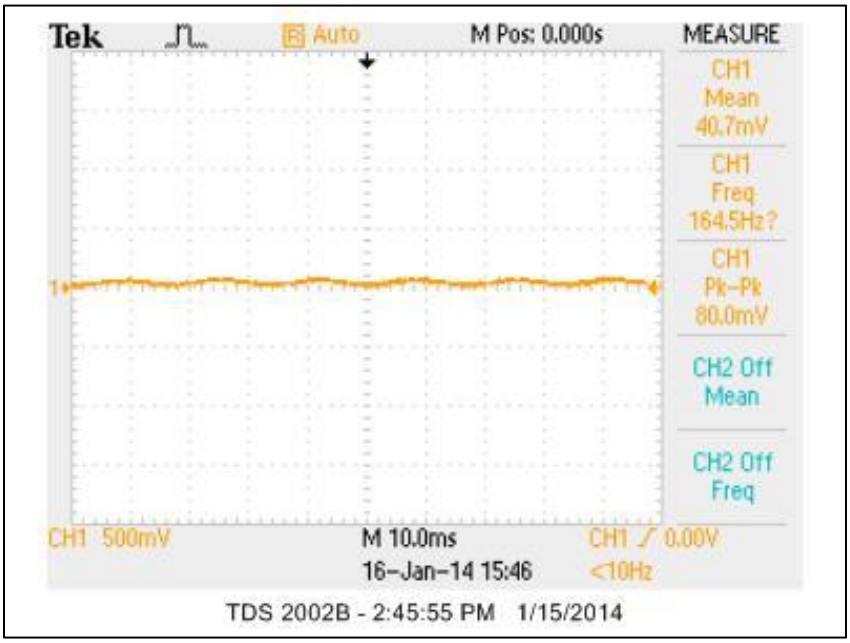


Fig. 14. Graph of EEG Signal after Notch Filter and Instrumentational Amplifier

Low-Pass Filter

The next band-pass filter of the circuit is definitely one of the most crucial. The signal is sent through a 1-31 Hz band-pass filter, which operates almost identically to a low-pass filter. The low-pass filter is used to filter any high frequency that may be present in the circuit. The TL084 operational amplifier was used for the section of the circuit as well as several resistors, which can be seen in Fig. 15.

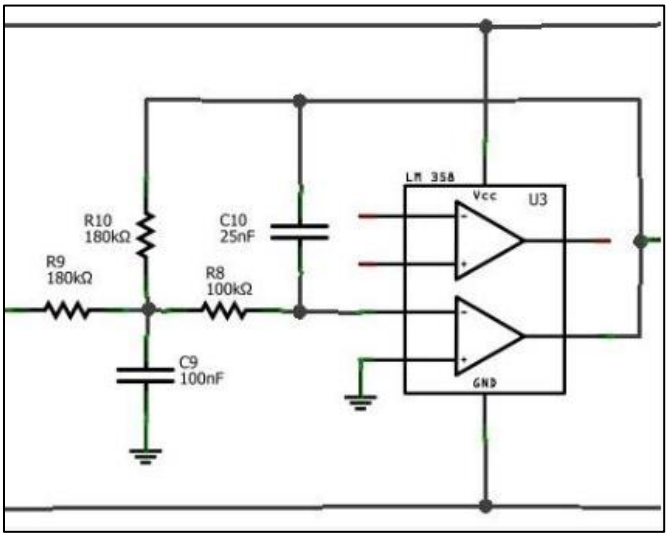


Fig. 15. 31 Hz Low-Pass Filter [9]

After the differential signal was passed through the low-pass filter, the signal began to react to eye and arm movements, which was a very positive result. Basically, whenever an eye would move right the noisy signal would have either a positive or negative pulse depending on the placement of the EEG probes, and the same was true when the eye would movement left only it would be a pulse in the opposite direction. These pulses were significant enough to show that the circuit that was created was already reading the data correctly; however, there were still two more steps to the circuit before the Arduino could read it. Fig. 16 shows the signal after this section of the circuit.

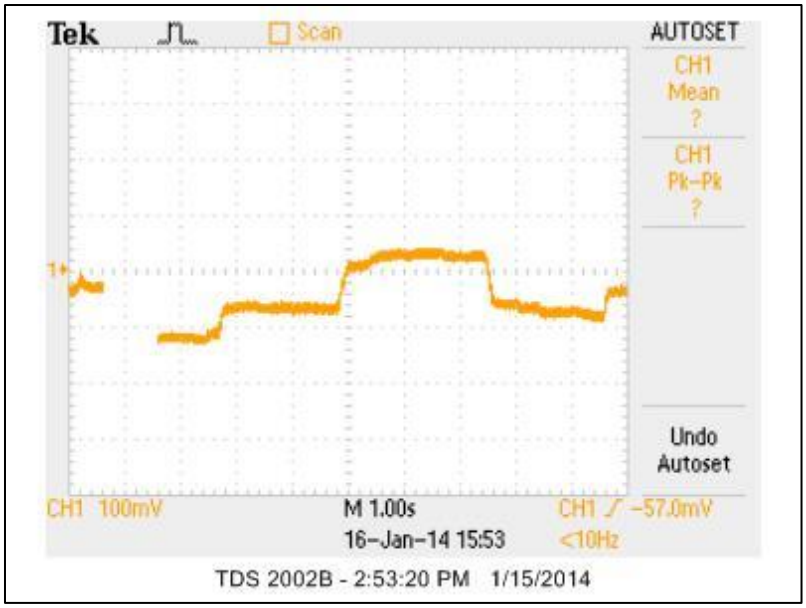


Fig. 16. Graph of EEG Signal after the 31 Hz Low-Pass Filter with Eye Movement

Gain Stage

The Gain Stage section of the circuit was used to adjust the gain of the differential signal between 50-100 V/V using a 10kΩ potentiometer. The same operational amplifier that was used for the low-pass filter was used once again for this section of the circuit. Fig. 17 shows what the configuration for this circuit should look like; however, a few resistor values were adjusted proportionally.

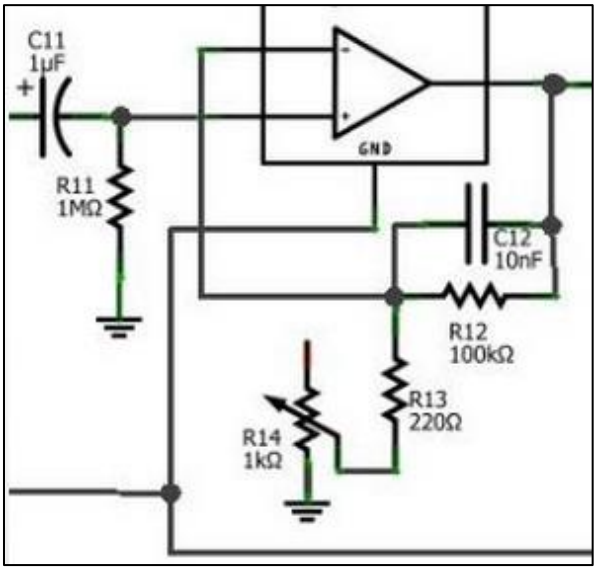


Fig. 17. Gain Stage of the Circuit [9]

After the signal passed through this part of the circuit, it was amplified from an amplitude just above 20mV seen in Fig. 16 to an amplitude of about 4.0V seen in Fig. 18, which in this case was a rough gain of approximately 200 V/V.

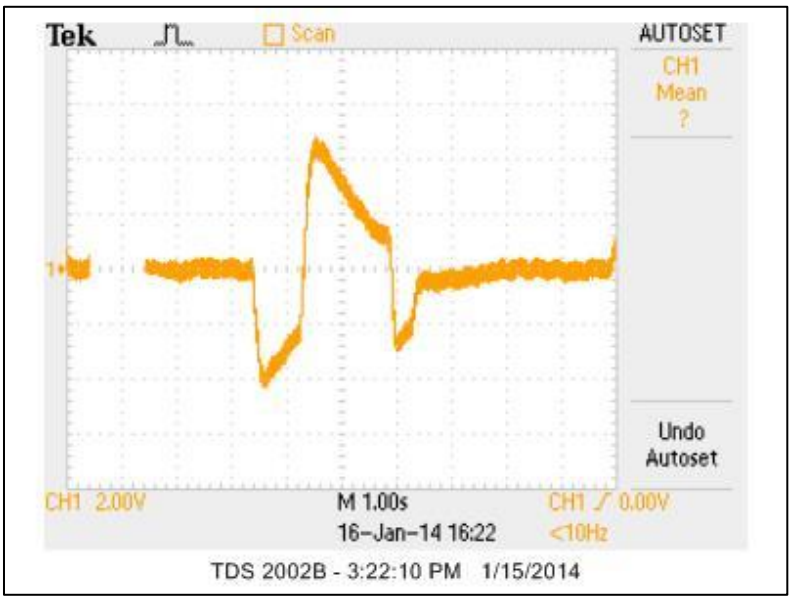


Fig. 18. Graph of EEG after Gain Stage with Eye Movement

After the Gain Stage, a signal with only arm movements was found, which yielded a graph with a slightly smaller amplitude, but still significant. Fig. 19 depicts this graph with one arm moving up and then down.

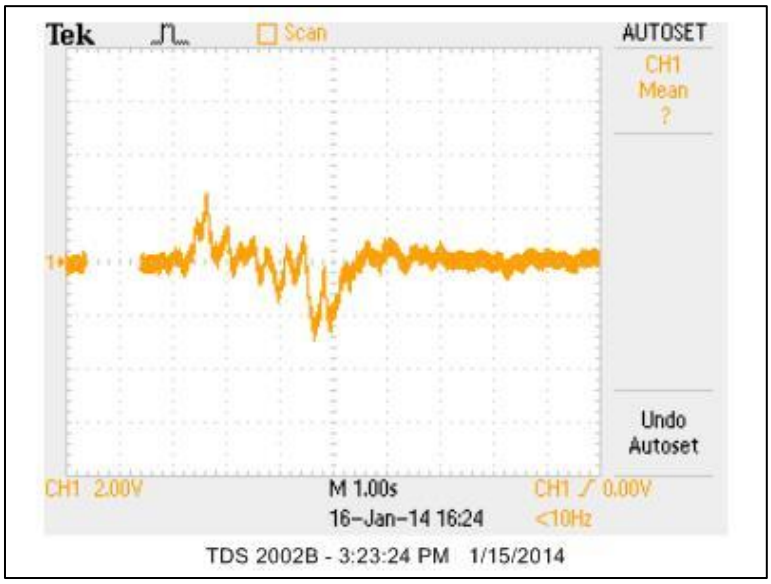


Fig. 19. Graph of EEG after Gain Stage with Arm Movement

Clamper Circuit

The fifth and final addition to the circuit was the clamper circuit, which was added specifically to get a reading for the Arduino. The problem was that the Arduino can only read voltages between 0 and 5V, and the signals kept yielding values that were both above and below zero volts, so a solution was found by creating a clamper circuit that was used to offset the voltage enough that every voltage change on the Arduino could be read accordingly. Fig. 20 shows what the clamper circuit configuration looks like.

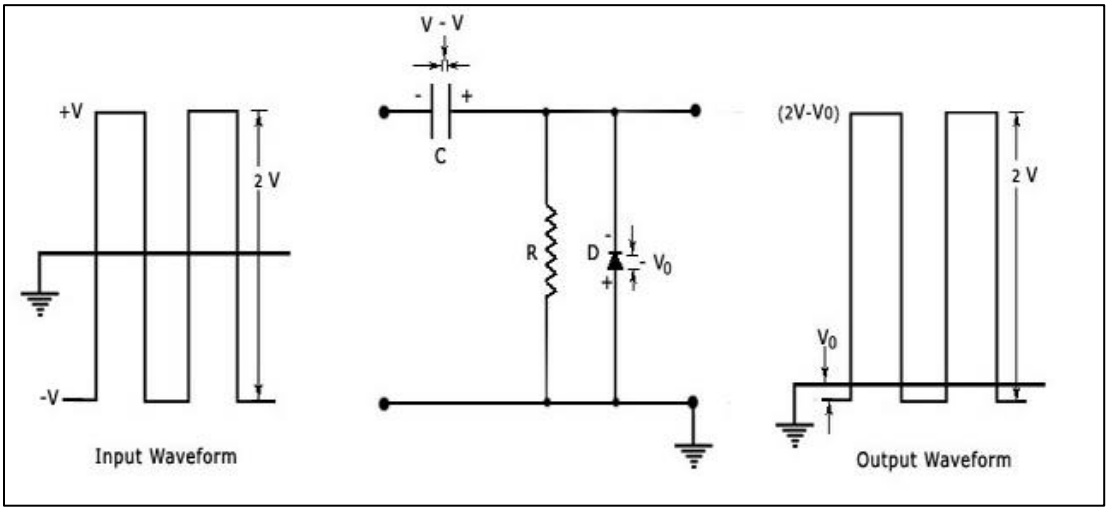


Fig. 20. Clamper Circuit [10]

In order to make the circuit offset just enough to get it above 0 Volts yet stay below 5V. This is accomplished by offsetting the signal by 3.3V from the Arduino output. Fig. 21 shows the comparison when offsetting the circuit by 3.3V.

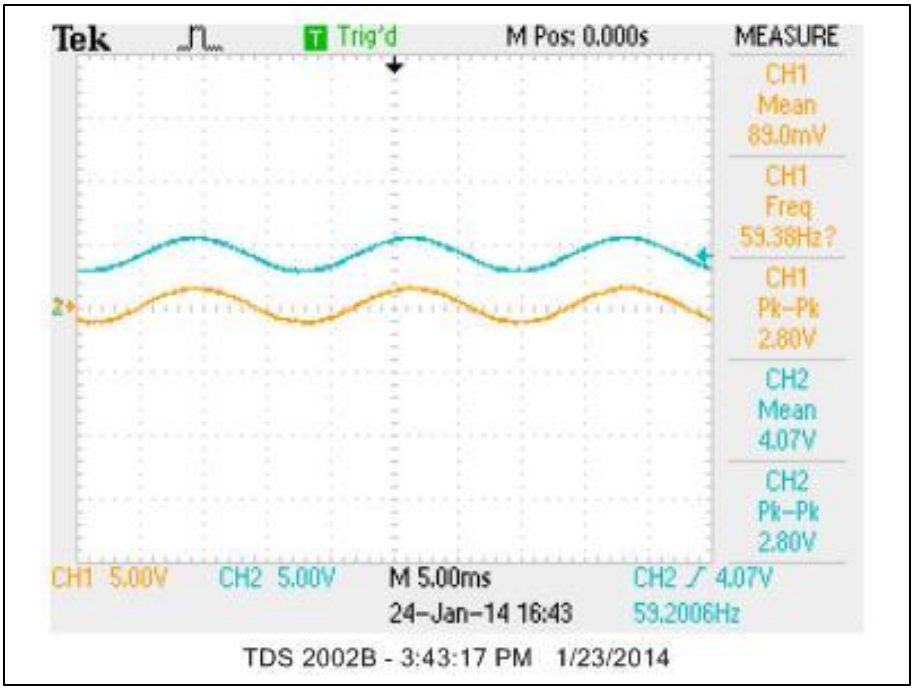


Fig. 21. Offsetting Sinusoidal Signal by 3.3V

Fig. 22 shows the final readings that were output to the Arduino after the EEG signal passes through all five sections of the circuit. The signal can be read by the Arduino and shows the eye and arm movements for the project.

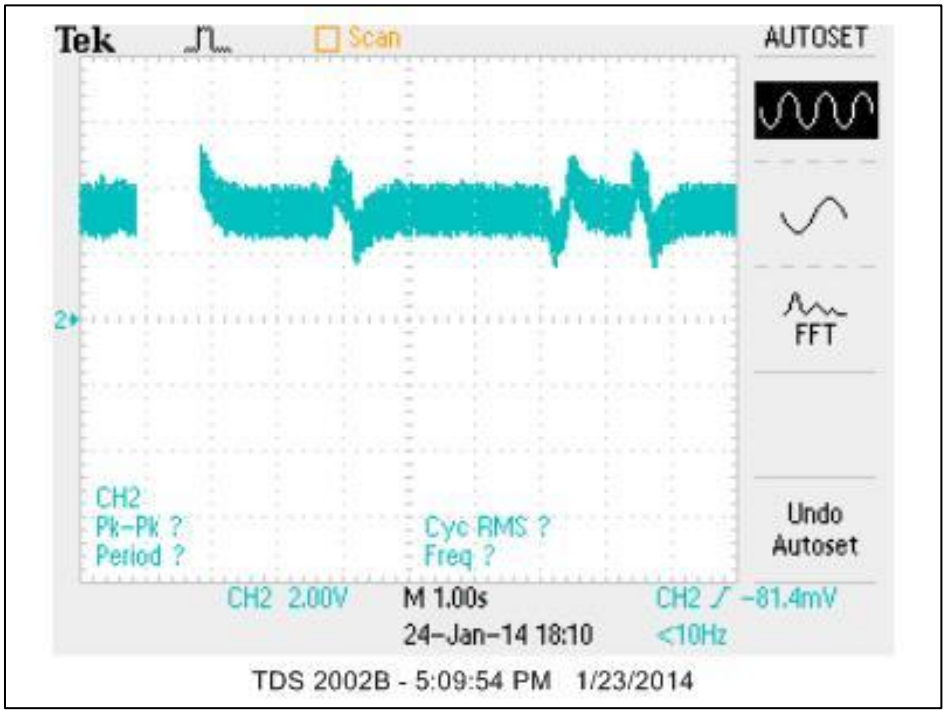


Fig. 22. Graph of EEG after Clamper Circuit Offset with Eye Movement

III. EEG SIGNAL PROCESSING DEVELOPMENT

A. FILTERING SIGNAL ON MATLAB

As previously indicated, the EEG signal from the brain has passed through five sections of the circuit. To see each response for each section of the circuit individually, see Fig. 11, 14, 16, 18, and 22. There is no graph for the original EEG signal, which is so small that it is on the microvolts scale. The oscilloscope cannot even distinguish from the noise, which is why it was necessary to pass the signal through the circuit. Once the Arduino was able to read the differential signal seen in Fig. 22, the next step of the project was reading that data at a very high sampling rate on the Arduino. The goal of the project was to obtain a 1kHz sampling rate. The sample size during each reading of the EEG data during testing was 10,000 samples, which would make the resolution of the data extremely high. Due to the capabilities of the Arduino, the fastest that the Arduino can collect all of this data was approximately 12 seconds. To reach the desired goal of 1 kHz, the 10,000 data points would need to be collected in 10 seconds or less. However, obtaining 10,000 data samples in 12 seconds did not have a dramatic impact on the overall accuracy of the data collection.

The data was sampled as fast as possible and yielded an average sampling rate of approximately 837Hz, which is only 163Hz shy of the 1 kHz desired goal. The data was stored on a text file in approximately 12 second intervals using the interface hyperterminal to collect the data. Fig. 22 shows the reaction to eye and arm movements and the data after the circuit, but it is not a typical EEG signal such as in Fig. 4.

The signal that was read in Fig. 22 obtained more of that 60Hz noise that was present in the breadboard, so it was necessary to filter the data again; however, this time the data was filtered on MATLAB using a 31Hz low-pass filter, which eliminated any noise above 31Hz. This

filter was used and yielded data that looked much more like what was expected. The amplitude of the signal was adjusted by approximating the gain through the circuit. First, the Arduino reads an ADC input, which is in a range from 0-1024, which corresponds to the range of 0-5V, so the data received must be divided by 1024 and multiplied by 5. Secondly, the instrumentational amplifier adds its own gain to the system, which was found to be approximately 25.7V/V by calculating the gain using equation 1 and an RG value of 2kΩ. For the final gain value, the gain from the instrumentational amplifier to the final output of the system was found by finding the ratio of the peak-to-peak values from the output of the instrumentational amplifier to the output of the entire system before the 3.3V offset. The value was found to be approximately 200V/V. The data was multiplied by a gain factor to yield an approximation of the actual EEG signal. Also, because the data had been offset, the data was shifted down by subtracting by the mean of the data to get rid of this offset of the data. Fig. 23 shows a graph of both the unfiltered and filtered data, of the EEG signal associated with eye movements over a ten second period.

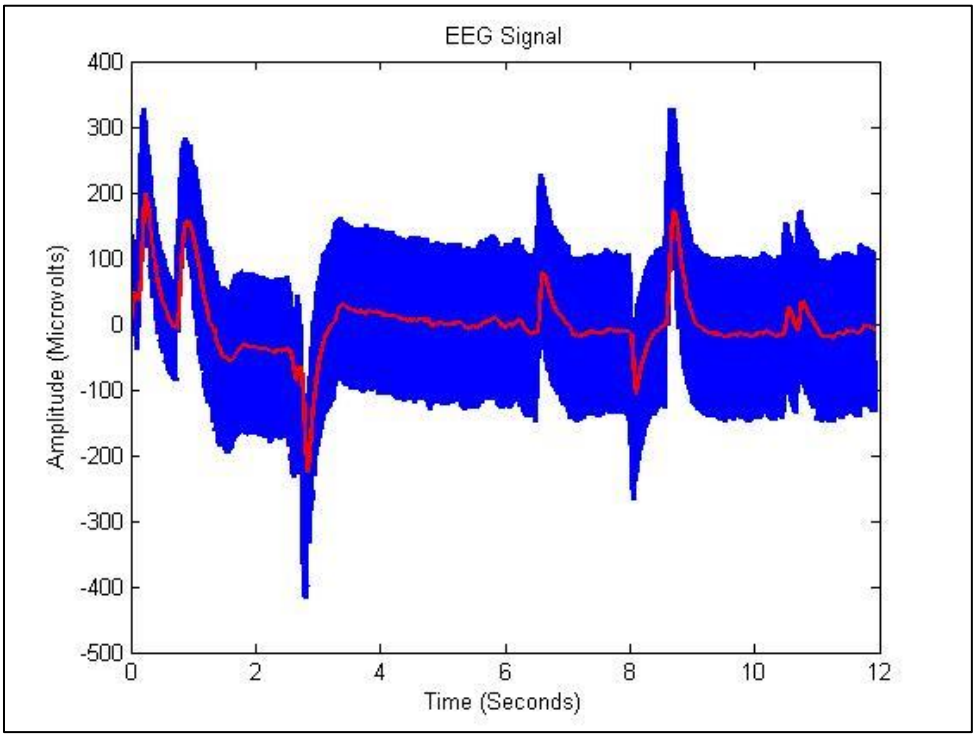


Fig. 23. Graph of Unfiltered and Filtered EEG Signal with Eye Movement

The red line, the filtered signal, shares a similar periodic pattern with the signals seen in Fig. 4. The accuracy of this signal to a regular EEG signal was crucial to the project and why the project was performed in the first place. The amplitude for this data was as high as 200µV, but that was the amplitude for eye movement from right to left. A graph of the data sampled without any eye movement yielded a result that was more consistent with what a typical EEG signal looks like as shown in Fig. 24.

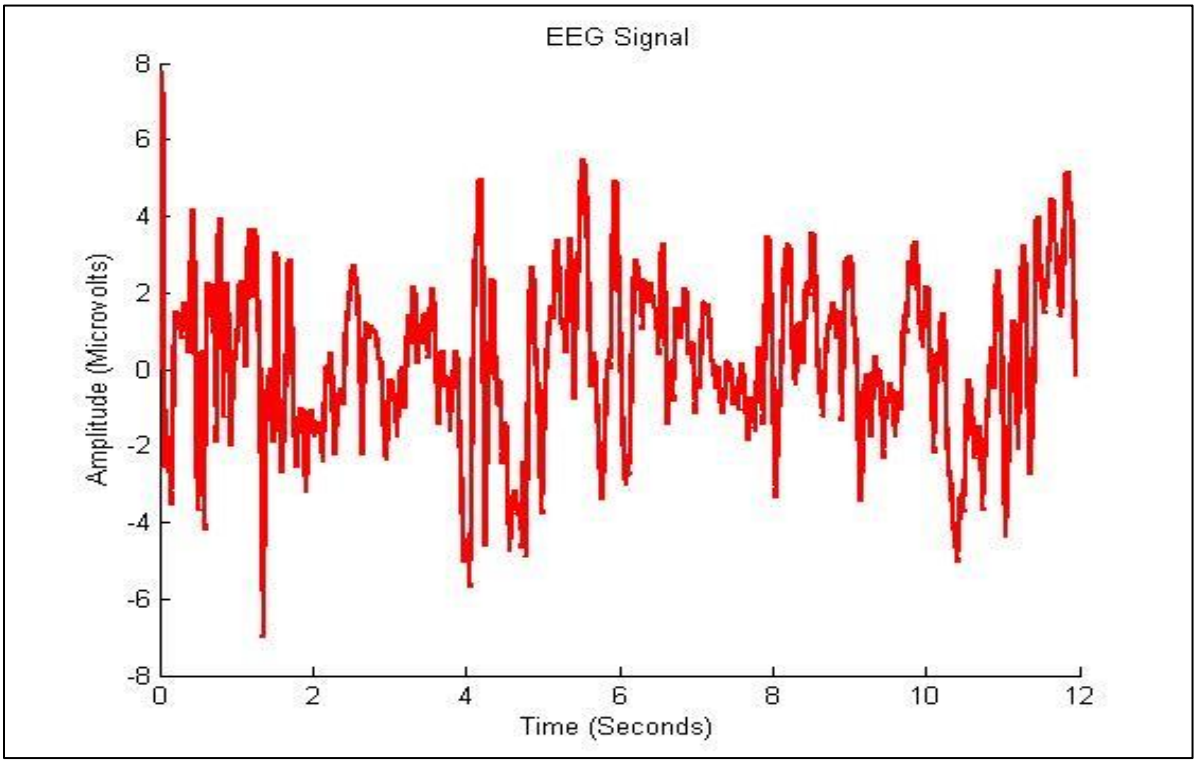


Fig. 24. Graph of Filtered Data with no Eye Movement

B. CONFIGURING XBEE RADIOS

The original XBee radios that were used for this project were XBee Zigbee Series 2 Radios. They were a nightmare to configure because they would not communicate with the X-CTU software that was downloaded to program these radios. It was soon decided that in order to send a simple signal from one Arduino to another, it would be necessary to order XBee Series 1 (802.15.4) Radios to complete this project. When those were ordered, the process became much

simpler. The XBee radios were configured on the X-CTU interface with relative ease. Fig. 25 shows what the X-CTU interface settings looked like for this project [11].

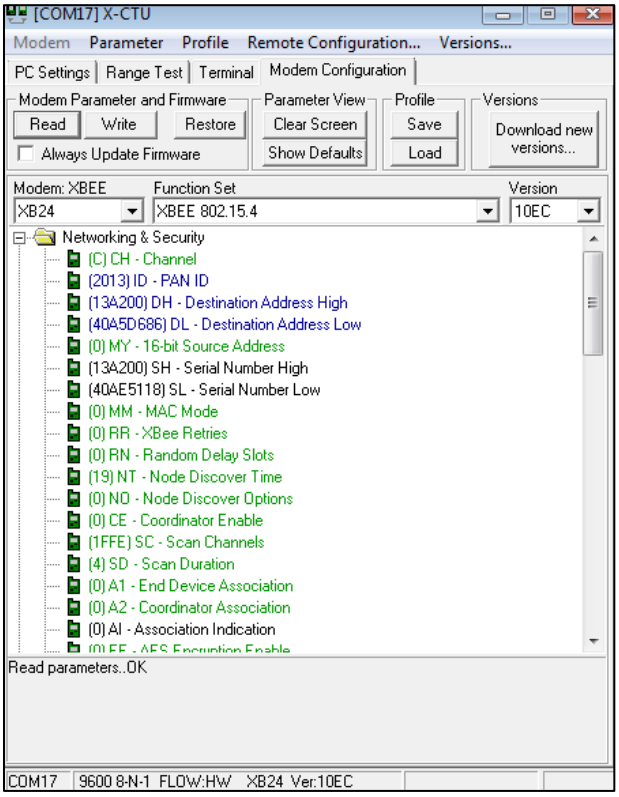


Fig. 25. X-CTU Settings for Coordinator XBee

C. WRITING ARDUINO CODE

Writing the Arduino Code for the Project was a difficult task because numbers had to be modified to provide the utmost efficiency and to account for the noise from the signal when determining which command to make. The Arduino code was written to read the code as fast as possible in order to increase the sampling rate of the system for the data acquisition part off the project. For the second part of the project to control a robot using EEG signals, the data was averaged over 20ms and compared to data that it had read over the preceding approximately 20ms to determine whether to proceed or not. If the data difference was above a certain level, then the program would average data for the next 160ms to determine whether the change in the signal was positive or negative. If the data difference was negative, then the signal would

command the Boe-bot to turn left; if it was positive, then the signal would command the Boe-bot to turn right. It was not greater than a certain amount, then the program would send a Null command to account for noise. The program would then wait approximately 300ms before trying to collect data again. This process would repeat continuously and occur in a single second or however long it took to pick up a reading. The Reading EEG Signal/Sending Command Arduino Code can be found in Appendix A, and the Boe-Bot Control/Receiving Command Code can be found in Appendix B.

IV. EXPERIMENTAL RESULTS

The results from reading the EEG data on the Arduino were very promising. They showed how well the experiment was performed as well as the accuracy of the circuit that was built to collect EEG data. The data from the Arduino had to be filtered to ensure the elimination of any noise that may have been added to the signal in the circuit, and the EEG signal had to be amplified, but overall the EEG signal was collected successfully with a high accuracy of predicting whether an individual was thinking right or left.

A. ANALYZING AN EEG SIGNAL

The process of analyzing this EEG data was done with large quantities of similar data. The data was collected from an individual thinking right 250 times, and then this process was repeated with the individual thinking left. Because large amounts of data were collected, the sample size was reduced to 3000, and the sample time was reduced to 3.5 seconds; however, the sampling rate did not change from 837 Hz. The goal of this data sampling was to determine if a pattern would emerge when an individual is thinking right, as this would help to predict whether an individual was thinking right or left. Fig. 26 shows four graphs of data where the individual was thinking right, and Fig. 27 shows four graphs of data where the individual was thinking left.

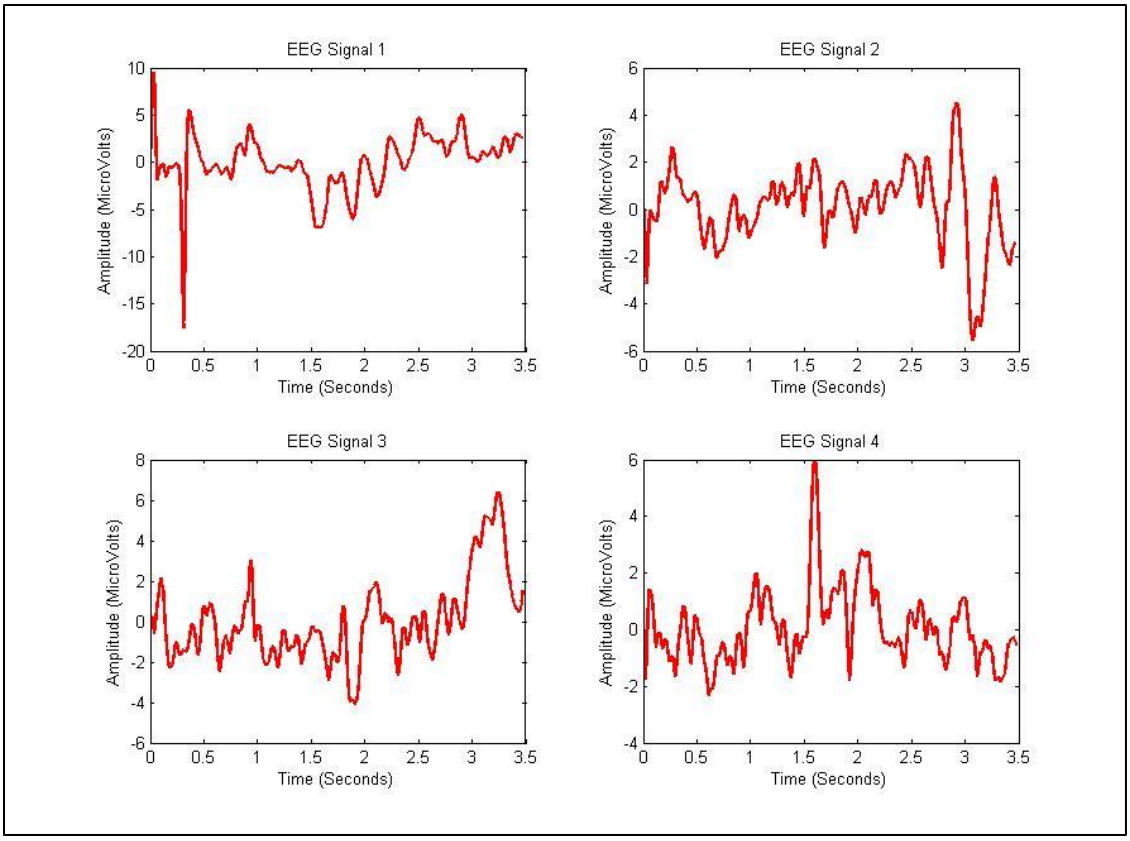


Fig. 26. Graph of Filtered Data Thinking Right

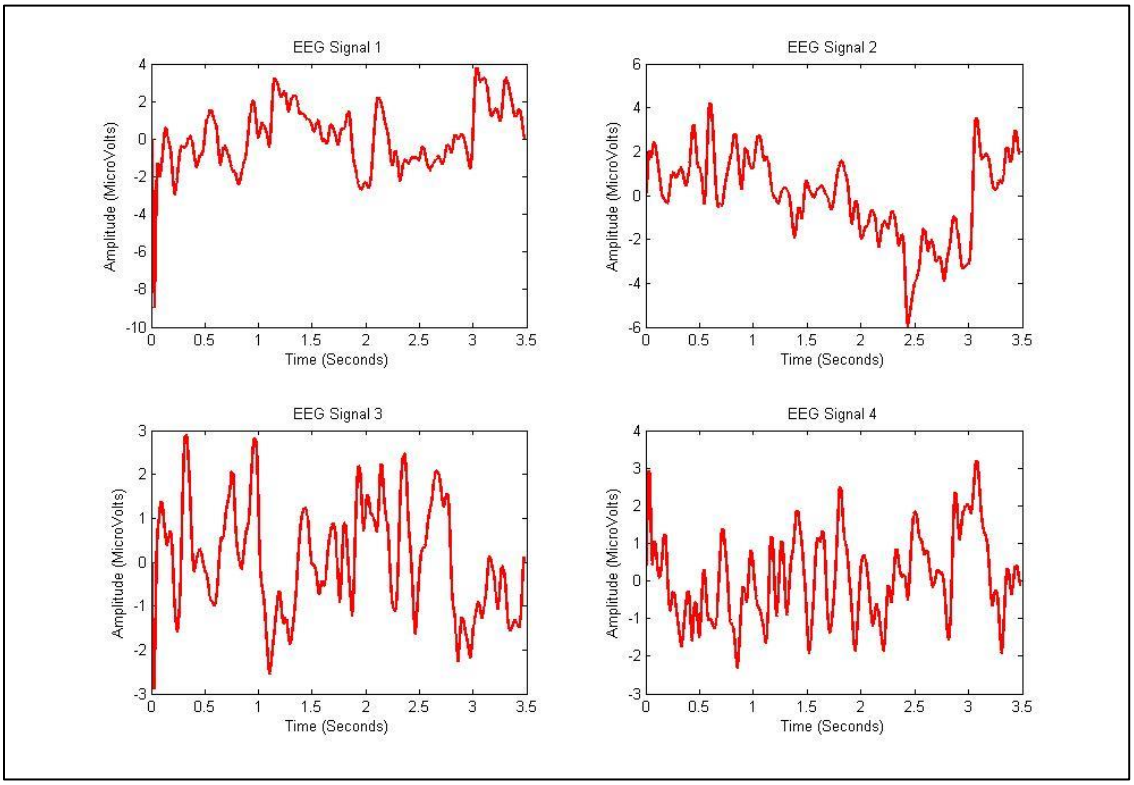


Fig. 27. Graph of Filtered Data Thinking Left

After the data was collected, the graduate student working on this project ran a test of the data to determine the level of accuracy of the data. 150 samples of thinking right and left were used to teach the machine learning program the patterns between a signal where an individual is thinking right or left. Then, the other 100 samples of data of thinking right and left were processed by the machine learning program to determine the type of signal. The machine learning program had an accuracy of predictability of 96% for thinking right, and 97% for thinking left, which is a great improvement from the previous readings that only had approximately 80% accuracy.

B. CONTROLLING A BOE-BOT WIRELESSLY

The main result of this project was controlling a Boe-bot wirelessly using EEG signals. A system was configured in order for two Arduinos to process information and communicate with each other using two XBee radios. The result of the project was a successful trial where the Boe-bot was controlled only by eye movement multiple times.

The Amplifier Circuit Configuration

The first part of the data acquisition had the user connect three EEG probes to the head as seen in Fig. 3. The input EEG signal would pass through the circuit that was built for this project and result in an output that could be read by the Arduino analog input. This output would be processed and would prompt a command to be sent from the first Arduino-XBee configuration, which can be seen in Fig. 8, to the second Arduino-XBee configuration that was connected to the Boe-bot. Fig. 28 shows the Amplifier Circuit that was built for this project.

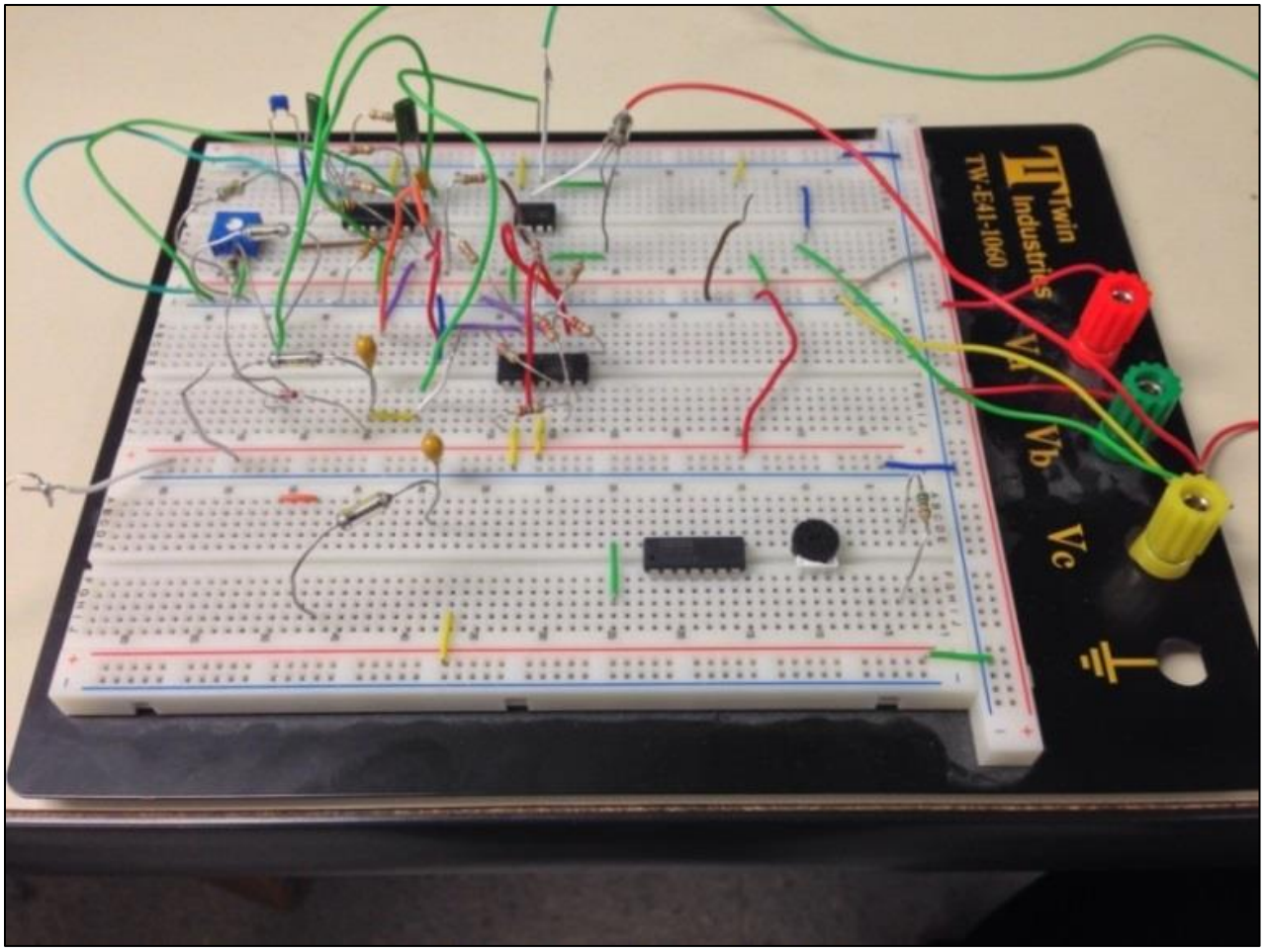


Fig. 28. EEG Amplifier Circuit

The Boe-bot Configuration

The Boe-bot was powered by a standard 9V battery, so that it had nothing connected to it and the breadboard. The Arduino-XBee configuration was connected to the Boe-bot with wires, and acted like the brain of this device. The Boe-bot would perform an action whenever it received a command from the other Arduino-XBee configuration, which was continuously reading the analog signal from the amplifier circuit that was created from the EEG reading of the individual's brainwaves. Fig. 29 illustrates what the Boe-bot configuration looked like.

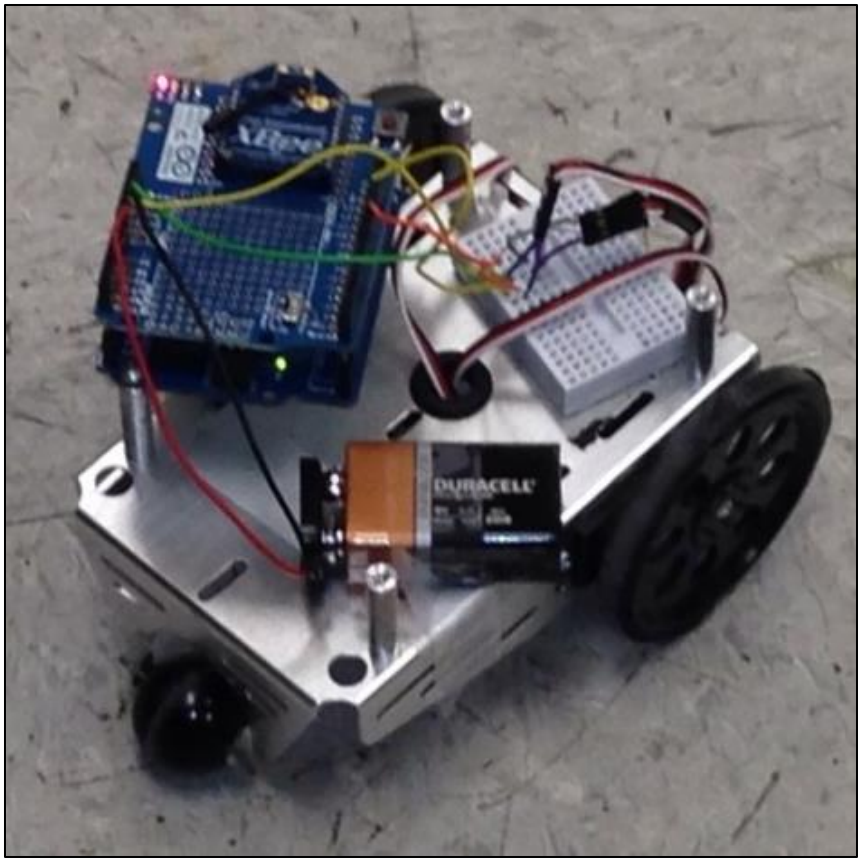


Fig. 29. Boe-bot Setup to receive a command based on EEG Signal

Combining the Two Systems

The Boe-bot was controlled by using eye movements to make the Boe-bot move in the direction that the eyes are looking. Therefore, if someone looks right, the Boe-bot would turn right; and if he or she looks left, the Boe-bot would turn left. In order to control the Boe-bot, it was necessary to find a mean of the data every few seconds and compare it to the mean of the data after another few seconds to see if there was a dramatic change in the data in either the positive or negative direction. If there was a notable change, then that would indicate whether the individual looked right or left, and the command would be sent to the Boe-bot to perform a specific action. The problem that kept arising with reading directly from the Arduino was that the signal was very noisy, which can be fixed on MATLAB, but the objective of the project was to send the signal between two Arduinos only. Arduinos do not have a filtering program as

sophisticated as the one on MATLAB, so an algorithm was written on the Arduino interface to solve this problem. The Arduino algorithm worked and the Boe-bot turned in the direction that the eyes were moving, but only when there was not a large amount of ambient noise from the circuit.

V. CONCLUSION

Overall, the objective of the project was accomplished successfully. The project required that an EEG signal be read and processed so that a command could be sent successfully to a robot to perform an action. The hardware implementation of the data was difficult, but it was completed successfully. After amplifying, filtering, and offsetting the EEG signal, the Arduino successfully read the analog data and output its readings to the computer. This allowed the computer to collect several hundred samples of EEG data for processing. After that, the signal was filtered and then tested to see if it accurately represented the brainwaves of a human being.

The results of the EEG analysis were also very encouraging. Not only did the EEG data get collected successfully, but it was also very accurate in predicting whether someone was thinking right or left. The accuracy of the EEG data turned out to be 97% accuracy for thinking left and 96% accuracy for thinking right. This accuracy was larger than 95%, which indicates that the inaccuracy of the data was less than the alpha value of 5% and therefore there is strong evidence that the EEG Data collected accurately corresponded to the actual EEG data for the brain.

Finally, the software implementation to control the Boe-bot with wireless commands was also successful. When the noise of the circuit was minimized the Arduino would interpret the eye movement of an individual and turn right upon the eyes moving right or left upon the eyes moving left. The robot demonstrated one of the many applications that wireless EEG signals

have in today's world. Overall, the data was successfully collected, analyzed and used to make a robot move, thus accomplishing the goal of this project.

VI. FUTURE RESEARCH

There are several directions that this project could go in the future. One of the prime applications for sending an EEG signal wirelessly is giving athletes a much freer range of motion while EEG data is being sampled. The application of this wireless EEG measuring device would allow scientists to measure brain activity for sports such as soccer, football, baseball, etc. Furthermore, this project provided a greater understanding for everyone involved on how to use the Arduino equipment, how to interpret EEG measurements, how to make these devices communicate on a wireless network using XBee radios, and how to write, compile and implement these Arduino programs outside of a class setting. One way to insure the continuation of this project is creating a PCB of the Amplifier Circuit that was created for this project to spare future researchers the task of rebuilding what has already been created.

The Amplifier Circuit that was created for this project was created on a PCB board for future research. The PSPICE Schematic of the Amplifier Circuit can be seen in Fig. 30.

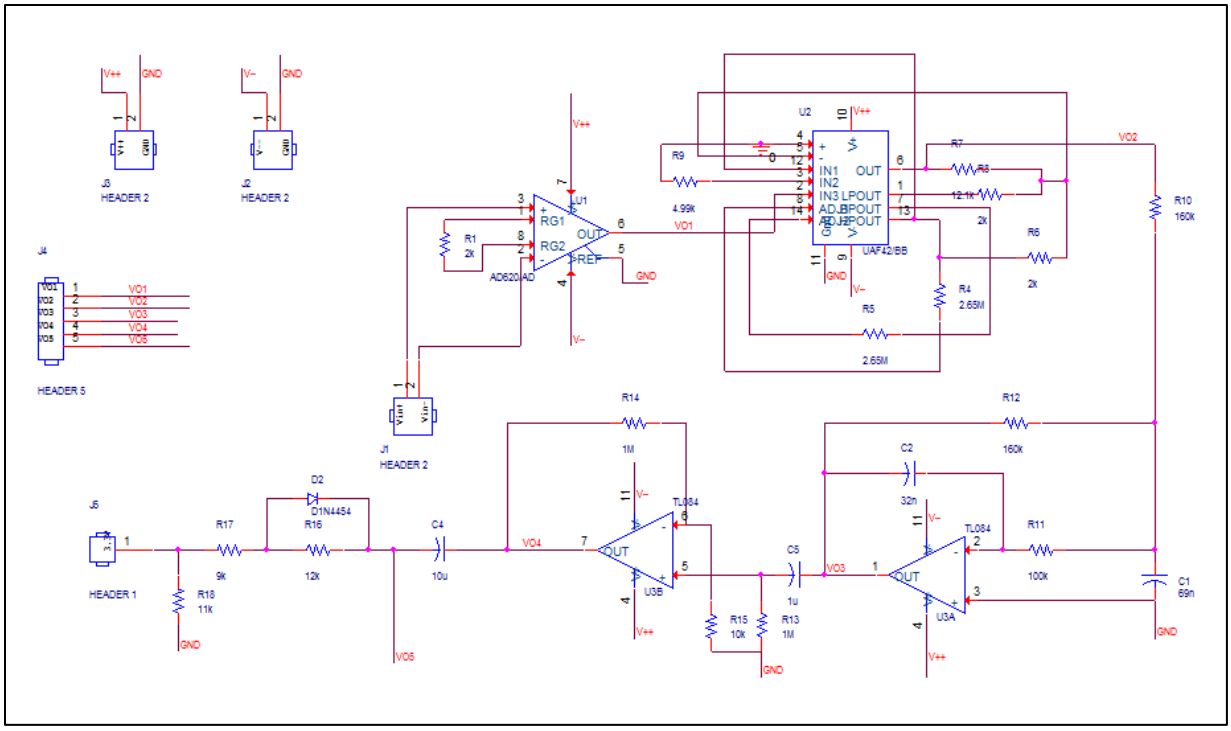


Fig. 30. PSPICE Schematic of Amplifier Circuit

Netlisting of the PSPICE schematic was used to configure the Amplifier Circuit on the Allegro PCB editor program, which created the design for the circuit, so that it could be created using the milling machine. Fig. 31 shows the PCB Editor schematic of the Amplifier Circuit.

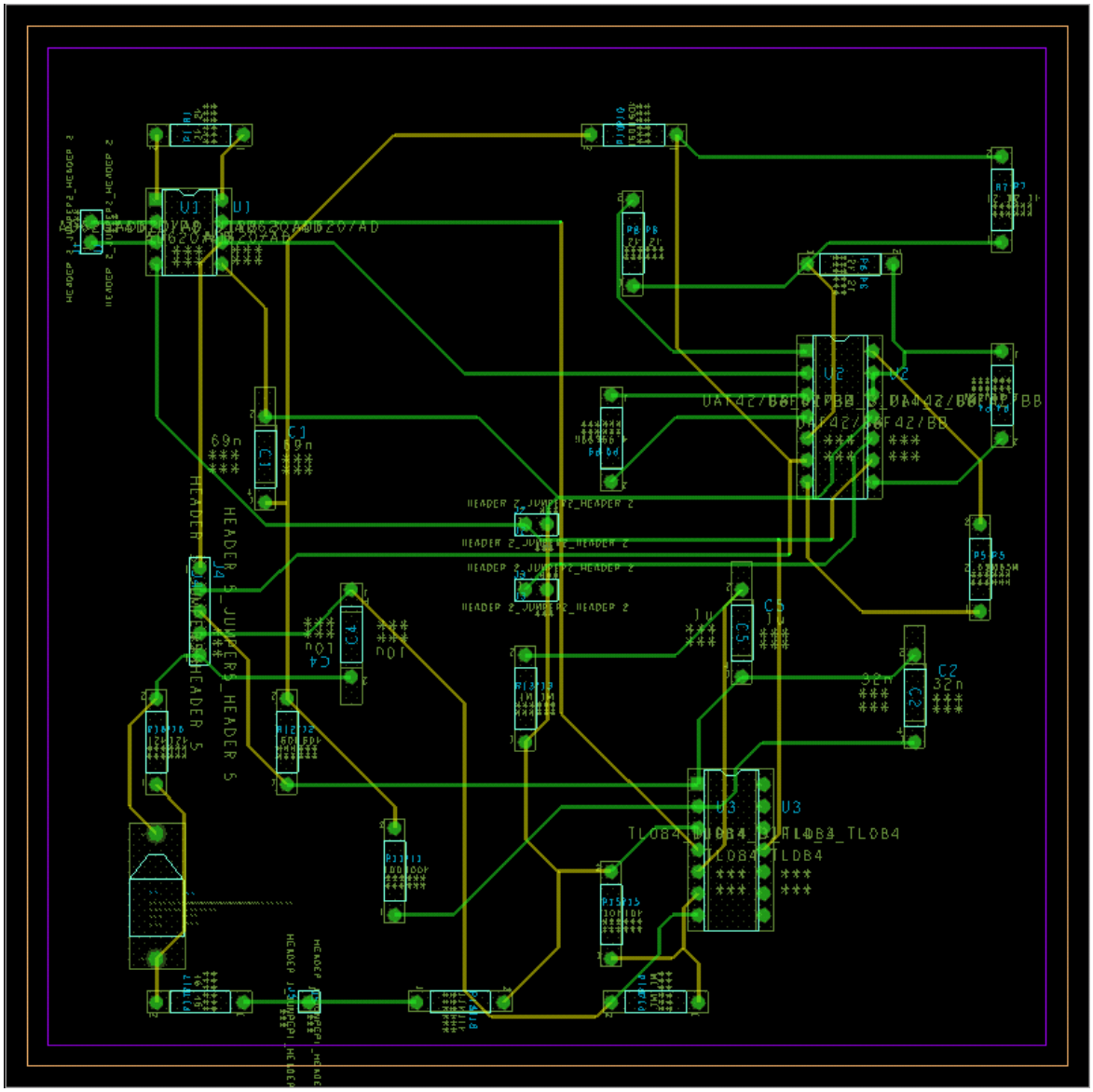


Fig. 31. PCB Schematic of Amplifier Circuit

This PCB Board was created for two reasons. The first reason was to make the data collection part of the lab more accurate, and the second reason was to reduce noise from the circuit. The circuit that was built for the actual data collection of the lab was created on a breadboard, which provided user-friendly platform with which to create a prototype circuit, but still lacked the direct and efficient structure of a regular PCB board. There was a lot of 60Hz noise from the breadboard circuit that distorted the final data that was collected from the circuit. The second reason why it was necessary to create this circuit was to ensure future progress on

the project. By creating this PCB board future researchers merely have to connect a few different pins together correctly in order to make an EEG measurement successful. Other students should find it easier to collect the data in future research, and perhaps find an application of the data for other uses as listed from before.

The applications of this project are far reaching. Not only can wirelessly sending EEG signals make data collection for biomedical sensing easier, but it can also be used as a sort of feedback to tell an individual whether they are performing a particular task correctly. For example, telling someone when to take a shot in archery would be a perfect application for this system as well as improving an individual's golf swing or performance in some sport involving skilled accuracy. Another application would be modeling games in order to have an individual play a video game using only his or her mind. Medical monitoring applications or the possibilities for increased access to technological solutions for those with limited mobility are broad. This just shows that the uses of wireless EEG signals for data analysis and robot control are limitless.

VII.REFERENCES

[1] A. Callaway. (2012). *Data mining of portable EEG brain wave signals for sports performance analysis: An Archery case study* [Online]. Available FTP: http://www.academia.edu Directory: 1255318/Data_mining_of_portable_EEG_brain _wave_signals_for_sports_performance_analysis_An_Archery_case_study File: Alpha-Active_ICSEMIS_abstract.pdf

[2] (2011). *Anatomy of the Brain* [Online]. Available FTP: http://www.fairviewebenezer.org/ Directory: HealthLibrary/Article/83352

[3] (2013). *Biomedical Signals Acquisition* [Online]. Available FTP: http://www.medicine.mcgill.ca/ Directory: physio/vlab/biomed_signals/eeg_n.htm

[4] D. Kushner. (2011) *The Making of Arduino - IEEE Spectrum* [Online]. Available FTP: http://www.spectrum.ieee.org Directory: geek-life/hands-on/the-making-of-arduino

[5] (2013) *Arduino Uno* [Online]. Available FTP: http://arduino.cc/ Directory: en/Main/ArduinoBoardUno#.UxjReT9dV4c

[6] (2011) XBee radios: *Simple Wireless Communication* [Online]. Available FTP: http://www.ladyada.net/ Directory: make/xbee/

[7] (2011). *AD620: Low Drift, Low Power Instrumentation Amp with Set Gains of 1 to 10000* [Online]. Available FTP: http://www.analog.com Directory: en/specialty-amplifiers/instrumentation-amplifiers/ad620/products/product.html

[8] J. Molina. (2000). *DESIGN A 60Hz NOTCH FILTER WITH THE UAF42* [Online]. Available FTP: http://www.ti.com/ Directory: lit/an/sbfa012/sbfa012.pdf

[9] (2012). *DIY EEG (and ECG) Circuit* [Online]. Available FTP: http://www.instructables.com Directory: id/DIY-EEG-and-ECG-Circuit/

[10] John. (2009). *Diode Clamping Circuits* [Online]. Available FTP: http://www.circuitstoday.com/ Directory: diode-clamping-circuits

[11] R. Faludi, “Ins and Outs,” in *Building Wireless Sensor Networks*, Sebastopol, CA: O’Reilly Media Inc., 2011, pp. 93-109.

VIII. APPENDIX

A. READING EEG SIGNAL/SENDING COMMAND CODE

```
#include <XBee.h>
#define FILTER_SHIFT 3
int32_t filter_reg;
int16_t filter_input;
int16_t filter_output;
//--- XBEE-SETUP ---------------------------
 XBee xbee = XBee();
 unsigned long start = millis();
 uint8_t payload[7];
 Tx16Request tx_5001 = Tx16Request(0x5001, payload, sizeof(payload));
 TxStatusResponse txStatus = TxStatusResponse();

//values to be transmitted 10 Bit, i.e. (0-1023)
int value1 = 0;
int value2 = 0;
int value3 = 0;
int value4 = 0;
int statusLed = 11;
int errorLed = 12;
int sensorPin = A0;
int average = 0;
int oldaverage = 0;
int times = 160; //ms
int command;
int sum;
int time1 = 0;
int time2 = 0;

void setup(void)
{
   // initialise Xbee feedback LEDs
   pinMode(statusLed, OUTPUT);
   pinMode(errorLed, OUTPUT);
   Serial.begin(9600);
   xbee.begin(Serial);

}

void loop(void)
{
   command = getcommand(1);
   command = getcommand(0);
```

```
    Serial.print("Command: ");
    Serial.println(command);

    payload[0] = command >> 8 & 0xff; // payload[0] = MSB.
    payload[1] = command & 0xff;
    //convert command to Byte
    xbee.send(tx_5001);
    //send data
    if (command != -1)
      delay(500);
}

int getcommand(int skip)
{
  int command = -1;
  int newaverage;
  int resolution = 20;
  sum = 0;
  for(int i = 0;i<resolution;i++)
      sum = sum + analogRead(sensorPin);
  value2 = sum/resolution;
  //collect 20ms of data
  do{
    value1 = value2;
    sum = 0;
    for(int i = 0;i<resolution;i++)
       sum = sum + analogRead(sensorPin);
    //collect 20ms more of data
    value2 = sum/resolution;
  }while(abs(value2-value1) < 12 || abs(value2-value1) > 20);
  //compare two samples of data to warrant eye movement change
  sum = 0;
  for(int i =0; i<times; i++){
     sum = sum + analogRead(sensorPin);
  }
  average = sum/times;
  //collect data average over a larger interval
  int oldcommand = command;
  Serial.println("Value");
  Serial.println(value2-value1);
  Serial.println("Average");
  Serial.println(average-oldaverage);
  //error checking

  if(average-oldaverage>50){
    //turn right command
```

```
   if (skip==0)
     while(getcommand(1)!=-1){}
   command = 0;
  }
  else if (average-oldaverage<50){
   //turn left command
   if (skip==0)
     while(getcommand(1)!=-1){}
     //infinite loop while no change present
   command = 1;
  }
  oldaverage = average;
  //reset
  return command;
}
```

B. BOE-BOT CONTROL/RECEIVING COMMAND CODE

```
#include <XBee.h>
#include <Servo.h>

Servo servo;
Servo servo2;
// XBEE setup ------------------------------
XBee xbee = XBee();
XBeeResponse response = XBeeResponse();
Rx16Response rx16 = Rx16Response();
Rx64Response rx64 = Rx64Response();

int command = -1;

void setup(void)
{
Serial.begin(9600);
 xbee.begin(Serial);

 servo2.attach(11);
 servo.attach(12);
}

void loop(void)
{
 stopmoving();
 if (command == 0){
    //turn right action
    turnright();
```

```
   delay(300);
  }
  else if (command == 1){
    //turn left action
    turnleft();
    delay(300);
  }

 xbee.readPacket(500);
 //read command
 if (xbee.getResponse().isAvailable()) {
   xbee.getResponse().getRx16Response(rx16);
   uint8_t analogHigh = rx16.getData(0);
   uint8_t analogLow = rx16.getData(1);
   command = analogLow + (analogHigh * 256);
   //convert command to integer
   Serial.println(command);
   //error checking
 }
}

void forward()
{
  servo.write(80);
  servo2.write(100);
}

void stopmoving()
{
  servo.write(90);
  servo2.write(90);
}

void turnright()
{
  servo.write(100);
  servo2.write(100);
}

void turnleft()
{
  servo.write(80);
  servo2.write(80);
}
```