



BACHELOR THESIS

Designing a Sleep Monitoring Device to Induce Lucid Dreaming

Jose M. Aponte Urmeneta

EEMCS / ELECTRICAL ENGINEERING
BSS

EXAMINATION COMMITTEE
Dr. Utku S. Yavuz
Carlijn I. R. Braem
Dr. Ir. Anne-Johan Annema

25/6/2020

UNIVERSITY OF TWENTE.

Table of Contents

Introduction	3
Background	5
Method	7
Hardware	7
EOG Electrodes	7
Amplification	8
Filtering EOG Signal	11
Optocoupler Readout	18
Filtering Optocoupler Signal	23
Processor	24
Communication Interface	25
Feedback Interface	25
Power Management	25
Software	28
MATLAB Connection	28
Programming Processor	28
Algorithm	28
Digital Filtering	28
Validation	29
Hardware	29
Supply Voltage Regulation	29
Virtual Ground	29
Optocoupler Readout	29
Band-pass Filter	29
EOG Amplification	30
Notch Filter	30
Low-pass Filter	30
Joint Filter Response	30
Software	30
Data Communication	30
Algorithm	31

Results.....	32
Eagle CAD Design	32
3D Render	32
PCB	33
Communication with MATLAB	35
Algorithm	35
Feedback Mechanism	37
Discussion.....	38
Conclusion.....	39
References	40
Appendix	42
Figures.....	42
Device Schematics.....	53
MATLAB Code	56
COM Port Data Acquisition	56
Algorithm Code	57
STM32 main.c Code	58

ABSTRACT – This thesis reviews the design process required to create a sleep monitoring device capable of detecting REM sleep. It explores current methods used for measuring and detecting sleep stages in a subject. The advantages and disadvantages of the current methods are discussed and compared with a newly proposed method for sensing eye movements. Optocouplers are already used for distance measurement, interruption based on object obstruction, and industrial processes. For this reason, the proposed method for detecting REM involves the use of optocouplers. The data gathered by the built device confirms the feasibility of using optocouplers to measure eye movements. The proposed algorithm to detect REM is able to detect it to some extent and it will continue to be improved.

Introduction

In the search for knowledge about the mysteries behind dreams, many people have tried to study the reason for why humans dream. Other people have focused more on a different approach – study dreams themselves and the different types and interactions one may have in said dreams. One example is Dutch author Frederik van Eeden. In 1896, Van Eeden started studying his dreams for years in an attempt to identify and classify the different types of dreams he experienced [1]. Van Eeden kept a journal where he would write down and keep a record of his dreams and the details he remembered [1]. During his 16 years of studying his dreams, he was able to identify seven different categories of dreams. The most interesting category is the seventh and last one; dreams which he calls lucid. He defines lucid dreams as reaching a state of perfect awareness where the sleeper is able to willfully carry out anything the sleeper desires [1].

Lucid dreaming was also experienced and identified by other people such as other writers. It was no mystery to people that lucid dreaming could be done. However, the scientific interest in researching more about this phenomenon would not take off until later. In 1990, Stephen LaBerge, Ph.D. in psychophysiology published a book in which he outlines the benefits of a lucid dream and how to induce one. LaBerge agrees that lucid dreaming has been known for a long time and that his personal experience and research, along with other researchers' findings have only recently started to uncover this "unusual state of consciousness [2]. According to LaBerge, lucid dreams can improve the quality of 'waking' life [2]. LaBerge's book also serves as a step-by-step guide on how to induce and maintain lucid dreaming. The techniques used in his book are among the most common, most practiced, and most researched methods of inducing lucid dreaming. While it is true that anyone following the steps in his book can achieve lucid dreaming, this process does take a long time [2].

Because of this, motivation was found in developing an easier method to induce lucid dreaming. LaBerge has spent time researching more about lucid dreams in his research group in Stanford University. His research group has conducted experiments where subjects realize they are dreaming by means of an external cue [3]. By finding the link between external cues and dreams, a new way could be paved to allow some technological enhancement to take advantage of such link in order to make the process of inducing lucid dreaming much easier. LaBerge's group has tested different feedback methods: auditory feedback, tactile stimuli, olfactory stimuli, and visual feedback with the latter one being the most promising one [3]. With this motivation, more studies have been made, research has been recorded, and projects related to the induction of lucid dreaming have been developed. Some examples of these projects are mentioned below:

A project which has the goal of trying to induce lucid dreaming after the REM sleep stage has been detected is the work of Fredrik Flornes Ellertsen, a master in Electronic System Design and Innovation. His work is based on a basic prototype and test protocol. Ellertsen looked at LaBerge's research for evidence regarding external cues to induce lucid dreaming. He wrote that in a study made by LaBerge, flashing lights were the external cue and the results were that lucid dreams were induced a total of 33 times over 58 nights with 44 subjects [4]. This data suggests that it is possible to create a REM detection device to induce lucid dreaming.

Another project similar to that of Ellertsen's is the open source project OpenLD. OpenLD was created by a group of engineers, students, and hobbyists. This project uses an 8-channel Electroencephalogram (EEG) to perform real-time brainwave measuring and analysis. This project was started on August of 2016 and to this day receives periodic updates on progress and data gathered [5].

A third project with a similar goal was started by the Massachusetts Institute of Technology (MIT). In 2019, the MIT conducted an experiment involving a new method to detect REM sleep in a user. A group of students developed an open source device that measures heart rate, skin conductance, and muscle activity from a glove they call Dormio. This device gathers the biosignals and sends them via Bluetooth to a different device, such as a mobile phone, which then uses a web server application to do the processing required to detect the sleep stages [6].

Similarly to the aforementioned projects, this thesis focuses on the design process of a REM-detecting sleep monitoring device. It explores currently available sensing methods while proposing a new measurement technique. This technique involves the use of a reflective optocoupler placed over closed eyes to detect eye movements based on the irregular shape of the eyeballs. In contrast to these projects, the proposed device has the advantages that it can be comfortably worn, is not obtrusive, and does not require specialized medical equipment to operate. In addition to these, the proposed device offers the possibility of processing and transmitting the measured data which could help in making sleep studies easier to carry out. It is worth noting, however, that the extent to which the feedback method is able to induce lucid dreaming should be left for a different research project.

Background

This section highlights the knowledge required from some measurement concepts. It describes the basic working principles behind the different sensing interfaces used during a polysomnography (PSG). Additionally, it explains how the advantages and disadvantages of each sensing method influence the design and requirements of a sleep monitoring device. Lastly, it provides information about the novel REM sensing method previously proposed.

There are different ways to gather information about sleep stages and sleep quality from a person. These methods include electroencephalograms (EEGs), electrooculograms (EOGs), electromyograms (EMGs), and electrocardiograms (ECGs). The combination of all of these measurement setups and signals is called polysomnography. With a polysomnographic measurement, a specialist can properly identify the different sleep stages of a subject. It is important to remark that only EEGs and EOGs are used in sleep stage detection studies while the incorporation of EMGs along with ECGs is only common in sleep disorder studies.

An EEG consists of measuring the brainwaves created by the electrical impulses in firing neurons. According to an article published by Johns Hopkins Medicine Institute, an EEG is a test that detects electrical activity created by the electrical impulses of neurons in the brain [7]. During the procedure, electrodes consisting of small metal discs with thin wires are pasted onto the scalp [7]. An EEG is used as a ‘golden standard’ in detecting different sleep stages during a polysomnography because brain activity is the most representative of the state under which the subject is at a given point in time. Based on the frequencies detected and the energy of an EEG signal, different brain activity patterns can be identified which aid in the further identification of a sleep stage.

An EOG on the other hand, measures the eye movement based on the potential difference between the cornea and the retina [8]. According to ADInstruments, an electronics development company specialized in the design of biosignal-detection devices, the amplitude of bioelectrical signals are typically in the range of a few hundred μV and an amplifier is necessary to accurately record the EOG [8]. By detecting the eye movements at any given point during sleep, a conclusion can be drawn to some extent that the user is undergoing Rapid Eye Movement (REM) depending on different properties of the signal.

An EMG measures the potential difference across a muscle created by the nerve’s stimulation of the muscle [9]. The use of an EMG measurement during a polysomnography is typically used to detect and diagnose sleep disorders. As for REM detection, according to Carmel Armon, MD, an EMG channel is used to record atonia during REM sleep or lack thereof in patients with parasomnias [10].

An ECG measures the potential difference on two points on the chest above the heart created by the contraction of the cardiac muscles [11]. The most common use for an ECG is a heart rate monitor. A heart rate monitor can be helpful in identifying the activity state of a person – at rest, exercising, sleeping, etc.

For the purpose of this research project, an EEG proves obtrusive to the user since it requires a large number of electrodes all around the scalp of the user. The electrodes used for a non-medical grade EEG are usually held on the right place by a ‘sensor cap’ which is difficult to place. Since the user of the sleep monitoring device will have to set up the product moments before going to sleep, this method is not an option. Thus, the proposed device must be unobtrusive and easy to use. Another reason why the use of the EEG is rejected is due to the low voltage levels of brainwave signals. The voltage of these signals typically ranges from 0.1 to 10 μV [12]. Detecting such low voltage levels usually requires specialized

medical equipment and would thus make working with these signals difficult on a low power, battery operated device.

Relying only on an EMG to detect REM sleep is not possible since it is unable to give conclusive information about any given sleep stage. Another limitation is that, since the placement of the electrodes in this measurement setup heavily influences the output signal, the user might not measure the right signal when using the device. Thus, the proposed device must be small so that it is easy to place and use.

The chosen method to use in this project is the EOG because it provides the closest interpretation of eye movements for REM detection while having a reasonable signal amplitude (0.1 mV to 1 mV). Additionally, Ellertsen has already experimented with this measurement method and provided proof that REM detection based solely on this method is possible. The disadvantage his setup has is that it is obtrusive and depends on a lot of wiring. Therefore, the proposed device should be as comfortable as possible to wear as to not affect the quality of sleep of the user.

In addition to the EOG method, a novel idea will be researched. This idea involves the use of optocouplers over the eyes to detect the small changes in eye movements due to the irregular spherical shape of the eyeball [13]. For this specific purpose, a reflective optocoupler which consists of an infrared light-emitting diode (LED) and a phototransistor in one package with the top uncovered is used. This type of optocoupler is usually used to measure distances, to act as a proximity switch, as an object counter, or in industrial control [14].

The constant impact of infrared light emitted by an LED on the eye is not harmful to it [15]. According to Renesas Electronics, Light-emitting Diodes (LEDs) have not been found to cause any damage [15]. As an added layer of protection, the light would not directly hit the lens since the eyes would be closed. The light intensity would be diminished by the eyelids before hitting the retina. Therefore, it is safe to conclude that this measuring method would not be hazardous to the user.

Method

The current section focuses on the design process of the sleep monitoring device. The design of the device is broken down into two sub-sections: hardware and software. The hardware section highlights the choice of components for sensors, analog filters, processor, communication interface, connectivity, and power management. The software section outlines the choice of processing application, digital filtering, and algorithm for REM detection.

When designing a system as an engineer, requirements, functions, and tradeoffs must always be determined and reviewed beforehand. In the case of this research project, there are three key requirements that must be considered vital when designing the sleep monitoring device, namely: the device should not be obtrusive to the user, the device should be able to detect REM sleep at any point during the night, and the device should not interfere with the quality of sleep of the user in a negative way. As for functions, the most basic ones include detecting eye movements, processing the data collected, storing or transmitting the process data, and provide feedback to the user.

With these specifications set, a system diagram is made. This diagram shows the basic functionality of the device and provides a visual insight into how the device should be built. The system diagram is shown in Figure 1.

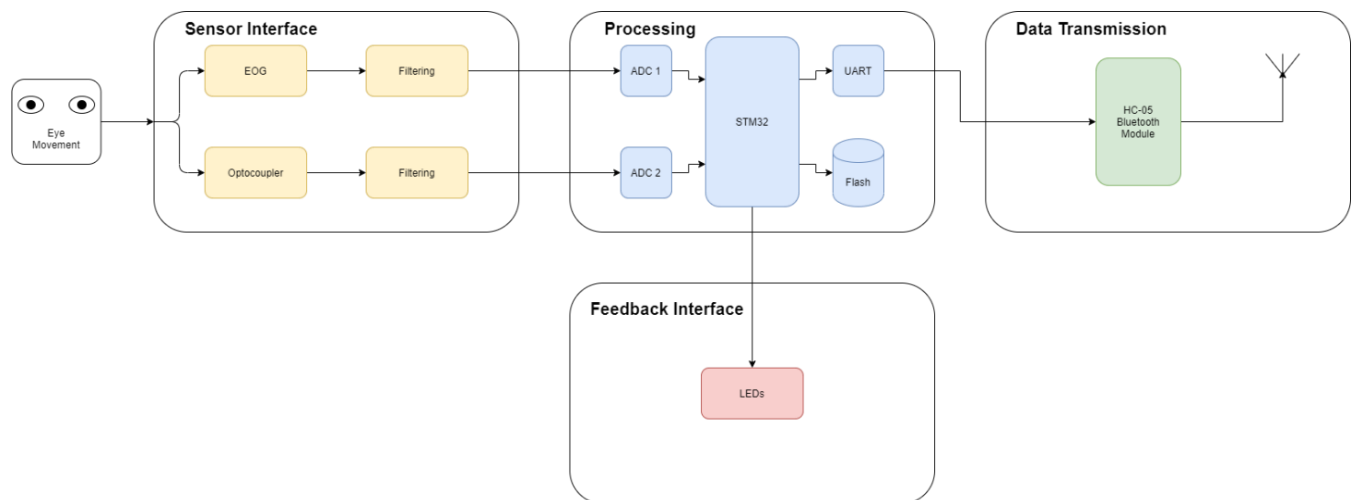


Figure 1 – System diagram showing the necessary blocks and interfaces for the sleep monitoring device.

After identifying the basic building blocks for the sleep monitoring device, a more in-depth mathematical analysis of each one can be made. Firstly, the sensor interface will be discussed and explained, then the processing section, then the data transmission, and finally the feedback interface. These will complete the hardware subsection, after which the software section will be explained by mentioning the software, algorithms, and data gathered from the device.

Hardware

The first block in the system diagram is the sensor interface. As mentioned before, one of the sensing methods is the use of EOG electrodes. In order to measure EOG signals, they must be amplified to ensure that an Analog-to-Digital Converter (ADC) is able to detect them accurately.

EOG Electrodes

The electrodes required to pick up the EOG signals have different types. These include wet electrodes, dry electrodes, gold plated electrodes, alloy electrodes, and active electrodes. For the purpose

of this project, commercially available electrodes are expensive and optimized to work with a medical grade device. For this reason, the decision to use custom-made electrodes was realized. The electrodes consist of a wire whose end terminal is connected to a round patch of aluminum foil. The aluminum foil is the terminal that provides contact with the skin at the desired locations. One of the electrodes is shown in Figure 37 attached to the appendix of this document.

Amplification

An EOG signal is usually in the range of 0.1 mV to 1 mV [8]. Assuming that an ADC with a resolution of 10 bits is used, the minimum change in voltage that said ADC would be able to detect is given by the following equation:

$$\Delta V = \frac{V_{ref}}{2^n} \quad (1)$$

$$\Delta V = \frac{3.3[V]}{1024} = 3.22[mV] \quad (2)$$

Even if an ADC with an improved resolution of 12 bits, it still would not be enough to detect a pure, unaffected EOG signal. With a 12-bit resolution, the new minimum voltage ‘step’ is around 800 μV as shown by using the previous equation with 12 bits instead of 10.

For this reason, the EOG signal needs to be amplified. However, the amplification cannot be done with a common Operational Amplifier (OpAmp). An OpAmp is an electronic component whose purpose is to amplify the voltage between its input terminals by a gain constant – ideally infinity. The schematic symbol of this device is shown in Figure 38 in the appendix.

If this OpAmp were to be used to amplify the EOG signal as shown in Figure 2, then the output would have no useful meaning since any difference in voltage between the two electrodes would be output as either the positive supply (V_{cc}) or the negative supply (Gnd) on the OpAmp.

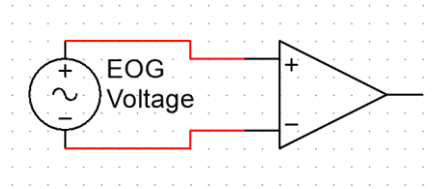


Figure 2 – Voltage source connected to both inputs of the OpAmp with no reference to ground.

Thus, another approach must be followed. There is another device that can accurately control the amount of gain given to the input terminals of an OpAmp. This device is called an Instrumentation Amplifier (InAmp). With such an amplifier, the differential voltage created by the EOG signals can be amplified with any factor in order to amplify the signal’s voltage as desired. The chosen Instrumentation Amplifier for the sleep monitor is the AD8237 because of its low supply current usage (115 μA), its rail-to-rail input and output capabilities, its ability to set the gain with two external resistors, and its “excellent gain accuracy” [16].

The symbol and pin configuration of this amplifier is shown in Figure 39 in the appendix. According to its schematic symbol, the output equation, or transfer function, of the AD8237 is:

$$V_{out} = A((IN_+ - IN_-) + (REF - FB)) \quad (3)$$

Where A is a high-value gain constant. Based on this equation, the schematic for EOG signal detection and amplification can be designed. A common amplifier circuit is a so-called non-inverting amplifier. The schematic for this circuit is shown in Figure 3.

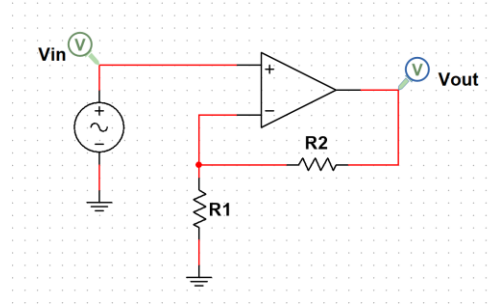


Figure 3 – Non-inverting amplifier circuit.

This setup is called a non-inverting amplifier because of its behavior – the output voltage is ‘positively’ proportional to the input voltage. According to its transfer function, whose derivation is shown below, the input voltage’s nature is not inverted. Hence, the non-inverting amplifier name.

$$V_{out} = A(V_+ - V_-) \quad (4)$$

$$V_+ = V_{in} \quad (5)$$

$$V_- = \frac{R_1}{R_1 + R_2} V_{out} \quad (5)$$

$$V_{out} = A \left(V_{in} - \frac{R_1}{R_1 + R_2} V_{out} \right) \quad (6)$$

$$V_{out} \left(1 + A \frac{R_1}{R_1 + R_2} \right) = A V_{in} \quad (6)$$

$$V_{out} = \frac{A}{1 + A \frac{R_1}{R_1 + R_2}} V_{in} \quad (7)$$

$$\lim_{A \rightarrow \infty} V_{out} = \frac{R_1 + R_2}{R_1} V_{in} \quad (7)$$

Similar to this amplifier circuit, the AD8237 can be used to make a non-inverting amplifier. The difference with the previously mentioned circuit is that in a normal non-inverting amplifier, the input voltage must be referenced to ground ($V_{in} = V_+ - 0[V]$). With the AD8237, the input voltage does not require a reference to ground. For instance, in the case of the EOG electrodes, the non-inverting amplifier with the AD8237 will amplify the difference in voltage between the two electrodes ($V_{in} = V_+ - V_-$). The circuit for this configuration is shown in Figure 4.

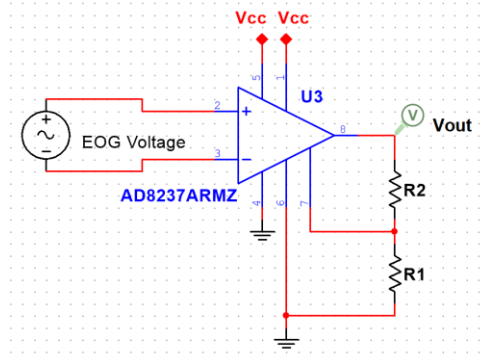


Figure 4 – Non-inverting amplifier with an AD8237 as an OpAmp.

The transfer function for this amplifier can be derived as follows by using (3):

$$\begin{aligned} IN_+ - IN_- &= V_{in} \\ REF &= 0[V] \end{aligned} \quad (8)$$

$$FB = \frac{R_1}{R_1 + R_2} V_{out}$$

$$V_{out} = A \left(V_{in} + 0[V] - \frac{R_1}{R_1 + R_2} V_{out} \right) \quad (9)$$

$$V_{out} \left(1 + A \frac{R_1}{R_1 + R_2} \right) = A V_{in} \quad (10)$$

$$V_{out} = \frac{A}{1 + A \frac{R_1}{R_1 + R_2}} V_{in}$$

$$\lim_{A \rightarrow \infty} V_{out} = \frac{R_1 + R_2}{R_1} V_{in} \quad (11)$$

The AD8237's datasheet includes a table for the recommended resistor values to set the gain of the amplifier in this configuration. This table is shown in Figure 40 in the appendix. By following this table, the values for the resistors are chosen to be 1 kΩ for R_1 and 512 kΩ for R_2 since it is the value closest to 499 kΩ available in the electronics supplier. Another reason for choosing 512 kΩ is the extra gain that it adds to the amplifier. The resulting gain is then given by (11):

$$G = \frac{V_{out}}{V_{in}} \quad (12)$$

$$G = \frac{\frac{R_1 + R_2}{R_1} V_{in}}{V_{in}} \quad (13)$$

$$G = \frac{R_1 + R_2}{R_1} \quad (14)$$

$$G = \frac{1[k\Omega] + 512[k\Omega]}{1[k\Omega]} = 513 \quad (15)$$

The reason for not choosing 1 MΩ for R_2 is that such a high resistor value starts contributing noticeably to the noise in the circuit. While it is true that the gain of the amplifier would be maximized, the InAmp would be working close to its limits which also contributes to noise in the measured signal

[16]. In addition to this, any further gain that may prove necessary can be added in the filters that are located after the first amplification stage.

Filtering EOG Signal

The amplified signal must be fed to a filter in order to only gather the relevant frequencies affecting the signal. Since the device will incorporate two sensing methods to detect REM sleep, it must be noted that two different, and independent, filters must be designed. Firstly, the design of the first filter which is located at the output of the EOG amplifier will be discussed. Secondly, the design of the second filter which influences the optocouplers' measured signal will be explained.

As suggested by Carlijn Braem, a Biomedical Signals and Systems PhD student at the University of Twente, the 50 Hz mains line frequency from the power outlets is a significant problem when measuring biosignals. This frequency is parasitically coupled to the human body by stray capacitance between the body and ground due to the changing electric field created by the powerlines [17]. Thus, a filter that removes this frequency must be considered when designing the system.

It is also important to consider that in order to prevent aliasing when sampling an analog signal with an ADC, frequencies higher than that of half of the sampling frequency must be suppressed as much as possible. Aliasing is the presence of unwanted higher-frequency components in the sampled signal [18]. According to the Nyquist Sampling Theorem, a signal can be perfectly reconstructed if the maximum frequency present in the signal is at least half of the sampling frequency [19]. Therefore, it is paramount to incorporate an analog low-pass filter in the sensing portion of the device to remove any higher frequencies present.

The choice to include analog filters in the designed device is influenced by the previously mentioned fact. As previously mentioned, the 50 Hz frequency is problematic. The choice of including a band-stop filter, or Notch Filter, in addition to a low-pass filter will ensure that said frequency is suppressed as best as possible. Therefore, the first filter located at the output of the first amplifier stage is the Notch Filter.

Every filter can be identified as either passive or active. A passive filter consists of purely passive components; hence, the name passive filters. The passive components are resistors, capacitors, and inductors. A passive filter has some advantages and disadvantages over active filters. The advantages include having no need for extra tuning in terms of biasing voltage or conditions, less components are required, it can be built with only passive components, and, if the components are chosen to be good quality capacitors and inductors, less noise contribution. The disadvantages include not having the ability of adding gain to a certain bandwidth, high-valued components are needed for very low frequencies (0.1 Hz – 1 Hz), and the fact that loading effects on their output may play a role in defining the filter's response.

Because additional gain after the first amplifier stage is desired in order to amplify the EOG's signal as much as possible, an active filter is required. Also, since loading effects must be minimized, an active filter will prevent these effects by providing a point of low output impedance with an OpAmp in buffer configuration at the output of a filter. It should be noted that active filters typically depend on both positive and negative voltages because they rely on integrators and if integration is done solely on positive values, then the output signal will tend towards infinity.

In addition to the previously stated note, analog passive filters do not require any additional modifications. However, since the design incorporates the use of active filters, they need to be carefully biased. The term biasing usually refers to 'adding' a DC voltage or current to a transistor so that it

operates in a desired region. However, in this context, biasing refers to adding a DC voltage offset to a filter such that its behavior mimics that of an amplifier with both positive and negative supply voltages. For instance, an amplifier such as the one shown in Figure 5 has the property of inverting the input signal.

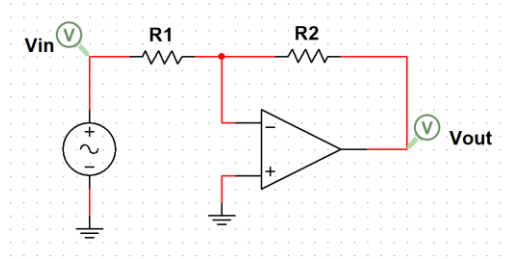


Figure 5 – Inverting amplifier circuit

The equation that portrays this behavior is derived as follows from (4):

$$V_+ = 0[V]$$

$$V_- = \frac{R_2}{R_1 + R_2} V_{in} + \frac{R_1}{R_1 + R_2} V_{out} \quad (16)$$

$$V_{out} = A \left(0[V] - \frac{R_2}{R_1 + R_2} V_{in} - \frac{R_1}{R_1 + R_2} V_{out} \right) \quad (17)$$

$$V_{out} \left(1 + A \frac{R_1}{R_1 + R_2} \right) = -A \left(\frac{R_2}{R_1 + R_2} V_{in} \right) \quad (18)$$

$$V_{out} = - \frac{A \frac{R_2}{R_1 + R_2}}{1 + A \frac{R_1}{R_1 + R_2}} V_{in}$$

$$\lim_{A \rightarrow \infty} V_{out} = - \frac{\frac{R_2}{R_1 + R_2}}{\frac{R_1}{R_1 + R_2}} V_{in} \quad (19)$$

$$V_{out} = - \frac{R_2}{R_1} V_{in} \quad (20)$$

Therefore, if a hypothetical voltage of 1 V would be applied to the input of this amplifier, the output would be proportional to -1 V. However, since there are no negative voltages in the device because it should be battery powered, the amplifier must be biased. This amplifier can be biased by adding a reference voltage source and a capacitor as shown in Figure 6.

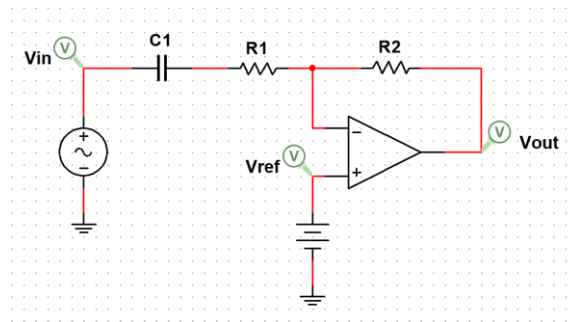


Figure 6 – Inverting amplifier with reference voltage for 'biasing' or offsetting.

Assuming that the capacitor is large enough to be considered a short for the significant frequencies in the signal, this configuration causes the transfer function to be modified in the following manner:

$$\begin{aligned} V_+ &= V_{ref} \\ V_- &= \frac{R_2}{R_1 + R_2} V_{in} + V_{out} \end{aligned} \quad (21)$$

$$V_{out} = A \left(V_{ref} - \frac{R_2}{R_1 + R_2} V_{in} - V_{out} \right) \quad (22)$$

$$V_{out} (1 + A) = A \left(V_{ref} - \frac{R_2}{R_1 + R_2} V_{in} \right) \quad (23)$$

$$V_{out} = \frac{A}{1 + A} \left(V_{ref} - \frac{R_2}{R_1 + R_2} V_{in} \right)$$

$$\lim_{A \rightarrow \infty} V_{out} = V_{ref} - \frac{R_2}{R_1 + R_2} V_{in} \quad (24)$$

Now, assuming that the supply voltage is V_{cc} , and the reference voltage is half of that ($\frac{V_{cc}}{2}$), then any voltage above $\frac{V_{cc}}{2}$ would be considered ‘positive’ voltage and any voltage under $\frac{V_{cc}}{2}$ would be considered negative voltage. Thus, by biasing an amplifier, there is no need for negative voltages and filters can work appropriately with reference to the bias voltage.

50Hz Notch Filter

A band-stop filter can be built in different ways. One example is by using a series RLC circuit as shown in Figure 41 in the appendix. This configuration would create a band-stop filter. However, this build is not ideal for the sleep monitor because the 50 Hz powerline frequency would require both a large inductor and capacitor. There is an alternative configuration for building a narrower band-stop filter, or notch filter, which is shown in Figure 7.

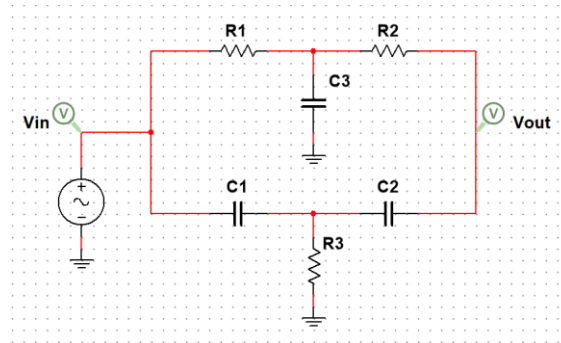


Figure 7 – Alternative band-stop filter circuit.

With this configuration, two filters are acting simultaneously on the input signal resulting in a steeper frequency reject response. There is a requirement for this filter to work properly, namely the values of the components must be proportional to each other. The requirement is $R_1 = R_2 = 2R$ while $R_3 = R$ and $C_1 = C_2 = C$ while $C_3 = 2C$ [20]. With this requirement set, the equation that shows the behavior of the filter is derived as follows by applying the node-voltage method:

$$n_1: i_{in} - \left(\frac{1}{2R} + j\omega C \right) V_{n_1} + \frac{1}{2R} V_{n_2} + j\omega C V_{n_3} = 0 \quad (25)$$

$$\begin{aligned}
n_2: \frac{1}{2R} V_{n_1} - \left(\frac{1}{2R} + \frac{1}{2R} + j\omega 2C \right) V_{n_2} + \frac{1}{2R} V_{n_4} &= 0 \\
n_3: j\omega C V_{n_1} - \left(j\omega C + j\omega C + \frac{1}{R} \right) V_{n_3} + j\omega C V_{n_4} &= 0 \\
n_4: \left(j\omega C + \frac{1}{2R} \right) V_{n_4} &= \frac{1}{2R} V_{n_2} + j\omega C V_{n_3} \\
V_{n_1} &= V_{in} \\
V_{n_4} &= V_{out}
\end{aligned}
\tag{26}$$

$$\tag{27}$$

By using n_2 , n_3 , and n_4 , the relation between the input and output voltages can be derived. The derivation is lengthy, so only the final relation is shown in Figure 8.

$$\frac{V_o}{V_i} = \frac{s^3 + s^2 \frac{1}{C_1} \left(\frac{1}{R_1} + \frac{1}{R_2} \right) + s \frac{1}{C_1 R_1 R_2} \left(\frac{1}{C_3} + \frac{1}{C_2} \right) + \frac{1}{C_1 C_2 C_3 R_1 R_2 R_3}}{s^3 + s^2 \left(\frac{1}{C_1 R_1} + \frac{1}{C_1 R_2} + \frac{1}{C_2 R_2} + \frac{1}{C_2 R_3} + \frac{1}{C_3 R_2} \right) + s \left(\frac{1}{C_2 R_3 C_1 R_1} + \frac{1}{C_2 R_3 C_1 R_2} + \frac{1}{C_3 C_1 R_1 R_2} + \frac{1}{C_2 C_1 R_1 R_2} + \frac{1}{C_2 C_3 R_3 R_2} \right) + \frac{1}{C_1 C_2 C_3 R_1 R_2 R_3}}$$

Figure 8 – Notch filter transfer function [21].

Thus, by setting the capacitors to a relatively low value such as 100 nF, the corresponding resistances may be calculated by applying the notch frequency equation [20] to set it at 50 Hz.

$$f_{notch} = \frac{1}{4\pi RC} \tag{28}$$

$$R = \frac{1}{4\pi(50[Hz])(100[nF])} \approx 15.9[k\Omega] \tag{29}$$

Having set the values of the components in the filter, an amplifier in buffer configuration is added to the output of the filter to provide a low output impedance to the filter circuit. This results in the circuit shown in Figure 42 in the appendix. It is important to note that the biasing point, in this configuration, is located in the common node between the two filters. Since the output of the first amplifier stage is biased around half the voltage of the supply, then applying the same bias voltage to the common node in the filter will result in the correct filter behavior which will present no problem to the buffer at the output of the filter. The biased circuit is shown in Figure 9.

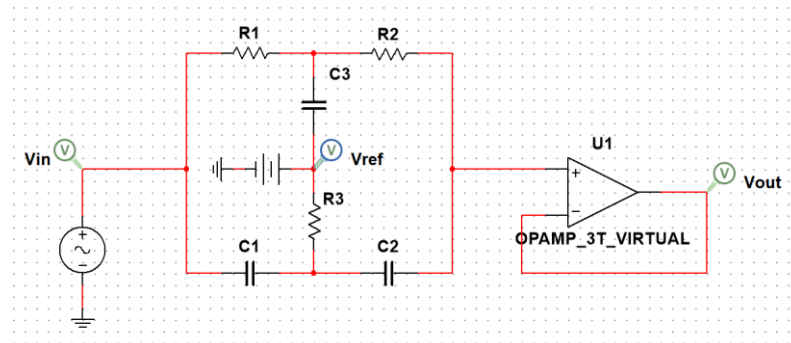


Figure 9 – Notch filter circuit with additional reference voltage.

Having these design criteria in mind as well as the component values, the circuit was created in Multisim, a circuit simulation software made by National Instruments. It is also important to note that in order to have the simulations be representative of the practical situation, component tolerances must be

introduced in the simulations. This will prevent the situation from being 100% ideal and it will make the simulation take into account the mismatch between the components' values. The filter's response can be seen in Figure 10.

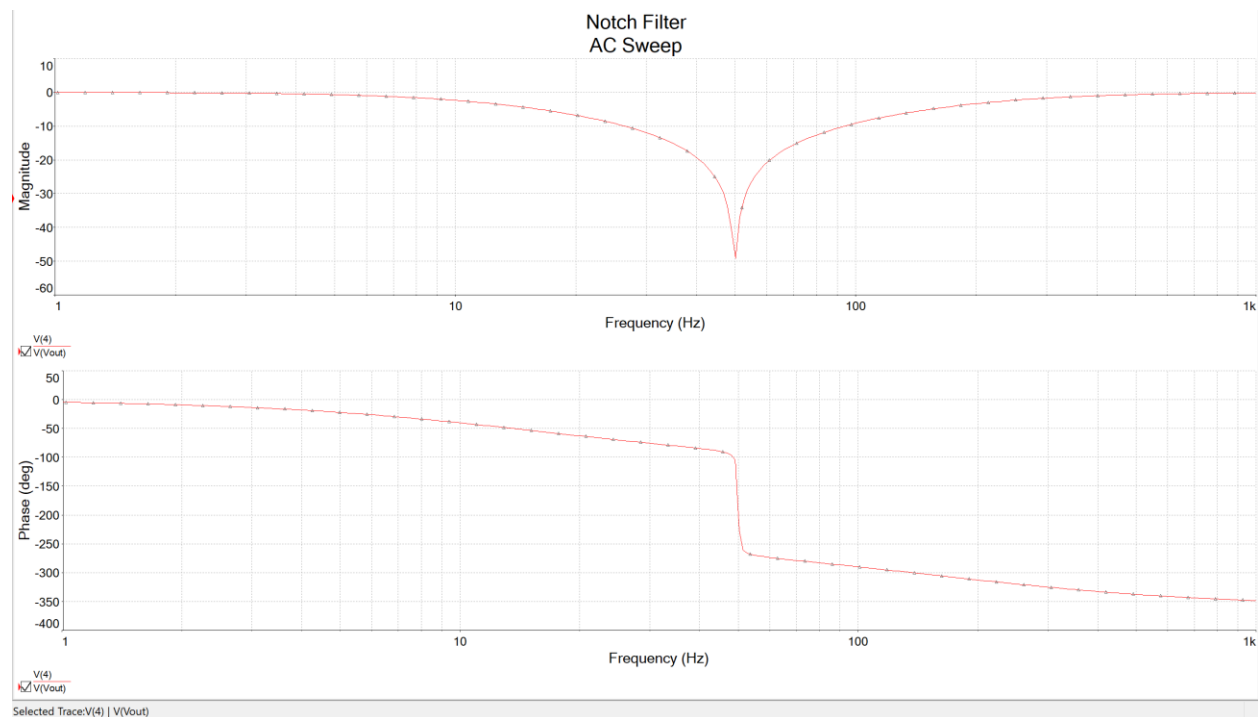


Figure 10 – Notch filter's response as shown in a Bode Plot.

By running a transient simulation, the 'real-time' response can be observed. Figure 43 shows the filter's output voltage when the filter is not properly biased and Figure 44 shows the output voltage when the filter is properly biased. These figures are attached to the appendix of this document.

In order to verify the filter better, an interactive simulation is run. In this mode, the simulation runs continuously and allows components such as a spectrum analyzer to be used. In order to test if the filter does in fact suppress the desired frequency, three voltage sources are placed at the input: one source at 10 Hz, the second one at 50 Hz, and the third one at 100 Hz. By looking at the spectrum analyzer's output shown in Figure 45 and Figure 46 which are located in the appendix, it can be concluded that the filter does in fact suppress the desired frequency.

Low-pass Filter

According to a study conducted by Kazumi Takahashi, 80% of all measured eye movement intervals range from 0.1 seconds to 2.6 seconds suggesting that the bandwidth of interest for EOG signals is between 0.3 Hz ($\frac{1}{2.6[s]}$) and 10 Hz ($\frac{1}{0.1[s]}$) [22]. Therefore, the cutoff frequency for the low-pass filter subsequent to the notch filter should be chosen according to this bandwidth. Since the filter should feature gain, the configuration shown in Figure 11 must be used.

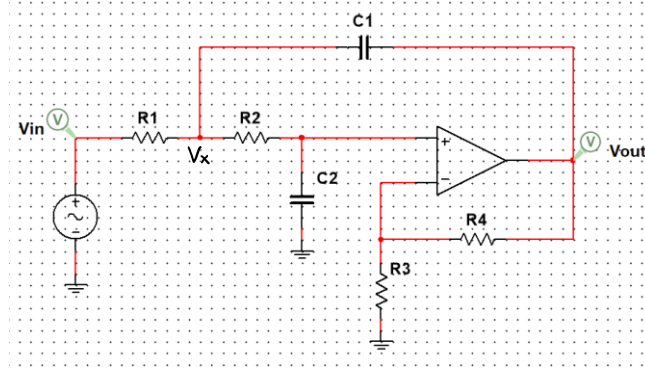


Figure 11 – Second order active low-pass filter circuit.

This is a second order lowpass filter with gain determined by the feedback resistors R_3 and R_4 . Because this filter also amplifies DC voltage, only a gain of 2 can be used; otherwise, the gain would amplify the DC voltage in such a way that the output of the filter would clip to the positive supply on the OpAmp. The cutoff frequency of the filter was chosen to be 1 Hz because the gain of the filter influenced the response of the previous notch filter. For instance, with a gain of 10 and a cutoff frequency of 10 Hz, the value of the 50 Hz frequency would be around 10 dB which already defeats the purpose of the first filter because it would amplify the already reduced signal.

The cutoff frequency equation for this filter may be derived as follows:

$$V_x: \frac{V_{in} - V_x}{R_1} + \frac{V_{out} - V_x}{\frac{1}{j\omega C_1}} - \frac{V_x - V_+}{R_2} = 0 \quad (30)$$

$$V_x = (1 + j\omega R_2 C_2) V_+ \quad (31)$$

$$V_+ = \frac{R_3}{R_3 + R_4} V_{out} \quad (32)$$

By substituting (31) and (32) into (30) and rearranging the equation, it results in:

$$V_{out} = \frac{j\omega C_1}{\frac{1 + j\omega R_2 C_2}{1} \left(\frac{1}{R_1} + \frac{1}{R_2} + j\omega C_1 \right) - j\omega C_1 - \frac{1}{R_2} \frac{R_3}{R_3 + R_4}} V_{in} \quad (33)$$

After rearranging (33), and assuming $R_1 = R_2 = R$ and $C_1 = C_2 = C$ the cutoff frequency equation is derived to be:

$$f_{cut} = \frac{1}{2\pi\sqrt{RC}} \quad (34)$$

By setting the size of the capacitor to 1 μ F, the required resistance is calculated as:

$$R = \frac{1}{(2\pi(1[Hz]))^2 (1[\mu F])} \approx 159[k\Omega] \quad (35)$$

To avoid having large amounts of noise in the filter, the resistance value was chosen as 100 k Ω instead. With the values for the components determined, the circuit was built in Multisim and simulated. The resulting circuit can be observed in Figure 47 in the appendix. The filter's response is gathered by running an AC analysis simulation. The output of said simulation is shown in Figure 12.

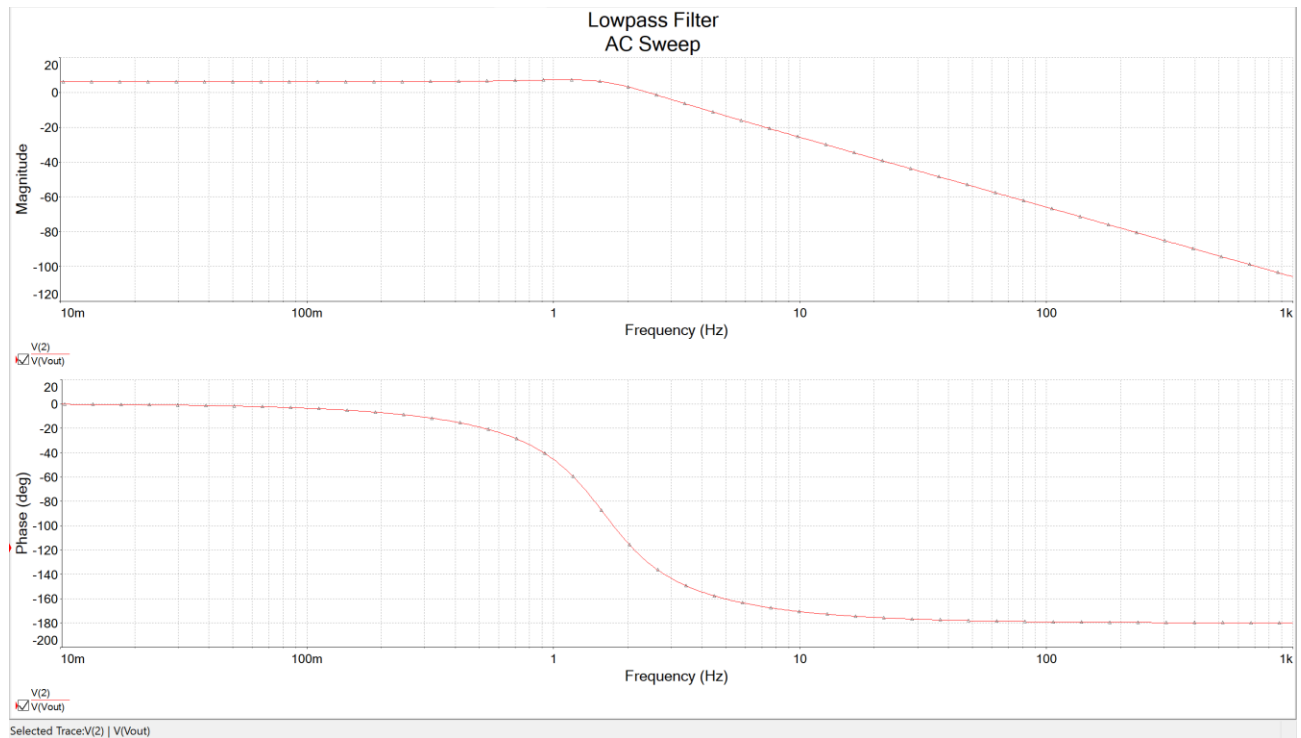


Figure 12 – Low-pass filter's response shown in a Bode Plot. The measured signal is Vout in Figure 11.

With both filters designed, their joint response must be verified. To do so, the filters are cascaded (put in series one after the other) as shown in Figure 13 and the simulation is run. The resulting response is shown in Figure 14.

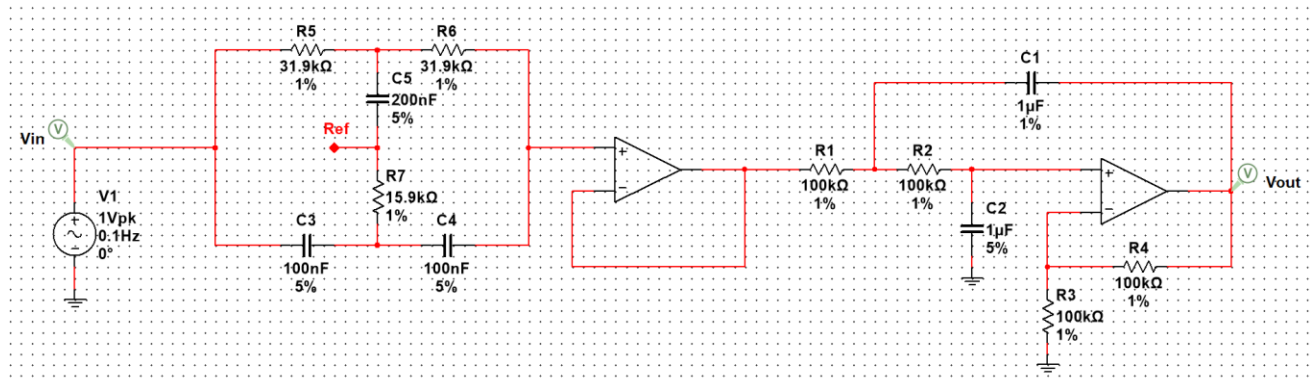


Figure 13 – Total filter schematic.

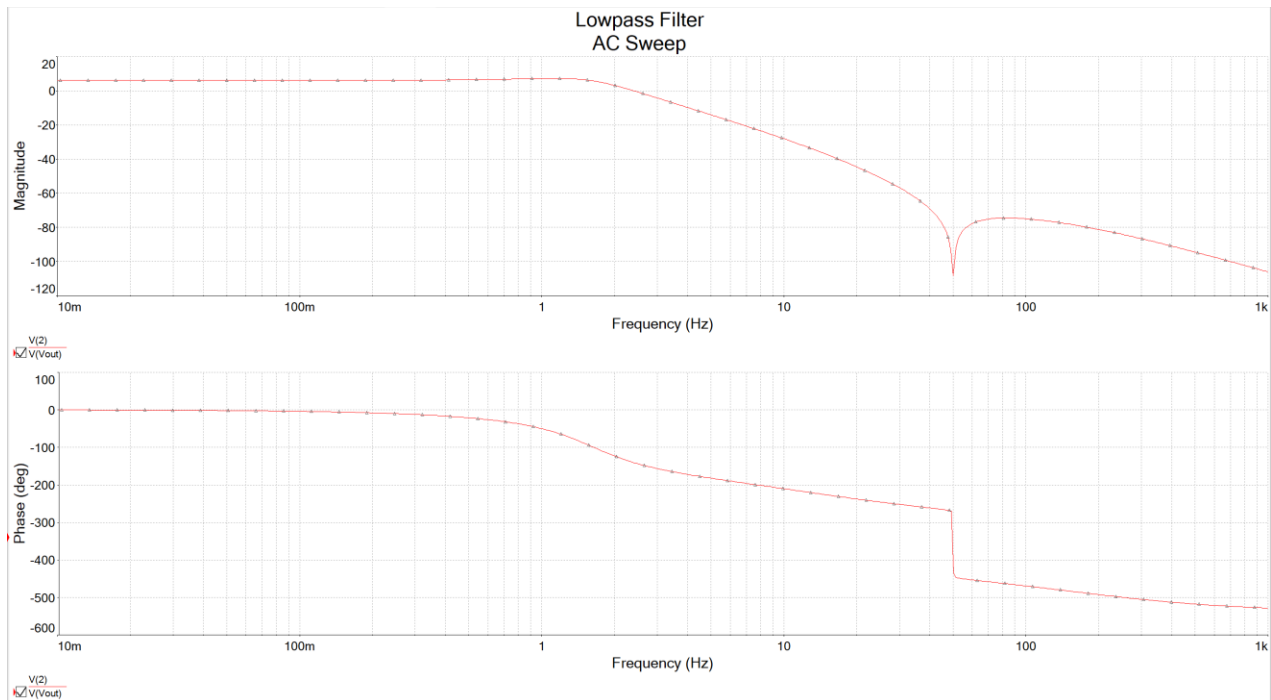


Figure 14 – Frequency response of both filters cascaded.

The first half of the sensing interface is now designed. The next section discusses the new method for REM detection: using optocouplers.

Optocoupler Readout

With the requirements set previously, many options for reflective optocouplers were found. The most suitable ones for the application were the QRE1113 and the EE-SY193. These two were chosen because of their very small footprint, low current operation, and 1mm detection distance [23]. For the following calculations and derivations, the QRE1113 will be used as standard for referencing the datasheet-required values such as operating current and output current.

The idea of using optocouplers to detect slight movements in eye rotation needs to be mathematically analyzed first. The electronic model of an optocoupler consists of a pair of a diode and a transistor, which are proportionally related by the amount of light interacting between them. The basic optocoupler symbol is shown in Figure 48 in the appendix.

A basic readout circuit for converting the output current of an optocoupler into a voltage readable by the ADC implemented inside of the microcontroller consists of biasing the diode side of the optocoupler with a current which makes the LED emit a certain amount of infrared light. The light is then able to bounce off an object and return into the phototransistor allowing a current to flow from the collector terminal into the emitter terminal. The most basic component that converts a current into a voltage is a resistor. By placing a resistor between the supply (V_{cc}) and the collector terminal in the optocoupler, the output voltage that can be obtained is proportional to the optocoupler's output current. The schematic of this setup is displayed in Figure 15.

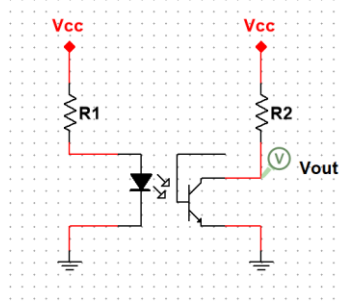


Figure 15 – Basic optocoupler readout circuit showing the output voltage on the phototransistor's side.

Out of this configuration, the bias current required for the input side (LED) on the optocoupler is determined by the diode's current-voltage relation and a resistor which limits the current flowing through the diode. The schematic model of such portion is shown in Figure 16.

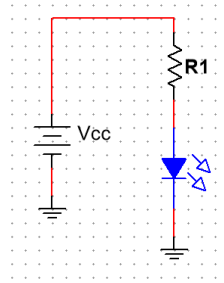


Figure 16 – Circuit to light up an LED.

Because the analytic representation, or element equation, for the LED inside the optocoupler depends on manufacturing, doping, and temperature factors, the datasheet provided by the optocoupler's manufacturer was used to determine the required current for recommended device operation. The recommended specifications for the LED's current are shown in Figure 17.

Electrical/Optical Characteristics ($T_A = 25^\circ\text{C}$ unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Units
INPUT DIODE						
V_F	Forward Voltage	$I_F = 20\text{mA}$		1.2	1.6	V
I_R	Reverse Leakage Current	$V_R = 5\text{V}$			10	μA
λ_{PE}	Peak Emission Wavelength	$I_F = 20\text{mA}$		940		nm
OUTPUT TRANSISTOR						
I_D	Collector-Emitter Dark Current	$I_F = 0\text{mA}, V_{CE} = 20\text{V}$			100	nA
COUPLED						
$I_{C(ON)}$	On-State Collector Current	$I_F = 20\text{mA}, V_{CE} = 5\text{V}^{(6)}$	0.10	0.40		mA
I_{CX}	Cross-Talk Collector Current	$I_F = 20\text{mA}, V_{CE} = 5\text{V}^{(7)}$			1	μA
$V_{CE(SAT)}$	Saturation Voltage				0.3	V
t_r	Rise Time	$V_{CC} = 5\text{V}, I_{C(ON)} = 100\mu\text{A}, R_L = 1\text{k}\Omega$		20		μs
t_f	Fall Time			20		

Figure 17 – Typical operating conditions for the QRE1113 as stated in its datasheet [23].

Having the recommended operating current is the first step in determining the necessary component values in the previously mentioned LED circuit. The relation for the value of the resistor which determines the necessary current for the LED is derived as follows:

$$V_R = I_R R_1 \quad (36)$$

$$V_R = V_{CC} - V_{LED} \quad (37)$$

$$V_{LED} = V_F = 1.2[V] \quad (38)$$

$$i_R = i_F = 20[mA] \quad (39)$$

By setting (36) and (37) equal to each other, the resulting equation can be used to derive the value for R_1 .

$$V_{cc} - V_{LED} = i_R R_1 \quad (40)$$

$$V_{cc} - 1.2[V] = (20[mA])R_1 \quad (41)$$

$$R_1 = \frac{V_{cc} - 1.2[V]}{20[mA]} \quad (42)$$

Assuming V_{cc} is 3.3 V, then the resulting value for the resistor is 105 Ω .

The output side of the optocoupler will then be influenced by how much of the light emitted by the biasing of the LED returns into the phototransistor. The current that the transistor outputs is proportional to the reflected infrared light emitted by the LED. Since the interest lies not on the exact current output, but rather whether the output current changed due to a changing distance or moving object, further mathematical analysis can be done by assuming that the proportionality constant (γ) is independent from the distance at which the object is located. Therefore, the output voltage of the circuit shown in Figure 15 is given by:

$$V_{out} = V_{cc} - V_R \quad (43)$$

$$V_R = i_R R_2 \quad (44)$$

$$i_R = \gamma i_{LED} \quad (45)$$

$$V_{out} = V_{cc} - \gamma i_{LED} R_2 \quad (46)$$

There is one disadvantage to this configuration, namely that the output resistor's value influences the rise time of the total response to an input stimulus. According to the QRE1113's datasheet, loads higher than 1 k Ω make the rise and fall time increase to more than 10 μ s [23]. Since the output current is in the range of hundreds of microamps, the resistor that converts the current to a voltage must have a high value in order to produce a noticeable output voltage. Thus, the conundrum here is that the resistor must have a high value to significantly convert the output current to a voltage while having a low value so that the output reacts as fast as the input. To solve this problem, instead of directly having a resistor convert the collector current to a voltage, a current mirror circuit can be placed in between the phototransistor's collector and output resistor. A current mirror, as its name implies, copies the current at its input to its output. The schematic of a current mirror is shown in Figure 49 in the appendix.

The model incorporates the use of a current mirror to supply current directly from the supply line to the load that converts the current created by the optocoupler into a voltage. By implementing the current mirror, the effective load as seen by the phototransistor is small while the total effect of conversion from a current to a voltage remains high due to the high valued resistance at the output. The resulting circuit is shown in Figure 18.

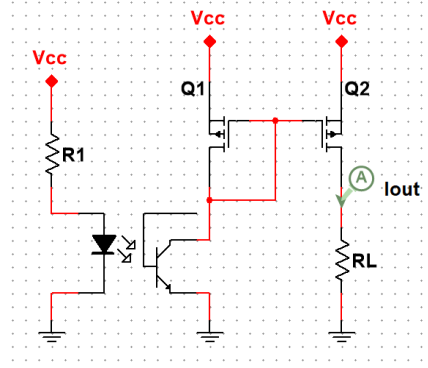


Figure 18 – Proposed optocoupler readout circuit. The output voltage node is located at the same point as Iout.

The resulting output voltage equation can be derived as shown in Figure 19.

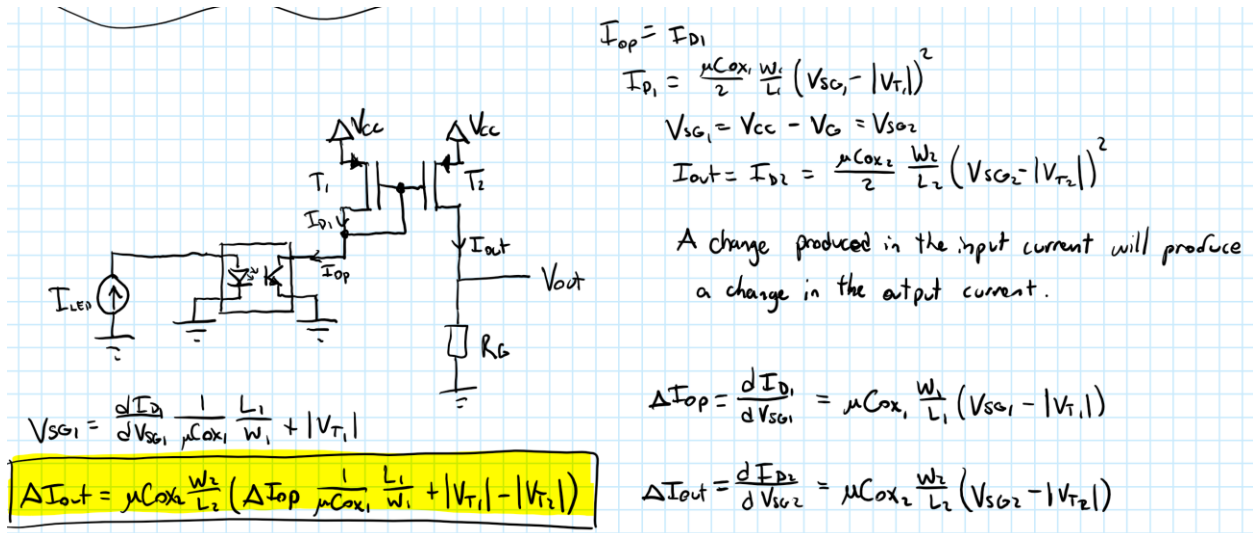


Figure 19 – Derivation for the behavior of the optocoupler readout circuit.

In order to determine the value of the necessary output resistor, it is important to consider what kind of current the optocoupler will output. By reading the datasheet and assuming that the device will be located at around 3mm away from the eyes, the output current can be determined from their performance figures. At a distance of 3 mm, the output current is around 100 μ A [23]. This current is present when the reflective object is at a fixed distance of 3 mm away from the optocoupler and is thus referred to as the steady state current. Because the steady state current will be present at all times, it must be noted that the output resistor will create a steady state voltage proportional to this current. For instance, since the optocoupler will always output 100 μ A and the output voltage must be around half of the supply to maximize output swing, the required resistor can be calculated with the following relation:

$$V_{out(steady)} = V_R \quad (47)$$

$$V_R = i_R R_L \quad (48)$$

$$V_R = \frac{V_{cc}}{2} \quad (49)$$

$$\frac{V_{cc}}{2} = i_R R_L \quad (50)$$

$$R_L = \frac{1.5[V]}{100[\mu A]} = 15[k\Omega] \quad (51)$$

Since the device may be placed closer to the eyes than expected, the value for the output resistor is chosen to be 10 k Ω to avoid clipping the signal. After mathematically analyzing the model, the circuit was created in Multisim and simulated. It is worth noting that there is no QRE1113 model in Multisim, so a similar optocoupler model was used. It is also important to mention that in order to show the behavior of a moving object in front of the optocoupler, voltage sources are used on the LED's side of the optocoupler. The circuit with its points of interest is shown in Figure 20.

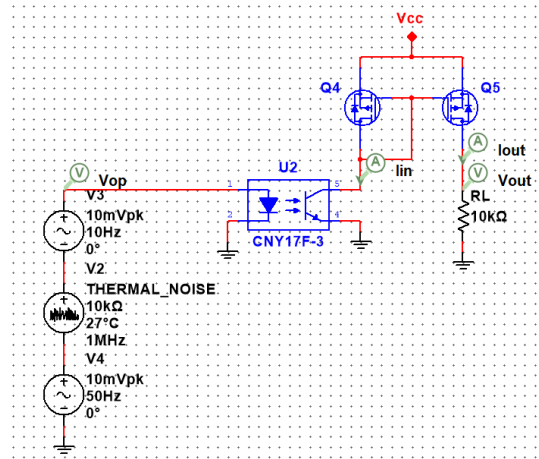


Figure 20 – Optocoupler readout circuit modelled in Multisim. There are three sources used to represent the different interference sources that the signal might have.

A transient simulation was run to demonstrate the behavior of the circuit over time and the resulting graph is shown in Figure 21.

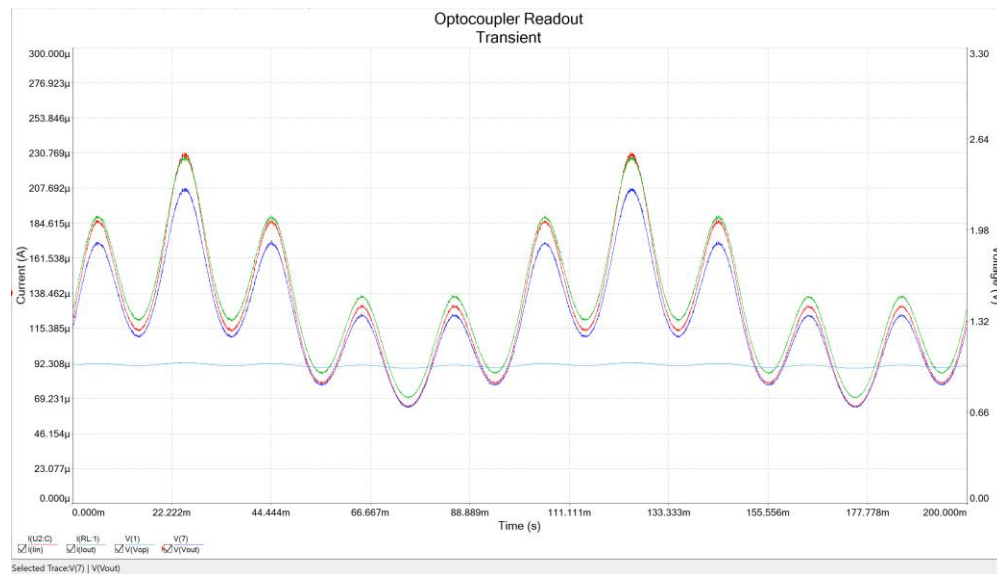


Figure 21 – Transient response of the circuit. The red signal is the current being output by the phototransistor, the green signal is the current flowing into the load resistance, the light blue signal is the voltage applied to the input side of the optocoupler which represents the eye movements, and the dark blue signal is the output voltage.

Now that the signal is properly transformed, the required filter that follows this stage can be designed.

Filtering Optocoupler Signal

As previously stated, the frequencies that are significant when detecting eye movement lie in the range between 0.3 Hz and 10 Hz. With these frequencies in mind, the cutoff frequency of a filter can be considered. The circuit mentioned above has the property that the voltage created at the output will have a certain DC offset at all times. Because the behavior of interest of the optocoupler measuring setup is the change in voltage produced by a change in reflection distance, the actual DC value of the signal is not necessary and must be removed. Therefore, the conclusion is that a high-pass filter is required.

Because the signal needs further amplification, the choice to use an active high-pass filter is made. However, because as previously mentioned, the signal needs to be low-pass filtered as well to remove all frequencies that would cause aliasing. Thus, for this measurement setup, a band-pass filter is required. The active band-pass filter configuration can be seen in Figure 22.

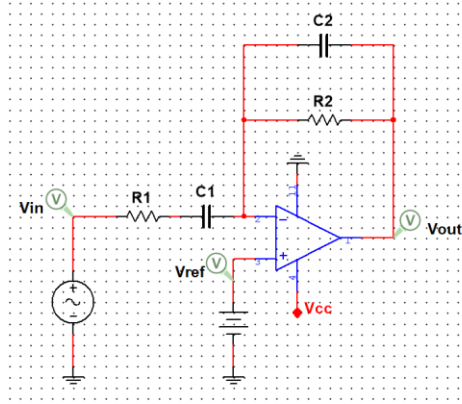


Figure 22 – Active band-pass filter circuit.

The behavior for this filter can be derived from its transfer function by analyzing the circuit in phasor domain in the following manner from (21 – 24):

$$V_+ = V_{ref}$$

$$V_- = \frac{\frac{R_2}{1 + j\omega R_2 C_2}}{R_1 + \frac{1}{j\omega C_1} + 1 + j\omega R_2 C_2} V_{in} + \frac{R_1 + \frac{1}{j\omega C_1}}{R_1 + \frac{1}{j\omega C_1} + 1 + j\omega R_2 C_2} V_{out} \quad (52)$$

$$V_{out} = \frac{R_1 + \frac{1}{j\omega C_1} + 1 + j\omega R_2 C_2}{R_1 + \frac{1}{j\omega C_1}} \left(- \frac{\frac{R_2}{1 + j\omega R_2 C_2}}{R_1 + \frac{1}{j\omega C_1} + 1 + j\omega R_2 C_2} V_{in} \right) \quad (53)$$

$$V_{out} = - \frac{R_2}{R_1} \frac{1}{1 + j\omega R_2 C_2} \frac{j\omega R_1 C_1}{1 + j\omega R_1 C_1} V_{in} \quad (54)$$

Now that the filter is drawn out, the values for the components can be related to the cutoff frequencies for the filter. Because the bandwidth of interest lies between 0.3 Hz and 10 Hz, the lower cutoff frequency $f_{cut_{low}}$ is chosen to be 0.3 Hz while the upper cutoff frequency $f_{cut_{high}}$ is chosen to be 1 Hz. The reason for choosing 1 Hz instead of 10 Hz for the upper cutoff frequency is that the gain of the

filter raises the amplification for all frequencies. Because the gain will also depend on how much the 50 Hz signal is attenuated with this filter, the upper cutoff frequency was chosen in such a way that the signal is sufficiently amplified while considerably suppressing the 50 Hz frequency.

Thus, the necessary values for the components can be calculated from previously derived equations in the following manner:

$$f_{cut_{low}} = \frac{1}{2\pi R_1 C_1} \quad (55)$$

$$f_{cut_{high}} = \frac{1}{2\pi R_2 C_2} \quad (56)$$

$$R_2 = \frac{1}{2\pi(1[Hz])(1[\mu F])} \approx 150[k\Omega] \quad (57)$$

$$R_1 = \frac{1}{2\pi(0.3[Hz])(100[\mu F])} \approx 8[k\Omega] \quad (58)$$

Therefore, the chosen values for the high-pass portion of the filter are 8 kΩ for the resistor and 100 μF for the capacitor. As for the low-pass portion of the filter, the values were determined to be 150 kΩ for the resistor and 1 μF for the capacitor. This configuration will also allow a voltage gain (A_V) proportional to the ratio between the two resistors, namely $A_V = \frac{150[k\Omega]}{8[k\Omega]} = 18.75$.

Processor

One design requirement for the sleep monitoring device is having the need to be as small as possible to not be obtrusive to the user when using it. Because of this requirement, the processor needs to comply with the size constraints. Therefore, an ATtiny85 from Atmel with a small footprint was considered.

The ATtiny85 has a 20 MHz clock, 5 I/O ports, 20 KB of flash storage, and 512 bytes of Random-Access Memory (RAM) [24]. These specifications seemed to be capable of handling all the math necessary to process the four signals being measured with the device. However, upon further testing, the amount of RAM this processor has is not enough to compute a Fast Fourier Transform (FFT) analysis of size greater than 128 points. The reason for this is that first, the signals read with the ADC must be stored in RAM as well. The size of the signals also influences the amount of RAM used. Therefore, in order to store and compute four different signals, more RAM is required.

After concluding that the ATtiny85 would not be a suitable choice of processor for the sleep monitoring device, other options were researched. The suitable alternative found was the STM32F1 from ST Microelectronics.

An STM32F1 operates at a 72 MHz clock frequency, 37 I/O Ports with two 12-bit, independent ADC channels, 128 KB of flash storage, and 20 KB of RAM [25]. All these features provide a significant improvement on the capabilities of the processing interface, all the while keeping a small form factor.

The STM32F1 also includes an internal RC oscillator which, in combination with its internal Phase-Locked Loop (PLL), can provide the required clock signal instead of relying on an external crystal oscillator.

Communication Interface

The sleep monitoring device must be able to either save or send the data it gathers throughout use to another device for further processing and corroboration. There are a few ways this can be done. One example is by using a simple USB connection to a computer. This method, however, is not ideal since the USB cable would be obtrusive to the user when in use during sleep. Therefore, a wireless solution would prove best in terms of obstruction.

For this reason, an HC-05 Bluetooth module is chosen. This module provides a wireless serial communication interface. With this interface, the data can be sent directly from the processor to the external device – in this case, a laptop running MATLAB.

The Bluetooth module is controlled via the UART communication protocol. This protocol allows asynchronous data transmission between devices with only two wires – RX and TX [26]. This is useful since it provides a bridge between devices for data transfer without the need to coordinate a start and an end signal for transmission.

Feedback Interface

For the feedback interface, different options were considered. By following LaBerge's previously mentioned research, it was determined that visual feedback or visual stimulation is the most promising feedback method. Thus, for feedback, flashing LEDs will be used. The color of the LEDs might also play a role in the effectiveness of having the stimulus appear in a dream. However, this question is left for further research on a different project. For the purpose of this device, bright, true-green LEDs with a small footprint of 1.5 mm by 0.8 mm were chosen.

The necessary circuit is identical to that of the optocoupler's. Because both circuits involve biasing an LED, previously calculated equations can be used to determine the necessary values. According to the LED's datasheet, the LED's forward voltage is typically 3.3 V at 20 mA of current flowing through the LED [27]. However, since the entire system's voltage will not surpass 3.3 V, if the LED were to be connected directly to one IO pin on the processor, it would work normally since the voltage output by the processor is 3.3 V. The IO pins in the processor are current-limited and typically limited to 20 mA [25]. For this reason, any small valued resistor would work fine in order to more carefully protect both the LED and the IO pin from burning due to overcurrent usage. Thus, the LED's resistor value was chosen to be 100 Ω .

Power Management

Another one of the hardware design requirements for the device is that it should have low power consumption. With a low power consumption constraint, the device should be powered with a small battery. Because of the low power usage constraint, the device must operate at a low voltage. Since most rechargeable batteries have an operating voltage range from 3.7 V to 2.6 V, the device should work at a voltage somewhere in between those two values.

The STM32F1 and the HC-05 Bluetooth module have a typical operating voltage of 3.3 V. Therefore, it is decided that the device should have a regulated voltage of 3.3 V. In order to regulate the battery's voltage to the desired level, a good voltage regulator must be chosen. For this application, a linear, low-dropout voltage regulator is the ideal choice since it provides low noise and a stable 3.3 V output, it has low power consumption, and it has a small form factor. These regulators are usually referred to as LDO's.

In order to minimize noise in the sensors' measurements, their supply voltage must be as clean and stable as possible. For this reason, two independent LDO's will be required: one for the analog filters

and sensors, and the second one for the digital, high-speed components such as the Bluetooth module. This configuration will prevent the digital ‘noise’ created by the digital communication between the processor and the Bluetooth module from polluting the sensed signal in the analog filters. The chosen LDO for this device is the NCP612 because of its small size, output voltage accuracy of 2%, and low power consumption [28].

For the same reason, multiple ‘virtual ground’ points for biasing the filters must be considered. By using different references for the bias voltage in the filters, crosstalk between said filters can be minimized. The term crosstalk refers to either currents or voltages from one path influencing or affecting the measurement in another path. Thus, three different ‘virtual ground’ connections are designed: one for the initial amplifier stage in the EOG detector, the second one for the filter after the amplifier stage, and the third one for the filter affecting the optocouplers’ signal.

Creating a reference voltage, or ‘virtual ground,’ can be done in different ways. One method is by applying the voltage divider equation and making a voltage divider out of resistors in order to set a desired voltage based on the ratio between the resistances. This method, however, relies heavily on the components’ values and if they are not perfectly matched to the theoretical values, then the behavior of the circuit might not be the desired one. Another disadvantage this method has is that it is more susceptible to noise on the supply lines than any other method. For reference, a voltage divider circuit is shown in Figure 50 in the appendix. The equation that shows the behavior of the output voltage is:

$$V_{out} = \frac{R_2}{R_1 + R_2} V_{cc} \quad (59)$$

A third disadvantage that this circuit has is that it is strongly susceptible to loading effects. The schematic diagram for a loaded voltage divider is shown in Figure 23.

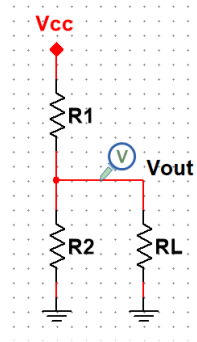


Figure 23 – Loaded voltage divider circuit.

For instance, if a load with resistance lower than that of R_2 is connected to the output, then the voltage will drop significantly since the new equivalent resistance at play is given by the following equation:

$$R_{eq} = R_2 // R_L \quad (60)$$

$$R_{eq} = \frac{R_2 R_L}{R_2 + R_L} \quad (61)$$

Assuming $R_2 \gg R_L$, then the following holds

$$R_{eq} \approx \frac{R_2 R_L}{R_2} = R_L \quad (62)$$

Therefore, the output voltage drops depending on the value of the load resistance.

A solution to these problems lies in using a different component altogether. A Zener diode, or reference, is a semiconductor device that if used in forward bias mode (positive voltage applied to the anode), it behaves as a normal diode. However, if used in reverse bias mode (positive voltage applied to the cathode), it has the property of ‘breaking down,’ or allowing current to flow once a certain voltage threshold is crossed. This causes the diode to force the voltage at its cathode to stay at its specified breakdown voltage. A circuit with the Zener diode reference is shown in Figure 51 in the appendix.

Thus, this circuit is less susceptible to noise on the supply lines, more stable, and loading effects only play a role for very small loads. However, to reduce loading effects even further, an amplifier in a buffer configuration can be placed at the output of this circuit resulting in the schematic shown in Figure 24. The chosen reference for this project is the TS4061 because of its low operating power consumption and high accuracy and stability [29].

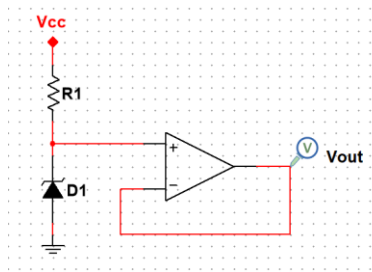


Figure 24 – Voltage reference circuit with output buffer stage.

Further expanding on the device requirements, it was previously mentioned that the device should operate throughout the night, when the user is sleeping. Therefore, this requirement can be interpreted as the device needing at least an 8-hour battery life. With these constraints, it is now possible to determine the required battery specifications.

In order to calculate the necessary battery capacity, two assumptions must be made. The first assumption is that all the components in the device will operate around their maximum current draw. This will ensure that the battery lasts for the set amount of time while consuming the most power possible at once. The second assumption is that the advertised battery capacity is indeed the true capacity of the battery.

A battery’s capacity is given in milliamp-hours, that is the number of hours the battery would last, provided it is constantly discharged at 1 mA. Therefore, to calculate the required battery capacity, one must apply the following equation:

$$Capacity[mAh] = i_{discharge}[mA] \times t[hours] \quad (63)$$

By reading each of the component’s datasheets, the total maximum current draw from the device was determined to be around 100 mA. This current is calculated by summing all the maximum currents per active component – OpAmps, InAmps, regulators, processor, and Bluetooth module. Therefore, if an 8-hour battery life is desired, the necessary battery capacity should be at least 800 mAh. Therefore, a lithium-polymer battery with a capacity of 1250 mAh was chosen. The extra capacity will guarantee that the device works for the intended amount of time.

Software

Another important part of the sleep monitoring device is the software. Without software, the device would not be able to apply algorithms to calculate and detect REM. This subsection details the choice of processing application for the first testing stage, the code running on the sleep monitoring device, the proposed REM detecting algorithm, and the choice of digital filtering.

MATLAB Connection

As a first prototype version of the software, all the processing of raw data was decided to be done in MATLAB. The reason for this setup is the fact that it allows testing and modifying the algorithms faster and more easily since the processor in the sleep monitoring device does not have to be constantly reprogrammed.

MATLAB has an internal function that allows it to set up and gather data from a virtual COM port [30]. Via this port, the device can connect to the computer through the Bluetooth module. A COM port is a two-way channel that serves as a tunnel to both transmit and receive data. Thus, by relaying the data from the device to the computer running MATLAB, real-time processing can be done as if the device were processing the data itself.

Programming Processor

Programming the sleep monitoring device is done via the STM32Cube IDE. This software includes basic libraries required to program an STM32 processor. The code running on the device should gather measurements through the ADC ports, store the values in memory, and finally, transmit the stored values via the Bluetooth module. The code used to read data from the ADC ports and send that data through the UART interface via the Bluetooth module is attached to the appendix.

Algorithm

The proposed algorithm is based on a threshold detection mechanism. It consists of gathering and processing data in 30-second intervals. The basic idea is that for every 30-second interval, a power spectral estimation is done. This estimation allows for the frequency contents of the received signal to be measured. Afterwards, the computed PSD is integrated, and the value is stored. Then, the signal measured over the 30-second interval is high-pass filtered with a digital filter to remove any low frequency interferers. Next, the PSD of the filtered signal is computed and integrated. A Boolean operation is done next. If the ratio between the newly calculated PSD and the original PSD is higher than 55%, then the signal most likely represents REM. This means that if more than half of the signal's power in a 30-second interval is contained within the bandwidth of interest (0.3 Hz to 10 Hz) then the user is undergoing REM. The flowchart that illustrates the algorithm is shown in Figure 52 in the appendix.

Digital Filtering

A digital filter is required to remove any interference that may still be present in the measured signal. The interference might include, but is not limited to a DC offset, head movements, heartbeat, and sliding or shifting of the device. Since these interference examples have low frequency behavior, a high-pass filter is required. As stated before, the lower cutoff frequency should be 0.3 Hz. To create the digital filter, the `highpass` function in MATLAB is used. This function creates a minimum order filter with a stopband attenuation of 60 dB [30]. This filter is ideal for this application because the only other interferers present in the signal are low frequency components.

Validation

After designing the necessary parts for each of the blocks seen in the system diagram, the physically built components must be tested and verified. This section focuses on showing that the built parts function as intended. All figures for this section can be found in the appendix.

Hardware

Supply Voltage Regulation

To test whether the voltage regulator works, the device is given power through the input terminals while the output is measured with a Micsig TO1104 oscilloscope. The input voltage supplied to the regulator is 5 V. The expected output of the LDO is 3.3 V. Figure 53 shows the result captured by the oscilloscope. As it can be seen from the figure, the output voltage is 3.305 V. This voltage is within specifications and is thus suitable for the sleep monitoring device.

Virtual Ground

In order to evaluate the virtual ground circuit, the device is powered via the LDO. For reference, the circuit in Figure 24 is supplied with 3.3 V at its V_{CC} terminals and the output located after the voltage buffer is measured with the oscilloscope. The resulting plot is shown in Figure 54. The expected output voltage of this circuit is 1.225 V as given by the TS4061's datasheet [29]. The oscilloscope reads a voltage of 1.230 V. A difference of 5 mV from the expected voltage will not hinder the behavior of the circuit in any way. Therefore, this voltage is acceptable.

Optocoupler Readout

To evaluate the optocoupler's readout circuit, a controlled test was carried out. The test involves placing a piece of paper 3 mm away from the optocoupler and measuring the output of the circuit with an oscilloscope. The piece of paper was vertically held in place with a pair of pliers. The output was recorded and is shown in Figure 55. The figure shows the steady state voltage output by the circuit at 1.156 V. As expected, the voltage is lower than half of the supply because of the choice made to change the output resistor's value from 15 k Ω to 10 k Ω .

Additionally, a different test was carried out to verify the dynamic behavior of the circuit. This test consisted of bumping the piece of paper located at 3 mm away from the optocoupler so that it would oscillate in harmonic motion. The behavior of the circuit can be observed in Figure 56. Because the oscilloscope's output follows the harmonic motion, the conclusion is that the optocoupler effectively detects small changes in distance which can afterwards be amplified for a better readout.

Band-pass Filter

Testing of the band-pass filter located at the output of the optocouplers' readout circuit was conducted with a My-DAQ, a data acquisition device developed by National Instruments. The My-DAQ has multiple features that are helpful for validating the designed circuits. It features an arbitrary function generator, an oscilloscope, a Bode Analyzer, and a dynamic signal analyzer.

Since the Bode Analyzer's minimum frequency output is 1 Hz, the low cutoff frequency of 0.3 Hz cannot be evaluated. Figure 57 shows the upper portion of the filter's response as gathered by the Bode Analyzer. To plot and observe the low frequency behavior of the circuit, a different approach was taken to measure the output of the filter. For this, the function generator in combination with the oscilloscope applications were used. The minimum frequency that can be output by the function generator is 0.1 Hz. The procedure was to output a sine wave with an amplitude of 25 mV for each frequency starting from 0.1 Hz until 100 Hz through the function generator. The signal is observed through the oscilloscope in

channel 0. Meanwhile, channel 1 measures the output of the filter. With cursors on the oscilloscope's screen, the output voltage's amplitude can be determined.

The different resulting amplitudes were recorded and put into MATLAB. Afterwards, a magnitude plot was made on a logarithmic scale. This plot follows the expected theoretical response for the filter since it features the proper cutoff frequencies. The resulting plot is shown in Figure 58.

EOG Amplification

To verify the instrumentation amplifier's stage, the MyDAQ's function generator application along with the oscilloscope were used. An input voltage with an amplitude of 5 mV was applied to the input of the amplification circuit. The oscilloscope's screen shows channel 0 is the input voltage while channel 1 is the output voltage. Figure 59 shows the output voltage is clipping. This is due to the fact that the amplification factor was designed to be 513. This means the input voltage of 5 mV is amplified 513 times and results in an output voltage amplitude of 2.565 V. Since the minimum amplitude that can be output by the MyDAQ is 5 mV, the output cannot be further verified without clipping.

Notch Filter

The Bode analyzer function was used in order to evaluate the notch filter's response. The resulting plot is shown in Figure 60 and closely resembles the theoretical behavior of the filter. The difference between the measured response and the theoretical response is that low frequencies in the physical filter have an attenuation of 5 dB. This may be due to the output signal clipping as shown in Figure 43. Despite this fact, the evaluation of this portion of the device is considered successful.

Low-pass Filter

In order to validate the low-pass filter that follows the notch filter, the Bode Analyzer application is used. The results are shown in Figure 61. The cutoff frequency of the filter is determined to be around 2 Hz instead of 1 Hz. This is due to the fact that the calculated resistance was not used, but instead a resistance of 100 kΩ. The important conclusion, however, is that this mismatch between frequencies is small and it does not influence the overall filter's behavior negatively.

Joint Filter Response

In a similar manner as previously stated, the response for the combination of both filters was gathered through the Bode analyzer. It is worth noting that the complete response does not match the theoretical expected values. The reason for this lies in the measurement capabilities of the My-DAQ. The input voltage amplitude applied was 0.5 V. At the point shown by the cursors in Figure 62, where the magnitude reaches -56 dB, the signal has undergone a reduction in amplitude of 630 times. This means that the applied input amplitude is divided by 630 which results in $V_{out} = \frac{0.5[V]}{630} = 793[\mu V]$.

According to the My-DAQ's datasheet, the typical absolute accuracy for a measurement range of 2 V is 4.9 mV [31]. Therefore, it is safe to conclude that 793 μV is too small a voltage to measure accurately and for this reason, the measured Bode Plot looks distorted.

Software

Data Communication

The verification of the connection between the sleep monitoring device and MATLAB consists of running the code attached to the appendix. This code sets up the virtual COM port that allows the data transmission between the sleep monitoring device and MATLAB. Then, the function readline() reads all

the bytes available on the COM port line by line and the data is stored in an array. Finally, the data is plotted in a figure that shows the read data versus sample number. This graph is shown in Figure 63.

Algorithm

In order to validate the proposed algorithm, a test signal consisting of a REM-frequency sinusoid, a low frequency interferer, and added white noise is applied as an input to the algorithm. The signal can be seen in Figure 64. The algorithm then analyzes the signal and outputs either a 1 or a 0 into an array depending on the REM state it determines. A 1 represents REM detection, while a 0 represents no REM. The resulting figure for the output of the algorithm is shown in Figure 65. From this figure, it can be concluded that the REM detection algorithm works.

Results

The current section displays the manufacturing process of the sleep monitoring device while presenting the tests run in order to evaluate the behavior of the device. The section is broken down into the Computer-Aided Design (CAD) layout of the device, the 3D model of the design, the manufactured board, the connection between the device and computer, the tests run to evaluate the algorithm, and the evaluation of the feedback mechanism.

Eagle CAD Design

With the previously mentioned circuits designed and simulated, the schematics are drawn out in Eagle, a CAD software made by Autodesk. The total schematic of the device is attached to the appendix of this document. Figure 25 shows the designed PCB layout.

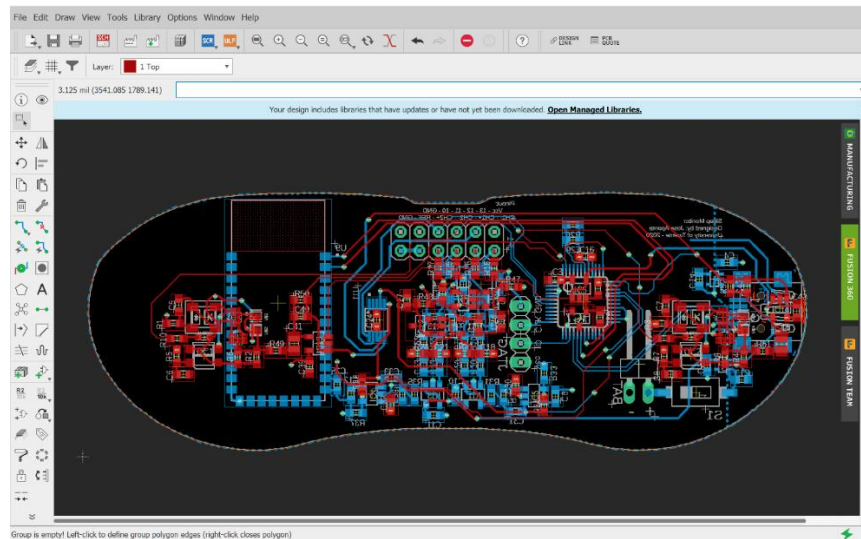


Figure 25 – PCB layout design created in Eagle.

While designing the layout for the PCB, the requirement that the device must have a small size and be comfortable to wear was considered. Therefore, the PCB was shaped in the form of a sleeping mask (blindfold). This shape allows the PCB to have a small size and be placed over the eyes comfortably. The device can be secured in this position with the use of a sleeping mask. The size of the designed PCB is 96.6 mm by 35.5 mm.

As it can be seen from the design, the board was made to be modular. This means that two different optocouplers can be used (not simultaneously) to test the device. The board also includes a space to place an external ADC with an I2C communication interface if there is a need for it. The PCB also includes multiple contact points to evaluate some communication signals. The inclusion of a mini USB port is due to the fact that an STM32 processor can be programmed with a bootloader that accepts the USB communication protocol for easier prototyping. After the CAD design was finalized, the board file was exported to Fusion 360 where a 3D model of the board was made.

3D Render

A 3D render of the PCB was made in Fusion 360, a 3D modelling software made by Autodesk. The reason for making a 3D model is budget approval. With this 3D model, a stronger appeal can be made that will help with acquiring approval for the necessary budget to build this device.

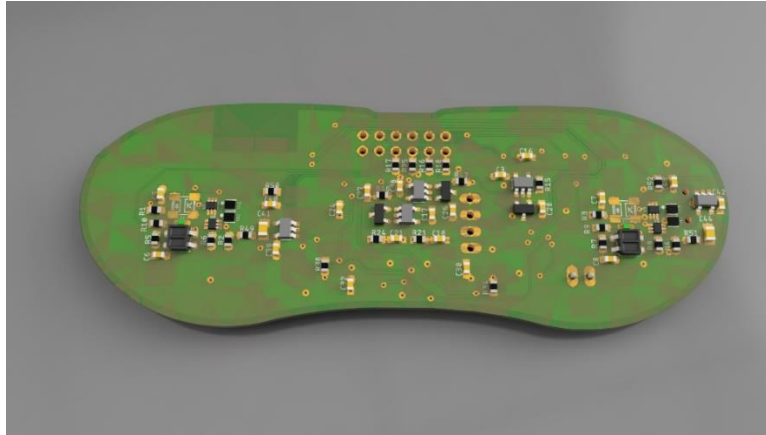


Figure 26 – Top view of the rendered 3D model of the designed PCB.

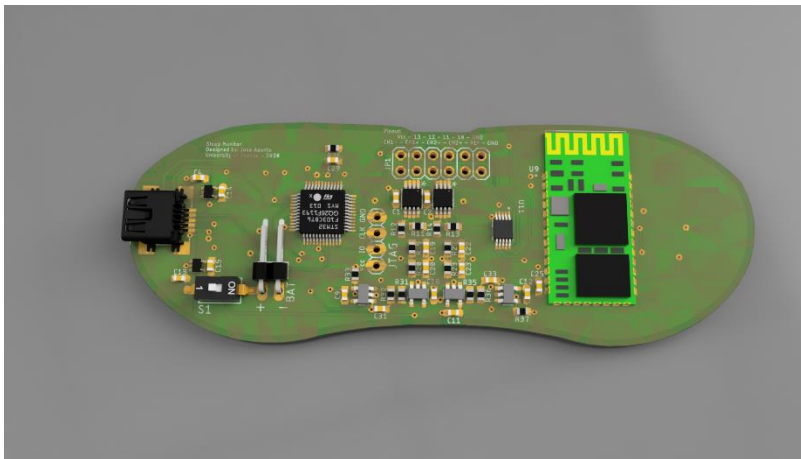


Figure 27 – Bottom view of the rendered 3D model of the designed PCB.

Figure 26 and Figure 27 show the rendered 3D model of the designed PCB. With the 3D modelled version of the PCB that includes all the components required on the board, the conclusion can be drawn that this board has a proper size in all three dimensions. With the components placed, the device has a thickness of 3.1 mm from the bottom side of the board to the tallest components on the topside – the optocoupler. The top side of the board is the side facing the eyes of the user. Thus, with a thickness of 3.1 mm, the board will sit comfortably above the eyes of the user without making contact with them.

PCB

The Printed Circuit Board (PCB) was ordered via JLCPCB, a Chinese PCB manufacturing company. The received boards are of good quality and show no signs of imperfections in the copper traces. The PCB is shown in Figure 28 and Figure 29.

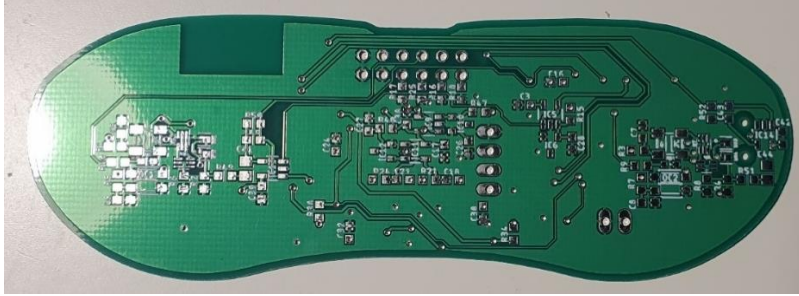


Figure 28 – Top side of the manufactured PCB.

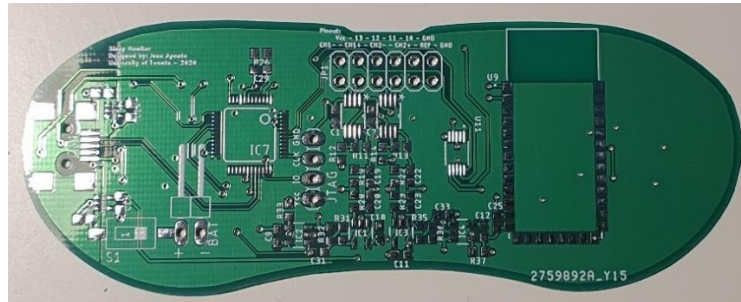


Figure 29 – Bottom side of the manufactured PCB.

The PCB was ordered to have a thickness of 0.6 mm. A small board thickness will ensure that the device sits comfortably above the subject's face because of its flexibility. After the PCB was received, the electronic components were soldered onto it. Figure 30 and Figure 31 show the finished sleep monitoring device.

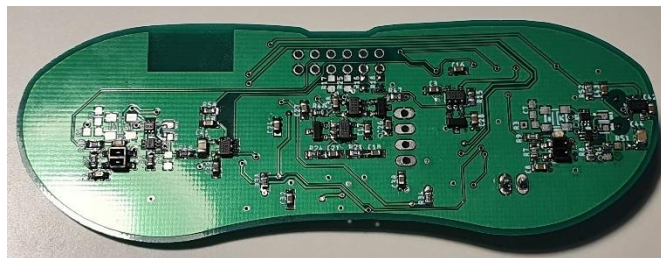


Figure 30 – Top view of the finished (soldered) sleep monitoring device.

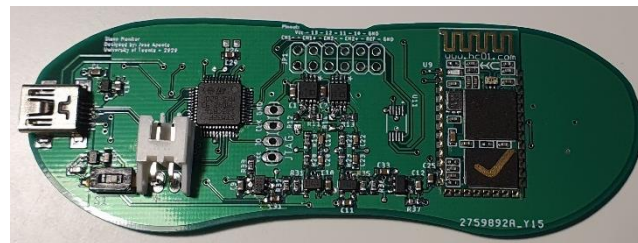


Figure 31 – Bottom side of the completed (soldered) sleep monitoring device.

The finished device is then fitted into a sleeping mask with grooves for the eyes. This type of sleeping mask is ideal for this purpose because it offers some clearance space between the eyes and the mask itself. The final product is shown in Figure 32.

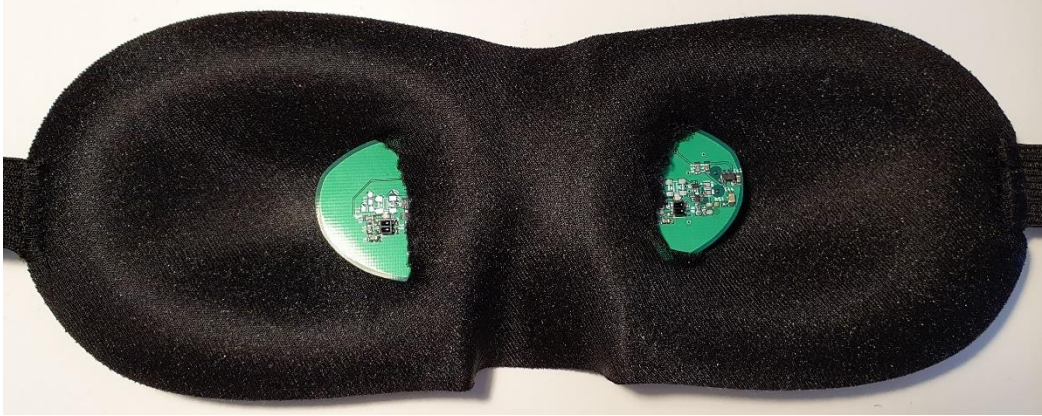


Figure 32 – Sleep monitoring device implemented in the sleeping mask.

Communication with MATLAB

Since the values transmitted by the sleep monitoring device are digital levels, these values must be mathematically converted into an analog voltage by multiplying the received value with $\frac{3.3[V]}{2^{12}}$. Once the values are converted, the resulting voltage is plotted against time. The resulting graph is shown in Figure 33.

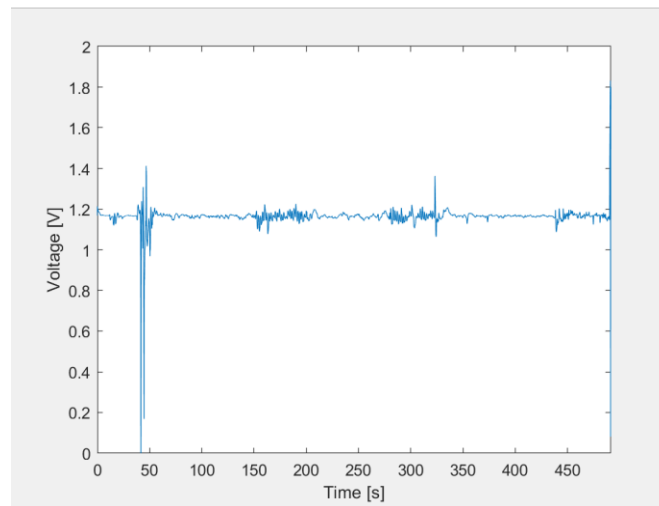


Figure 33 – Test signal involving three instances of REM.

This figure verifies that the connection between the sleep monitoring device and MATLAB works. The first oscillation in the signal is due to the user placing the device over the eyes. The rest of the increased-activity zones correspond to the experiment carried out. With the connection established between the two devices, further testing can be conducted. It is important to note that the sampling frequency of the ADC's interface is set to 31.25 samples per second. The reason for having this frequency is that the delay function for the STM32 plus the delay the device has when calling the sampling process in the processor add up to a delay of 32 ms.

Algorithm

The experiment that verifies the behavior of the algorithm involves a tight control scheme. This experiment produced the input signal shown in Figure 33. The experiment consisted of wearing the sleep

monitoring device while not moving the eyes as if trying to fall asleep for two and a half minutes. Afterwards, the eyes were moved from left to right in sync with a metronome playing at 60 bpm for 60 seconds. Then, the eyes were kept still for one minute. Then, the eyes were moved to the same metronome beat for one minute. Next, the eyes were kept still for a third time for two minutes. Finally, the eyes were moved with random speed and frequency for one minute. The experiment's data was then analyzed in MATLAB.

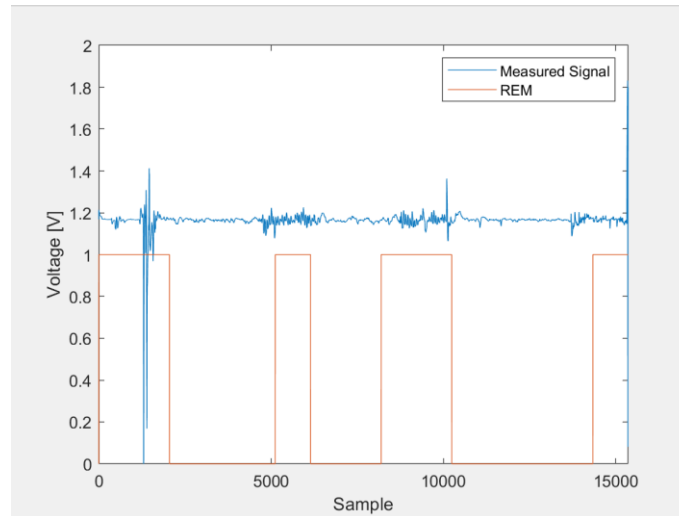


Figure 34 – Results of the REM detecting algorithm to a controlled signal input. There are three instances where REM happens – at 5000 samples, 9000 samples, and 13000 samples.

After inputting the recorded test signal to the REM-detecting algorithm, the output was produced and is shown in Figure 34. As it can be observed from the figure, the REM portions specified by the algorithm closely match those of the input signal. This result suggests that the algorithm can work when implemented in the device itself. Additionally, a test measurement was taken with actual sleep involved. This experiment consisted of placing wearing the device and sleeping for around one hour. After waking up, the data was interpreted. The output of the REM algorithm to the sleep input signal is shown in Figure 35.

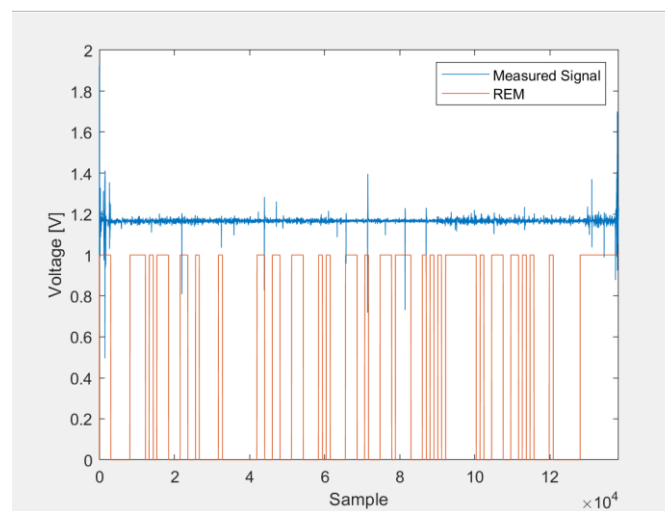


Figure 35 – REM algorithm's output with the sleeping user signal as an input.

In this instance, the algorithm is riddled with noise. However, there are two distinct sections where the width of the detected REM periods is greater than usual. This result suggests that REM was in fact detected. The input signal shows a smaller amplitude in the REM portions due to the fact that the device may have shifted from the right position during the experiment. Further testing should rule out this theory.

Feedback Mechanism

The implemented feedback mechanism consists of flashing LEDs. Figure 36 shows the chosen LEDs turned on upon startup. It is important to note that the effectiveness of the dynamic behavior of the feedback system is left for future work. The evaluation for whether constant lights is better or worse than blinking lights at inducing lucid dreaming is not discussed in this thesis due to a lack of time.

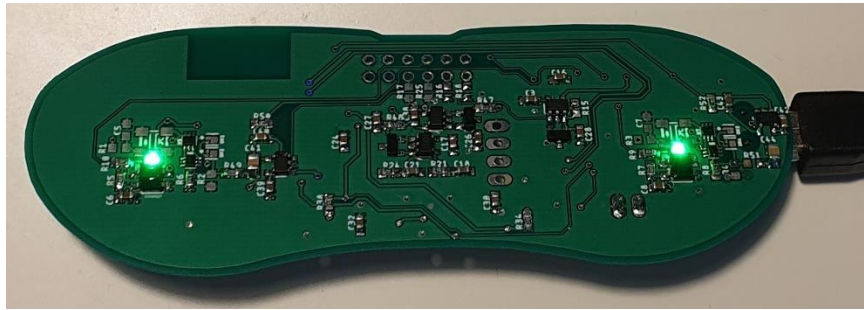


Figure 36 – Feedback LEDs flashing when REM sleep is detected.

While some aspects of the device were left out of this research project due to time constraints, it is important to note that these aspects were originally considered and are explained in the following section.

Discussion

The results shown above prove that it is feasible to use optocouplers to measure eye movements. After evaluating all the results, however, there are some points worth discussing. The measurements done with the interaction between the sleep monitoring device and MATLAB are only taken by one ADC channel. This ADC channel was measuring the output of the left optocoupler. The reason for only using one data channel instead of all four available is that the code required for multiplexing ADC channels in the processor was more complex than expected and it would not work. Therefore, the measurements were kept in the one available channel.

By not being able to gather more data from the rest of the available sensors, a ‘stronger’ or more reliable algorithm based on comparing all the signals available amongst each other was unable to be created. Despite this, the data gathered shows that with only one source available, the algorithm is still able to detect REM to some extent. The actual accuracy at which the algorithm detects REM sleep could not be evaluated since many more measurements are necessary. Additionally, another part of the algorithm that was originally considered, but omitted is the REM ‘counter’. This layer of the algorithm would only categorize REM as sleep REM if 3 or more measurement periods were detected to have REM. Due to lack of time, this layer could not be implemented.

Despite the fact that some aspects of the device did not work out as expected, the sleep monitoring device already has some positive advantages over the projects mentioned earlier. The sleep monitoring device is comfortable to wear, it is not obtrusive since it is worn as a sleeping mask, and, with its long battery life, the device can measure all throughout the night.

These milestones were achieved by depending on an external device, however. While it is true that the end goal of the sleep monitoring device is to be a standalone product capable of processing all the measured signals by itself, this goal was not achieved in the span of this thesis.

There is still room for improvement, however. For instance, the current algorithm only works if a low frequency interferer is present in the signal. A solution to this problem could not be found within the available time. Another problem the device presented was the fact that all measurements depended on the connection between the sleep monitor and MATLAB. If this connection was broken for any reason, the device would stop transmitting the data since there was no port available. Then, the data would not represent the experiment done accurately due to being ‘cut’ short.

As a solution to some of the problems encountered, future work on the sleep monitoring device will be continued. This work includes the translation of the MATLAB algorithm into C so that the STM32 can be programmed with it and the standalone version of the device can be tested. Additionally, the code for multiplexing the ADC channels in the STM32 will be explored further. This will allow the use of all the sensors present in the device after which some sensor fusion can be introduced. As an added layer of stability, the algorithm can be designed according to a Markov Model. Since the occurrence of REM is different in all subjects and at any point during the night, this process can be considered random and therefore, better modelled with a Markov Model. Finally, in order to verify that the device works outside of ‘lab’ conditions, the device must be tested with multiple subjects. This will increase the amount of data available to process, as well as different reference points to compare and build a stronger algorithm out of.

Finally, as mentioned earlier, the extent to which lucid dreaming can be induced is left for future work. Once the device is able to successfully detect REM sleep in standalone mode, overnight testing can be done. By wearing the device to sleep, if the device works properly, the effect of whether or not the

feedback method (light) can be recorded and reported. Additionally, the influence of the color of light can be studied to find the ideal combination of blinking, or steady colored light.

Conclusion

The results gathered by the experiments performed in this thesis confirm that designing a small and comfortable REM-detecting sleep monitoring device is possible. The device was successfully built according to specifications and all the set requirements were met. Moreover, the use of optocouplers to detect eye movements was successful. While it is true that the extent to which lucid dreaming can be induced is left for future work, it is the final intention behind this thesis. With the sleep monitoring device built, further sleep data can be gathered with an ease that could benefit research groups in conducting sleep studies. Finally, the product development of the sleep monitoring device will continue as previously discussed and will keep gathering more sleep data and polishing the REM-detecting algorithm to, in the end, have a standalone product capable of inducing lucid dreaming.

References

- [1] B. Wilfong, "Lucidity," The Lucidity Institute, 4 January 1999. [Online]. Available: <http://www.lucidity.com/vanEeden.html>. [Accessed June 2020].
- [2] S. LaBerge and H. Rheingold, *Exploring the World of Lucid Dreaming*, New York: Ballantine Books, 1991.
- [3] S. LaBerge, "Lucid Dreaming: Psychophysiological Studies of Consciousness During REM Sleep," *Sleep and Cognition*, pp. 109-126, 1990.
- [4] F. F. Ellertsen, "Fredrik Flornes Ellertsen," 25 April 2016. [Online]. Available: <https://fellerts.no/projects/eog.php>. [Accessed June 2020].
- [5] J. Choi, "Hackaday," 23 August 2016. [Online]. Available: <https://hackaday.io/project/13285-openId-lucid-dreaming-research-platform>. [Accessed June 2020].
- [6] A. H. Horowitz and O. Rosello, "MIT Media Lab," 2019. [Online]. Available: <https://www.media.mit.edu/projects/sleep-creativity/overview/>. [Accessed June 2020].
- [7] "Electroencephalogram (EEG)," Johns Hopkins Medicine, 2019. [Online]. Available: <https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/electroencephalogram-eeg>. [Accessed June 2020].
- [8] "EOG," ADInstruments, 2020. [Online]. Available: <https://www.adinstruments.com/signal/eog>. [Accessed June 2020].
- [9] "Electromyography (EMG)," John Hopkins Medicine, 2019. [Online]. Available: <https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/electromyography-emg>. [Accessed June 2020].
- [10] C. Armon, "What Is the Role of EMG Channel In Polysomnography (PSG)," Medscape, 29 April 2020. [Online]. Available: <https://www.medscape.com/answers/1188764-193025/what-is-the-role-of-emg-channel-in-polysomnography>. [Accessed June 2020].
- [11] "ECG," ADInstruments, 2020. [Online]. Available: <https://www.adinstruments.com/signal/ecg>. [Accessed June 2020].
- [12] "EEG," ADInstruments, 2020. [Online]. Available: <https://www.adinstruments.com/signal/eeg>. [Accessed June 2020].
- [13] K. Ishii, H. Iwata and T. Oshika, "Quantitative Evaluation of Changes in Eyeball Shape in Emmetropization and Myopic Changes Based on Elliptic Fourier Descriptors," *Investigative Ophthalmology and Visual Science*, vol. 58, no. 12, pp. 8585-8591, 2011.
- [14] V. Semiconductors, "Application of Optical Reflex Sensors," Vishay, 2002.

-
- [15] R. Electronics, "Eye Safety for Proximity Sensing Using Infrared Light-emitting Diodes," Renesas, 2016.
- [16] A. Devices, "Micropower, Zero Drift, True Rail-to-Rail Instrumentation Amplifier," Analog Devices, 2012.
- [17] W. T. Kaune, "Introduction to Power-Frequency Electric and Magnetic Fields," *Environmental Health Perspectives Supplements*, vol. 101, no. 4, pp. 73-81, 1993.
- [18] T. Zawistowski and P. Shah, "An Introduction to Sampling Theory," University of Houston, 2010. [Online]. Available: <http://www2.egr.uh.edu/~glover/applets/Sampling/Sampling.html>. [Accessed June 2020].
- [19] E. Por, M. v. Kooten and V. Sarkovic, "Nyquist-Shannon Sampling Theorem," Leiden University, Leiden, 2019.
- [20] R. Wang, "The Twin-T notch (Band-Stop) Filter," Harvey Mudd College, Claremont, 2019.
- [21] O. E. Design, "Twin T Notch Filter Design Tool," 2020. [Online]. Available: <http://sim.okawa-denshi.jp/images/CRTwinTD.gif>. [Accessed June 2020].
- [22] K. Takahashi and Y. Atsumi, "Precise Measurement of Individual Rapid Eye Movements in REM Sleep of Humans," *American Sleep Disorders Association and Sleep Research Society*, vol. 20, no. 9, pp. 743-752, 1997.
- [23] F. Semiconductor, "QRE1113, QRE1113GR Miniature Reflective Object Sensor," Fairchild Semiconductor, 2009.
- [24] Atmel, "Atmel 8-bit AVR Microcontroller with 2/4/8K Bytes In-System Programmable Flash," Atmel, 2013.
- [25] ST, "Medium-density performance line ARM-based 32-bit MCU with 64 or 128 KB Flash, USB, CAN, 7 timers, 2 ADCs, 9 com. interfaces," ST Microelectronics, 2015.
- [26] A. Osborne, *An Introduction to Microcomputers Volume 1: Basic Concepts*, Berkeley: Osborne-McGraw Hill, 1980.
- [27] Kingbright, "1.6X0.8mm SMD Chip LED Lamp," Kingbright Electronic, 2015.
- [28] O. Semiconductor, "Voltage Regulator - CMOS, Low Iq, SC70-5," ON Semiconductor, 2019.
- [29] ST, "TS4061 Precision Micropower Shunt Voltage Reference," ST Microelectronics, 2018.
- [30] MathWorks, "Help Center," MathWorks, 2020. [Online]. Available: <https://nl.mathworks.com/help/matlab>. [Accessed June 2020].
- [31] N. Instruments, "NI myDAQ Device Specifications," National Instruments, 2019.

Appendix

Figures



Figure 37 – Electrode showing aluminum point of contact.

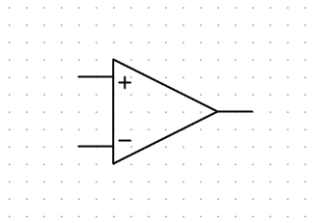


Figure 38 – Schematic symbol for an operational amplifier.

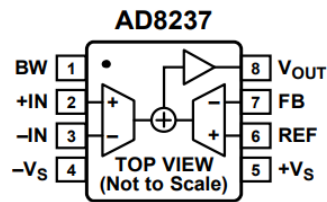


Figure 39 – Pinout diagram for AD8237 [16].

Table 7. Suggested Resistors for Various Gains (1% Resistors)

R1 (k Ω)	R2 (k Ω)	Gain
None	Short	1.00
49.9	49.9	2.00
20	80.6	5.03
10	90.9	10.09
5	95.3	20.06
2	97.6	49.8
1	100	101
1	200	201
1	499	500
1	1000	1001

Figure 40 – Table showing the recommended values for the gain setting resistors [16].

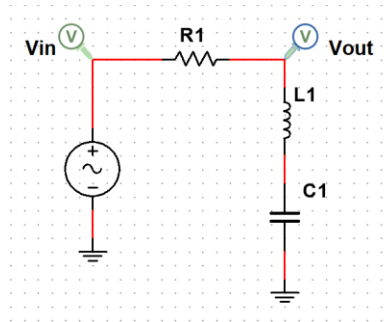


Figure 41 – Band-stop filter circuit.

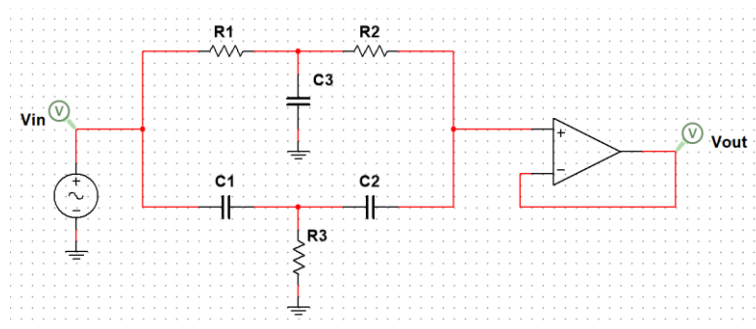


Figure 42 – Notch filter circuit with output buffer.

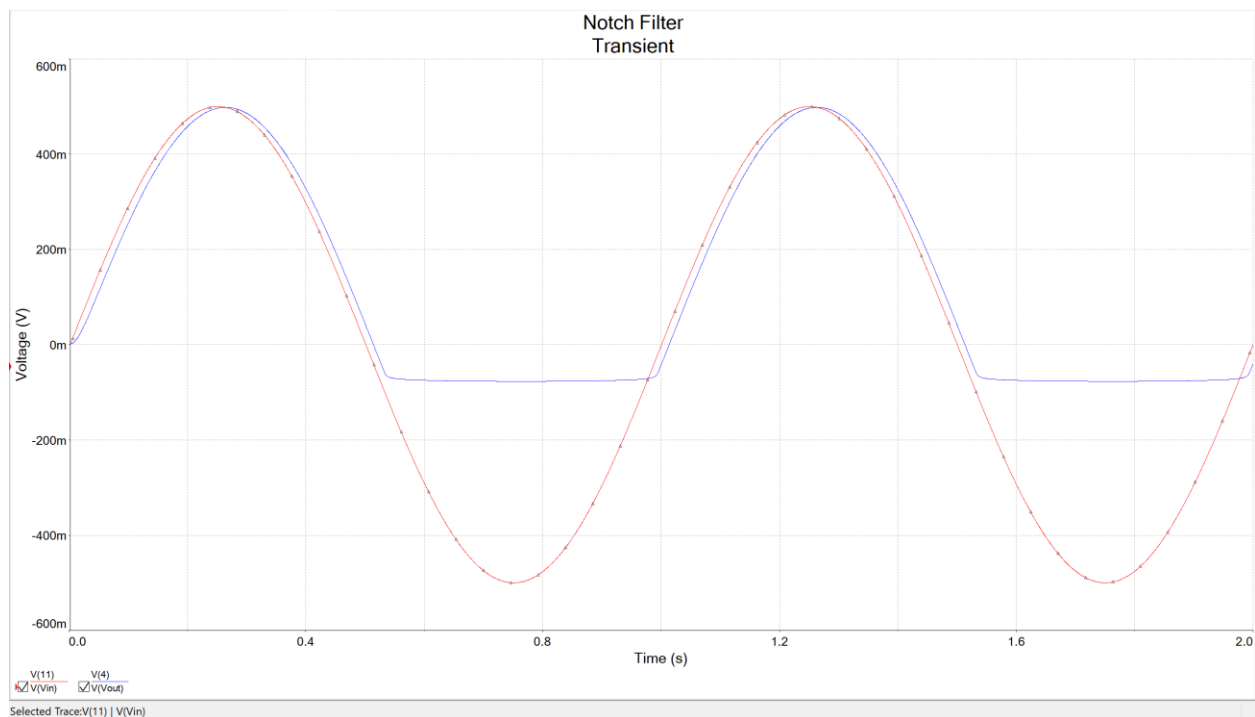


Figure 43 – Unbiased filter behavior. The red line shows the input of the filter while the blue line shows the output.

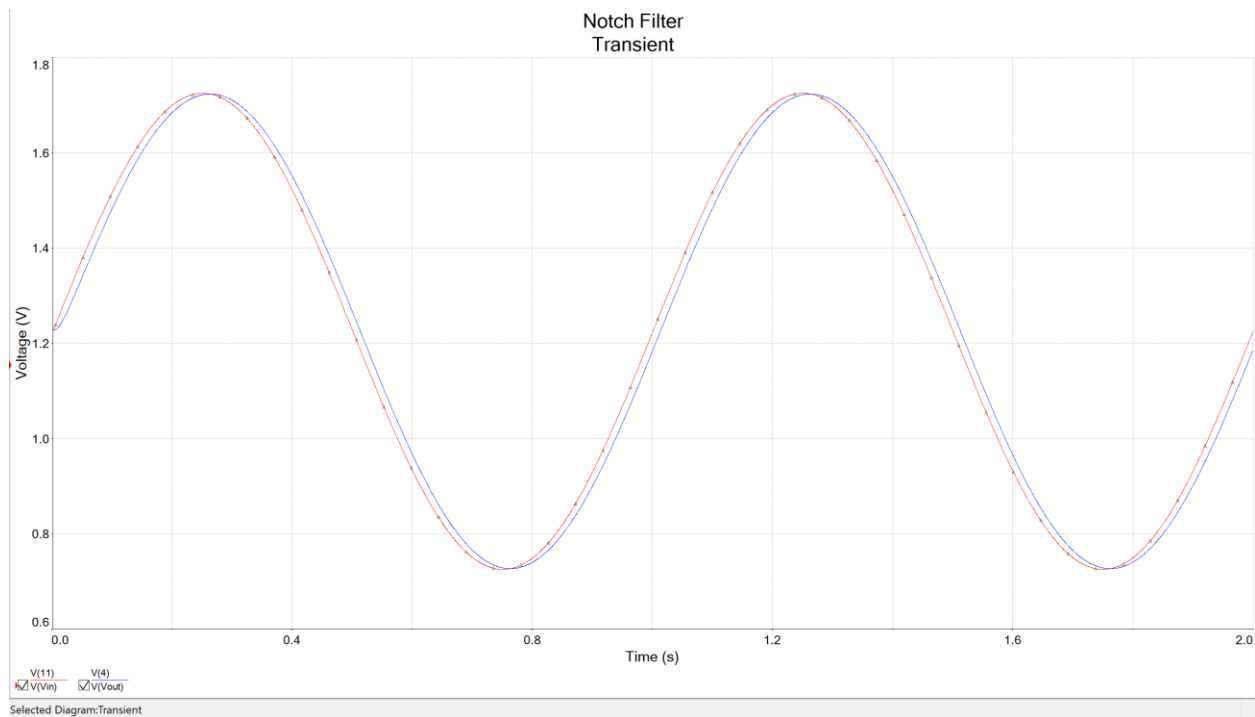


Figure 44 – Biased filter response. The red line shows the input of the filter, while the blue line shows the output.

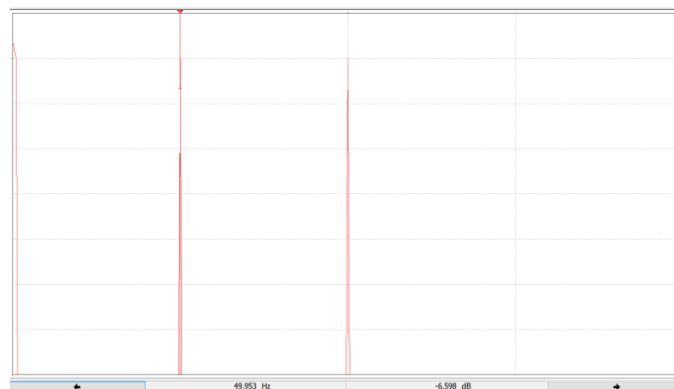


Figure 45 – Output screen of the Spectrum Analyzer instrument measuring the input signal, showing three distinct peaks (1Hz, 50Hz, and 100Hz).

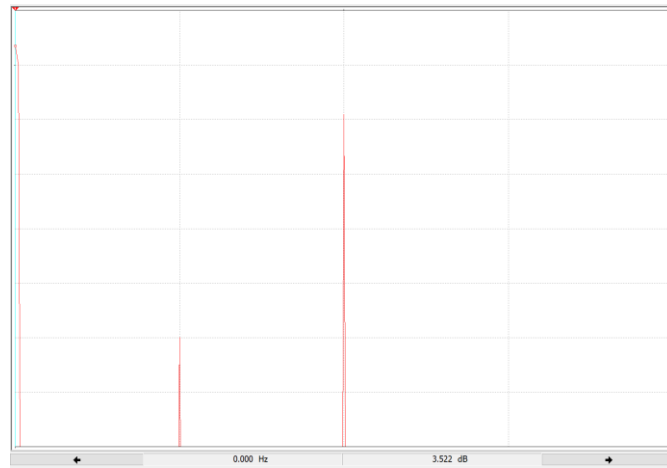


Figure 46 – Output screen of the Spectrum Analyzer instrument measuring the output signal of the filter, showing a clear suppression of the peak at 50Hz.

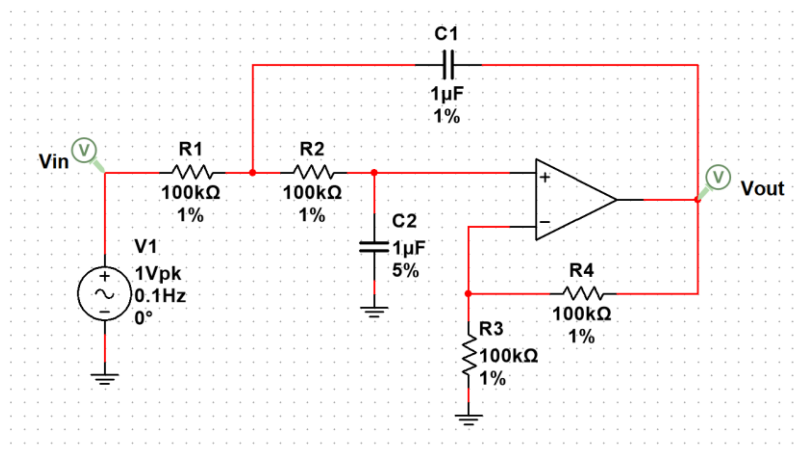


Figure 47 – Active low-pass filter built in Multisim showing component tolerances.

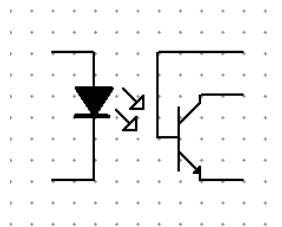


Figure 48 – Schematic symbol of an optocoupler.

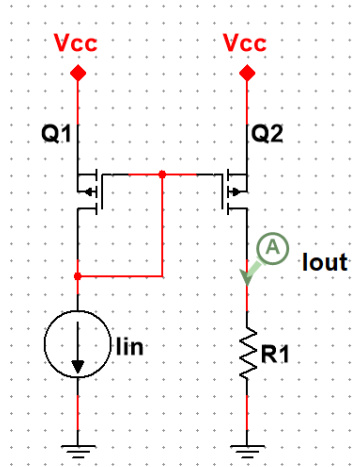


Figure 49 – Current mirror circuit using P-Channel MOSFETS.

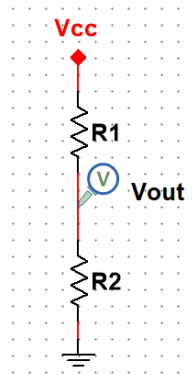


Figure 50 – Voltage divider circuit schematic.

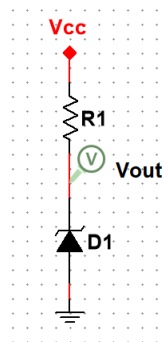


Figure 51 – Voltage reference circuit using a Zener diode.

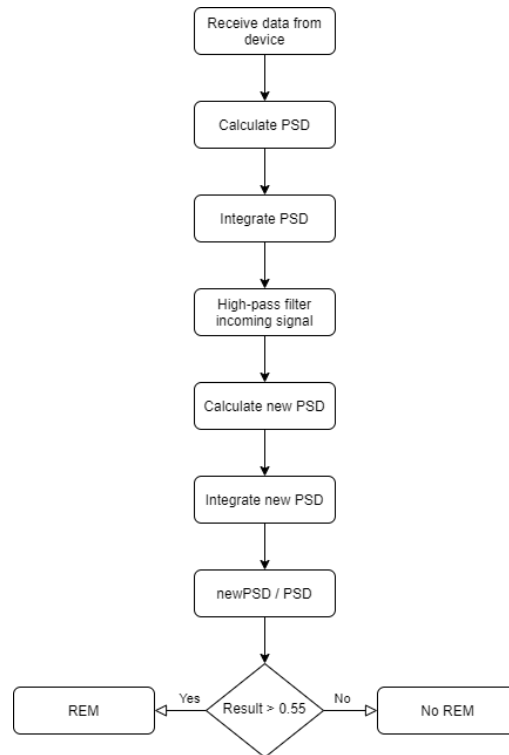


Figure 52 – REM detection algorithm flowchart.



Figure 53 – Oscilloscope's screen showing the behavior of the voltage regulator. Channel one (yellow) shows the input voltage of 5 V. Channel two (blue) shows the output of the LDO is 3.3 V.



Figure 54 – Oscilloscope's screen showing the behavior of the virtual ground circuit. Channel one (yellow) shows the supply line (V_{cc}). Channel two (blue) shows the output of the virtual ground circuit.



Figure 55 – Oscilloscope's screen showing the output of the optocoupler readout circuit with a piece of paper 3 mm away.

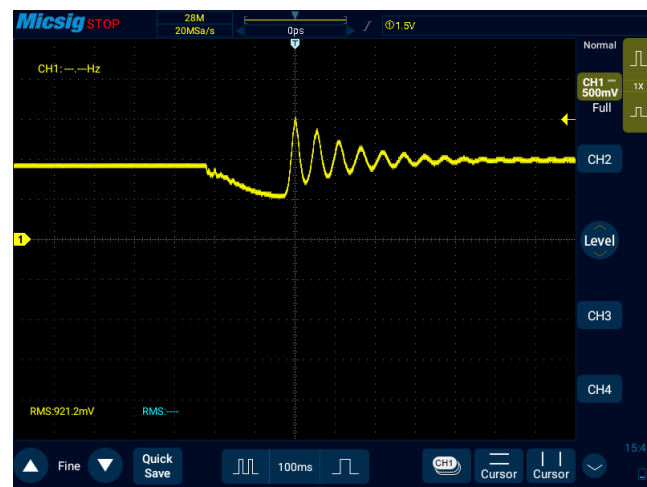


Figure 56 – Oscilloscope's screen showing the output of the optocoupler readout circuit when the piece of paper is moving back and forth.

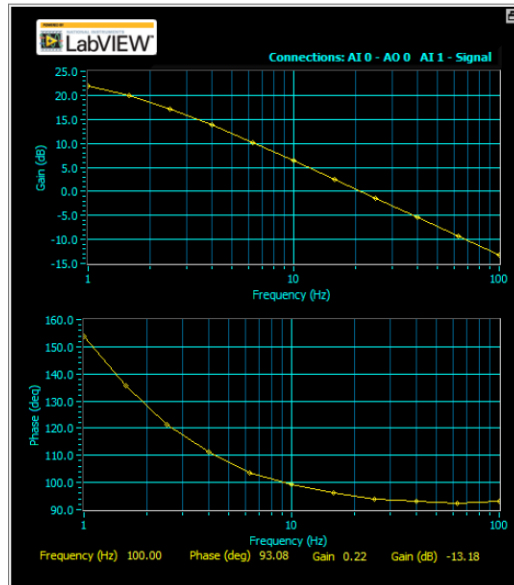


Figure 57 – Bode Analyzer’s attempt at evaluating the low frequency bandpass filter.

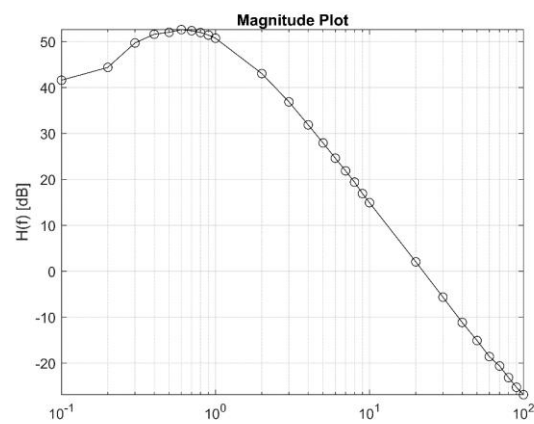


Figure 58 – Magnitude plot of the band-pass filter using an alternative measurement method.

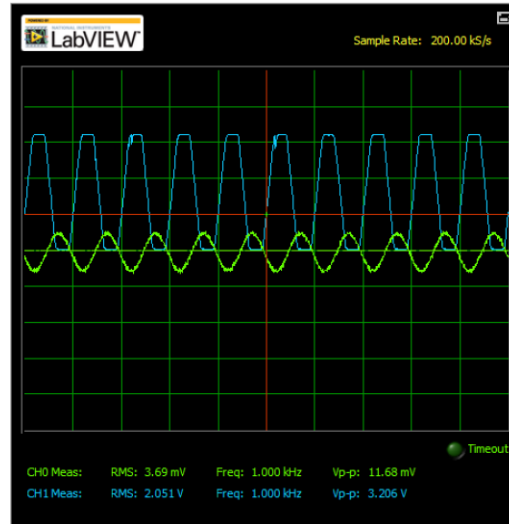


Figure 59 – MyDAQ's oscilloscope application screen showing the output of the InAmp circuit. The amplitude of the input voltage (green) applied is 5 mV. The output voltage (blue) is shown to have an amplitude greater than 2 V.

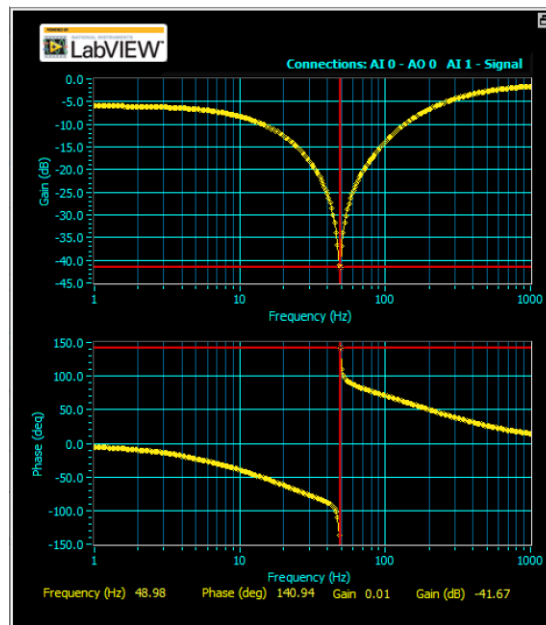


Figure 60 – Notch filter's response gathered via the MyDAQ's Bode Analyzer application.

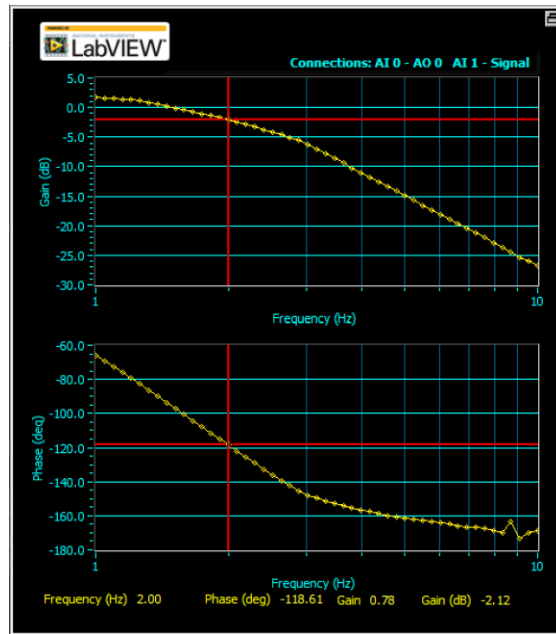


Figure 61 – Low-pass filter's response as measured by the MyDAQ's Bode Analyzer application.

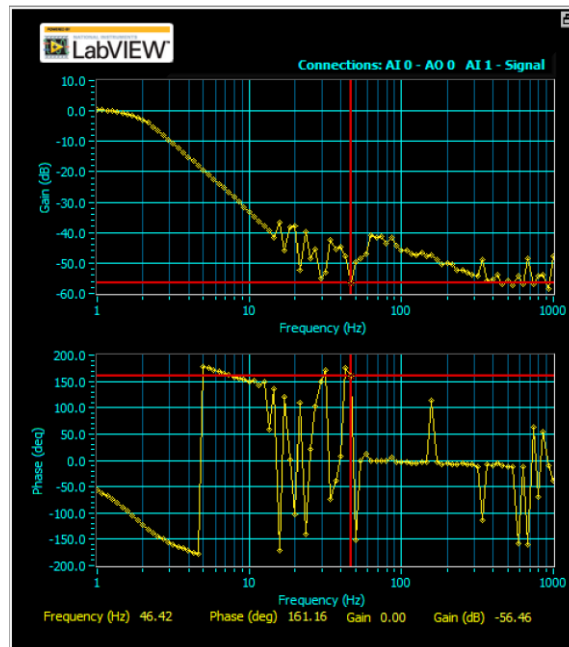


Figure 62 – Total response of the cascaded filters as measured by the MyDAQ.

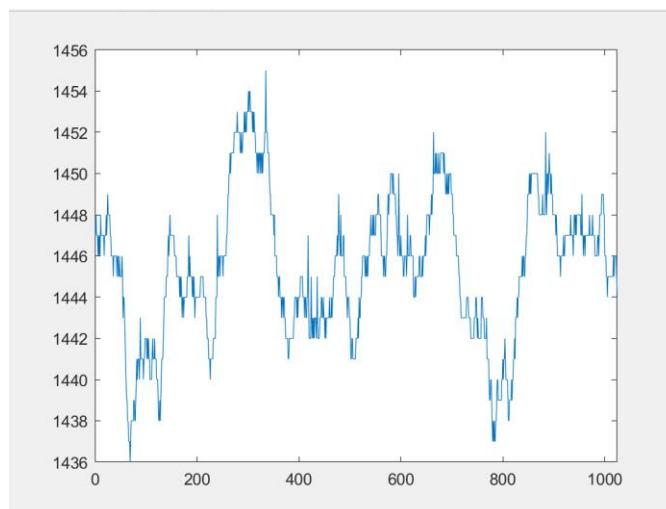


Figure 63 – Signal received on the COM port sent by the sleep monitoring device.

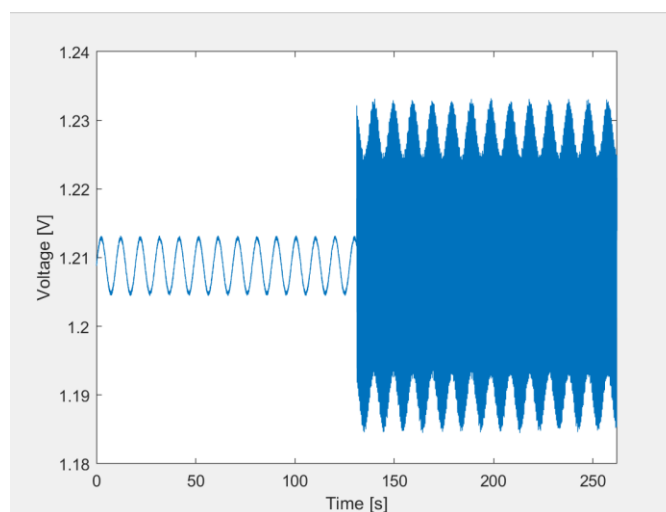


Figure 64 – Algorithm test signal consisting of a low frequency interferer with noise and REM frequency after 130 seconds.

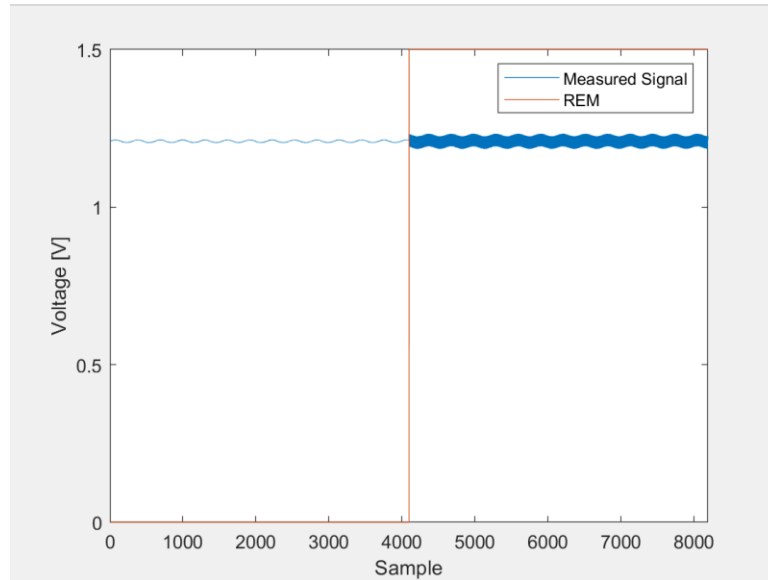
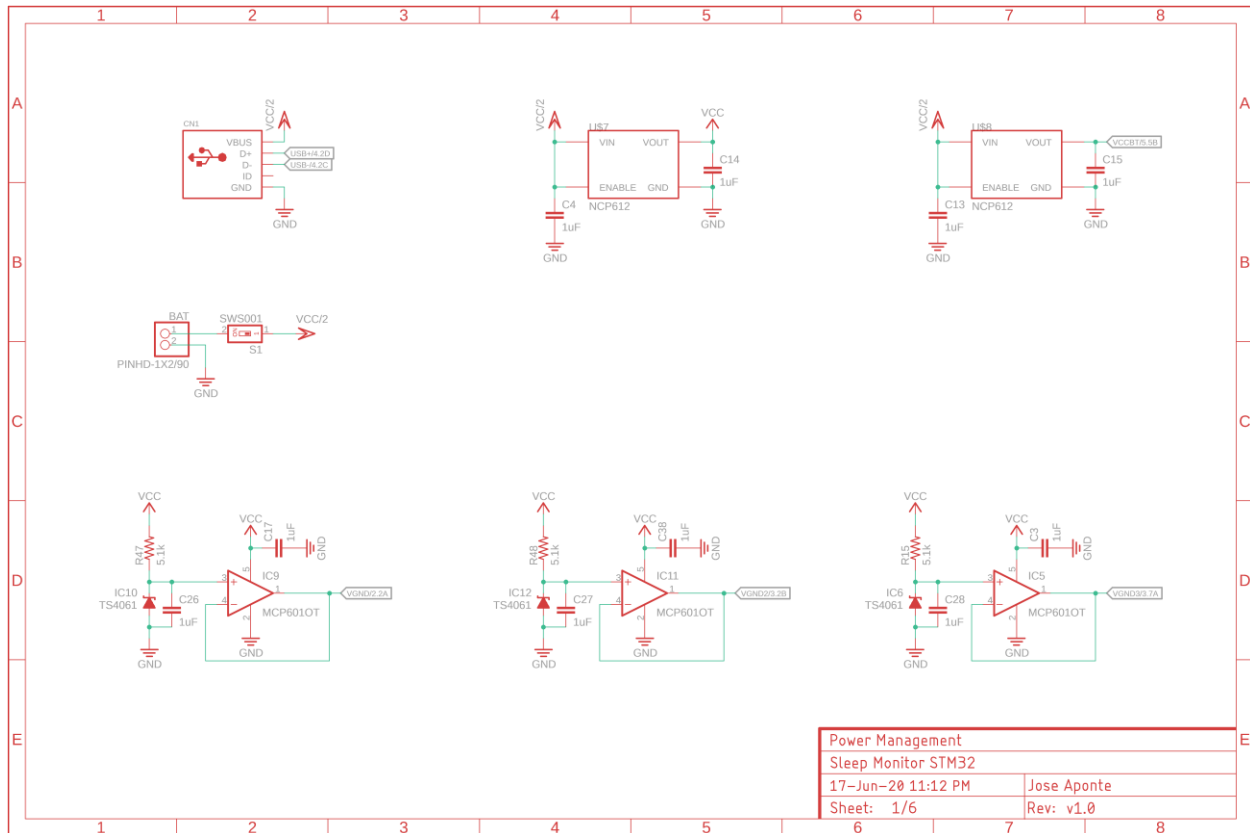
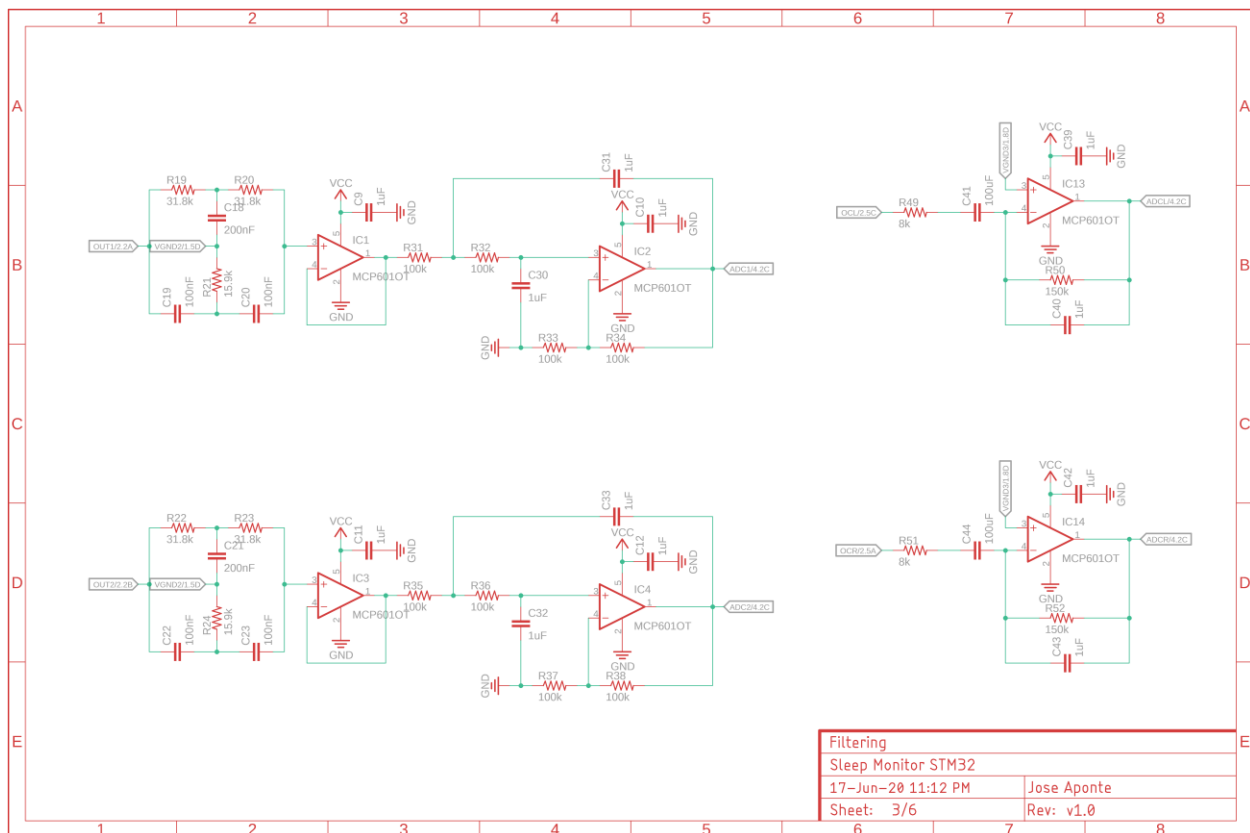
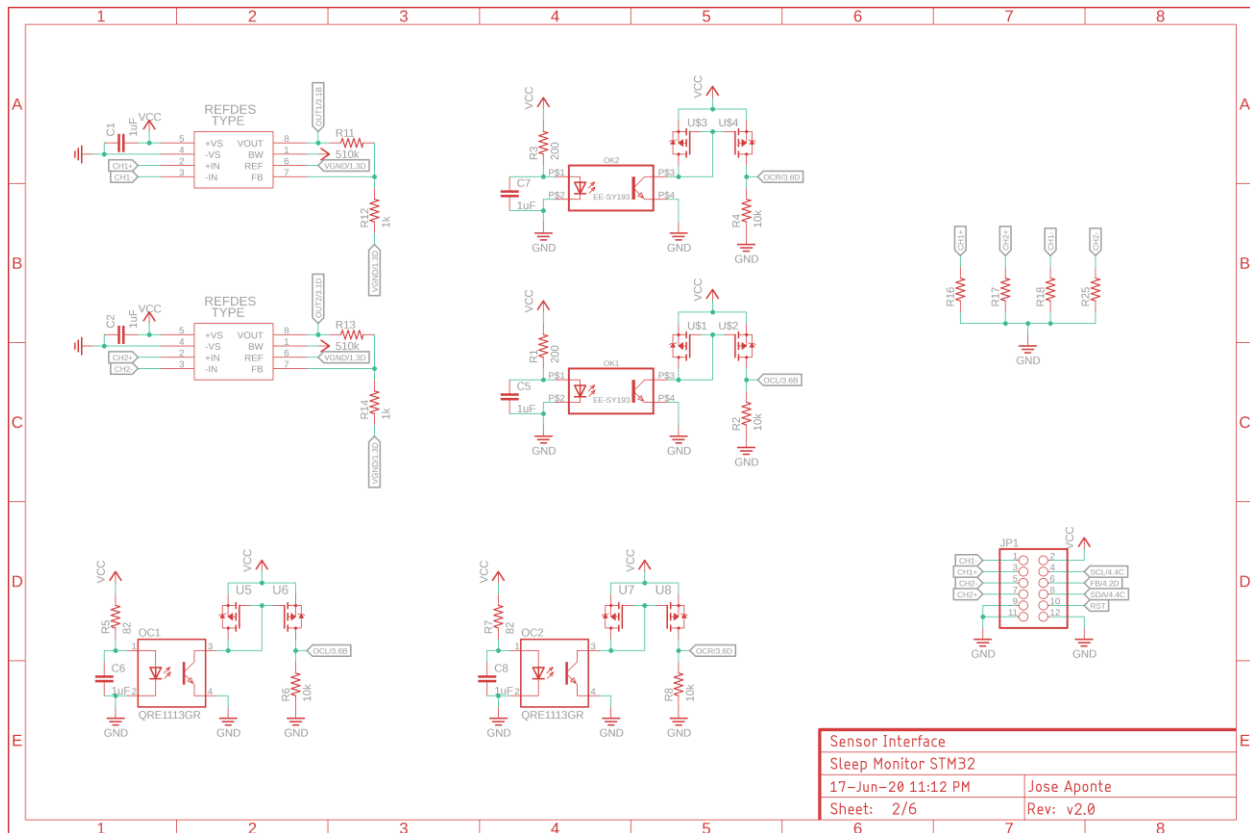
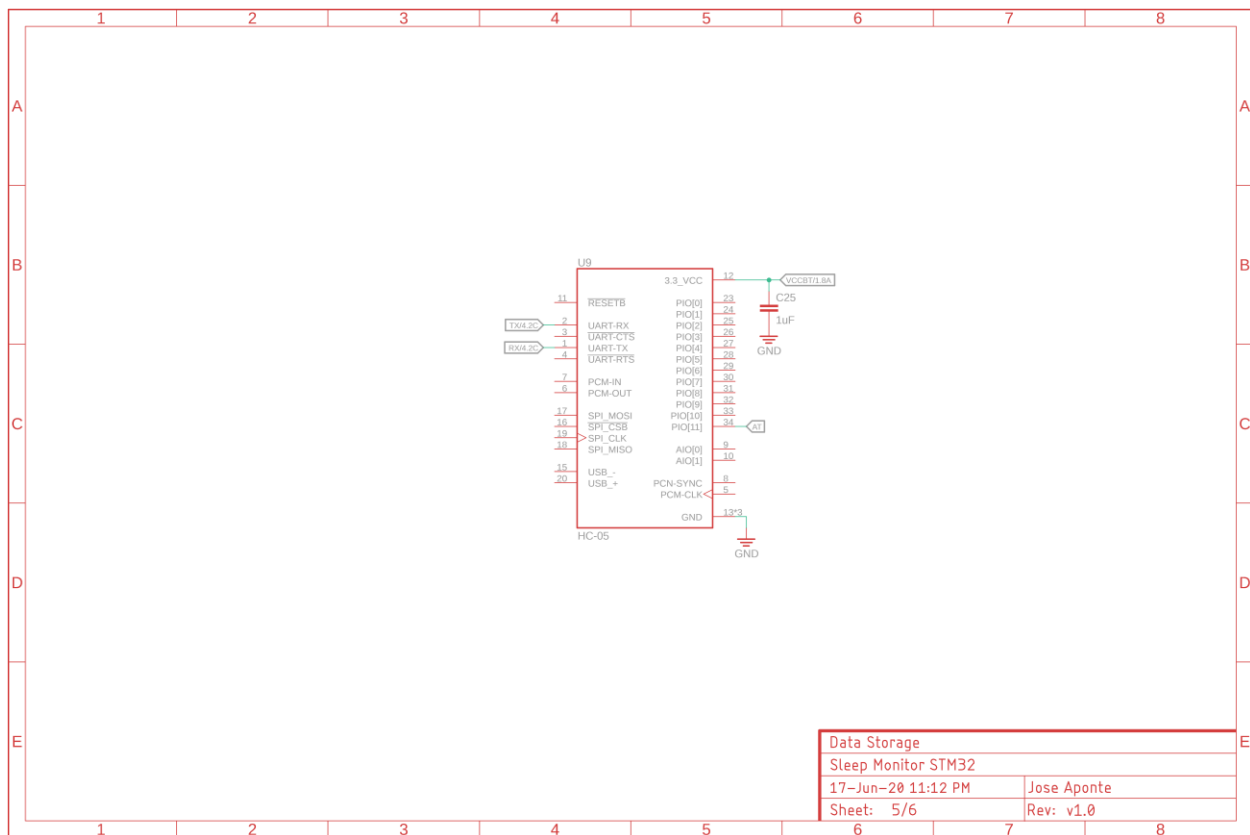
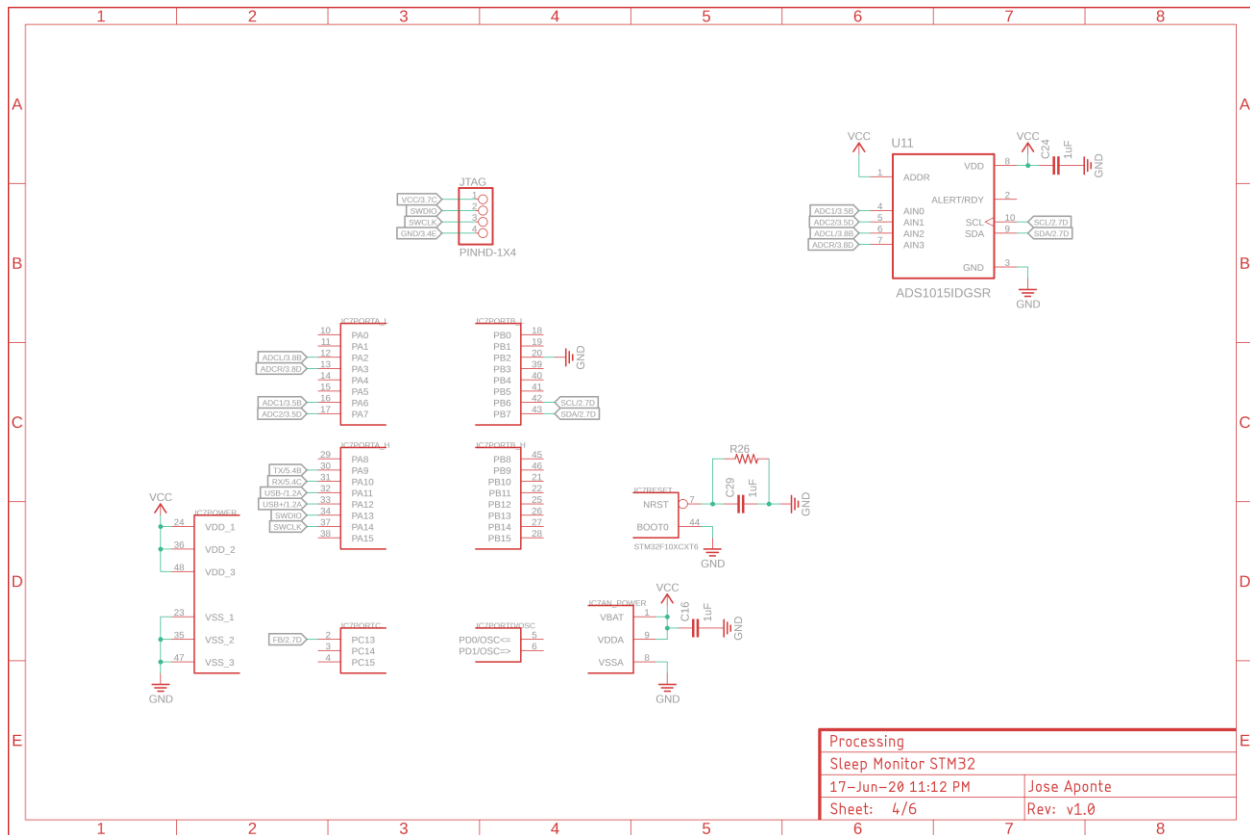


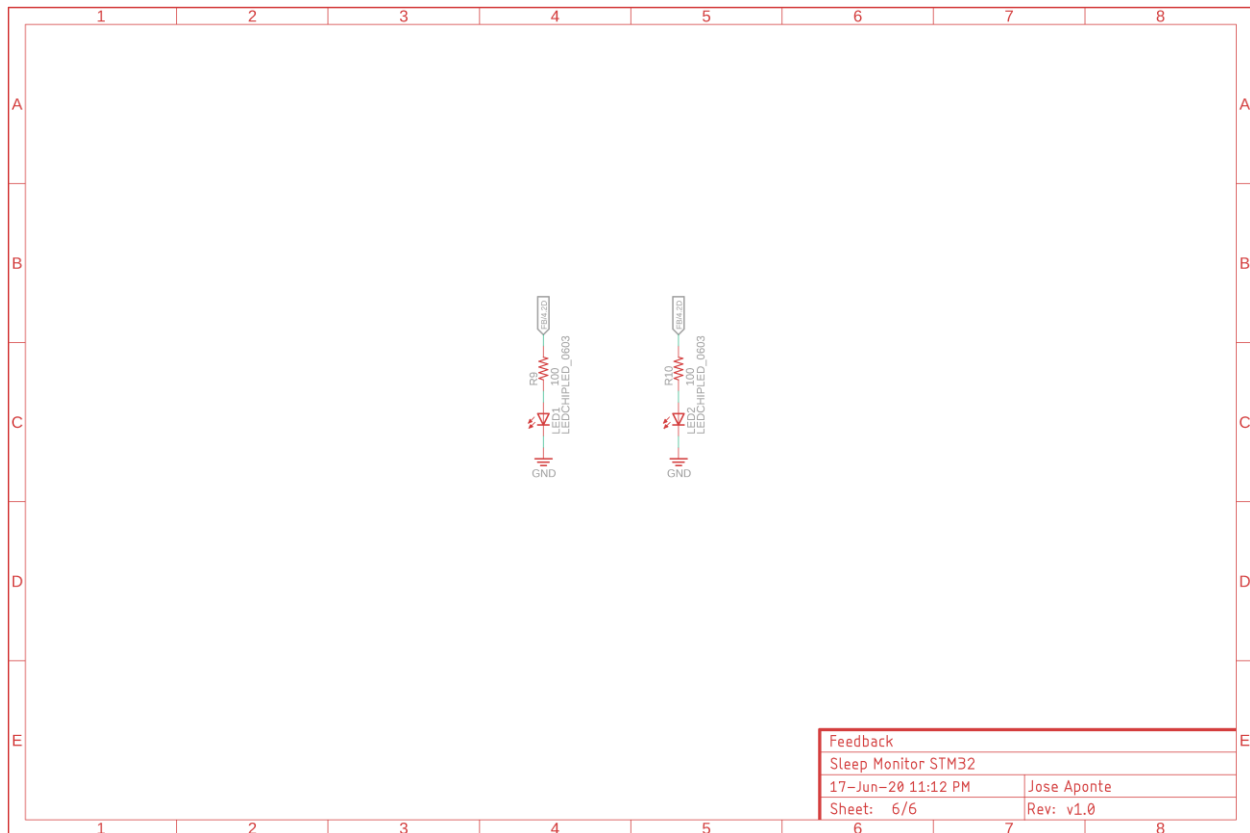
Figure 65 – Results output by the REM detection algorithm to the applied test signal.

Device Schematics









MATLAB Code

COM Port Data Acquisition

```
clear variables; close all;

s = serialport("COM12", 9600);

delay = 25;           %Time delay specified in the device's processor in ms.
fs = 31.25;           %Sampling frequency in Hz.
samples = 1024;       %Number of samples being sent at one time.
N = 8*samples;        %Size for the FFT.
f = [0:N-1];          %Frequency Axis Startup (samples).
f = f*fs/N;           %Frequency axis translation into Hz.
totalSignal = 1450;

while true
    %write(s, 49, "uint8");

    %Repeat for the number of samples being sent. Each sample value is one
    %line.
    for i = 1:samples
        signal(i) = str2double(readline(s)); %Read COM port data.
        if signal(i) == NaN
            signal(i) = 1450; %In case of an error, replace with mid value.
        end
    end

    totalSignal = [totalSignal signal]; %Store each received interval into a bigger array.

    %S = abs(fft(signal-mean(signal), N));
    SX = pwelch(detrend(signal));

    sf = 0:length(SX)-1; %Create frequency axis for PSD.
    sf = sf*fs/(2*length(SX)); %Translate frequency axis to Hz.
```

```

figure(1);
plot(signal);
xlabel("Sample");
ylabel("Digital Voltage");

% figure(2);
% plot(f, S);
% xlabel("Frequency [Hz]");
% ylabel("FFT Magnitude");
% xlim([0 f(end)/2]);

figure(3);
plot(sf, SX);
xlabel("Frequency [Hz]");
ylabel("Amplitude");
xlim([0 sf(end)]);
end

```

Algorithm Code

```

close all; clear variables;

%totalSignal = importdata("Test Signal.mat");
%totalSignal = importdata("REM Test with device 1.mat");
%totalSignal = importdata("REM Test with device 2 (noREM).mat");
%totalSignal = importdata("REM Test with device 2 (withREM).mat");
%totalSignal = importdata("REM Test with device 4 (1HzREM).mat");
%totalSignal = importdata("REM Test 5.mat"); %REM @ 136s, 265s, and 420s
%totalSignal = importdata("Attempt Sleep 1 no REM.mat");
totalSignal = importdata("Attempt Sleep 2 with 2dreams.mat");

totalSignal = totalSignal*3.3/4096;
samples = 1024;
fs = 31.25;
N = 4*samples;
f = [0:N-1].*fs/N;
t = [0:length(totalSignal)-1]./fs;

figure(1);
plot(t, totalSignal);
xlabel("Time [s]");
ylabel("Voltage [V]");
xlim([0 t(end)]);

for i = 0:round(length(totalSignal)/samples)-1
    chunk = detrend(totalSignal(i*samples+1:(i+1)*samples));

    totalPSD = pwelch(chunk, hanning(samples), [], N);
    totalPower(i+1) = trapz(totalPSD);

    chunk = highpass(chunk, 0.3, fs);

    currentPSD = pwelch(chunk, hanning(samples), [], N);
    currentPower(i+1) = trapz(currentPSD);

    sf = [0:length(currentPSD)-1].*fs/(2*length(currentPSD));

    percentage = currentPower./totalPower;

% figure(4+i);
% plot(chunk);
% hold on;

figure(2);
plot(sf, totalPSD);
xlabel("Frequency [Hz]");
ylabel("Power/Frequency [W/Hz]");
hold on;

figure(3);

```

```

    plot(sf, currentPSD);
    xlabel("Frequency [Hz]");
    ylabel("Power/Frequency [W/Hz]");
    hold on;
    %xlim([0 f(end)/2]);
end

for i = 1:length(percentage)
    if percentage(i) >= 0.55% || currentPower(i) >= 0.9%0.9 for test 4
        isREM(i) = 1;
    else
        isREM(i) = 0;
    end
end

plotIsREM = 0;

for j = 1:length(isREM)
    for i = 1:samples
        temp(i) = 1*isREM(j);
    end
    plotIsREM = [plotIsREM temp];
end

figure(4);
plot(totalSignal);
hold on;
plot(plotIsREM);
xlabel("Sample");
ylabel("Voltage [V]");
xlim([0 length(totalSignal)]);
legend("Measured Signal", "REM");

```

STM32 main.c Code

```

/* USER CODE BEGIN Header */
/**
 * ****
 * @file           : main.c
 * @brief          : Main program body
 * ****
 * @attention
 *
 * <h2><center>&copy; Copyright (c) 2020 STMicroelectronics.
 * All rights reserved.</center></h2>
 *
 * This software component is licensed by ST under BSD 3-Clause license,
 * the "License"; You may not use this file except in compliance with the
 * License. You may obtain a copy of the License at:
 *
 *             opensource.org/licenses/BSD-3-Clause
 * ****
 */
/* USER CODE END Header */

/* Includes -----*/
#include "main.h"
#include <string.h>

/* Private includes -----*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

```

```

/* Private define -----*/
/* USER CODE BEGIN PD */
/* USER CODE END PD */

/* Private macro -----*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables -----*/
ADC_HandleTypeDef hadc1;

TIM_HandleTypeDef htim1;
TIM_HandleTypeDef htim2;

UART_HandleTypeDef huart1;

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes -----*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM1_Init(void);
static void MX_TIM2_Init(void);
static void MX_USART1_UART_Init(void);
static void MX_ADC1_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code -----*/
/* USER CODE BEGIN 0 */
char message[] = "Sampled signal printed.";
double analogRead();

uint16_t samples = 512;           //Number of samples to take.
double fs = 2000.2;               //Sampling frequency in Hz.

uint32_t Ts = 0;
uint32_t pTs = 0;
/* USER CODE END 0 */

/**
 * @brief The application entry point.
 * @retval int
 */
int main(void)
{
    /* USER CODE BEGIN 1 */
    int x[samples];
    /* USER CODE END 1 */

    /* MCU Configuration-----*/

    /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
    HAL_Init();

    /* USER CODE BEGIN Init */

    /* USER CODE END Init */

    /* Configure the system clock */
    SystemClock_Config();

    /* USER CODE BEGIN SysInit */

    /* USER CODE END SysInit */

```

```

/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_TIM1_Init();
//MX_TIM2_Init();
MX_USART1_UART_Init();
MX_ADC1_Init();
/* USER CODE BEGIN 2 */
HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_13); //Blink LED Init
HAL_Delay(1000);
HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_13);
/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    /* USER CODE END WHILE */
    for (int j = 0; j < samples; j++){
        Ts = HAL_GetTick();

        x[j] = analogRead();

        //printNumber(x[j]);

        HAL_Delay(25);

        fs = 1000/(double)(Ts - pTs);

        pTs = Ts;
    }

    for (int j = 0; j < samples; j++){
        printNumber(x[j]);
    }
    /*println(&huart1, message);

    HAL_Delay(5.00);
    */
    /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
}

/* USER FUNCTIONS */
void println(UART_HandleTypeDef *huart, char* _out[]){
    HAL_UART_Transmit(huart, (uint8_t *) out, strlen( out), 1000);
    char newline[2] = "\r\n";
    HAL_UART_Transmit(huart, (uint8_t *) newline, 2, 1000);
}

double analogRead(void){
    uint32_t adcResult = 0;

    HAL_ADC_Start(&hadc1);
    HAL_ADC_PollForConversion(&hadc1, 10);
    adcResult = HAL_ADC_GetValue(&hadc1);
    HAL_ADC_Stop(&hadc1);
    return adcResult;
}

void printNumber(int number){
    char value[10]; //a temp array to hold results of conversion

    sprintf(value, "%d", number);

    HAL_UART_Transmit(&huart1, (uint8_t *) value, strlen(value), 1000);
    char newline[2] = "\r\n";
    HAL_UART_Transmit(&huart1, newline, 2, 1000);
}

```

```

}

/* END USER FUNCTIONS */
/**
 * @brief System Clock Configuration
 * @retval None
 */
void SystemClock_Config(void)
{
    RCC_OscInitTypeDef RCC_OscInitStruct = {0};
    RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
    RCC_PeriphCLKInitTypeDef PeriphClkInit = {0};

    /** Initializes the CPU, AHB and APB busses clocks
    */
    RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
    RCC_OscInitStruct.HSIState = RCC_HSI_ON;
    RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
    RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
    RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI_DIV2;
    RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL16;
    if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
    {
        Error_Handler();
    }
    /** Initializes the CPU, AHB and APB busses clocks
    */
    RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLOCK
                                |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
    RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
    RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
    RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
    RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

    if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
    {
        Error_Handler();
    }
    PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_ADC;
    PeriphClkInit.AdcClockSelection = RCC_ADCPCLK2_DIV8;
    if (HAL_RCCEx_PeriphCLKConfig(&PeriphClkInit) != HAL_OK)
    {
        Error_Handler();
    }
}

/**
 * @brief ADC1 Initialization Function
 * @param None
 * @retval None
 */
static void MX_ADC1_Init(void)
{
    /* USER CODE BEGIN ADC1_Init 0 */

    /* USER CODE END ADC1_Init 0 */

    ADC_ChannelConfTypeDef sConfig = {0};

    /* USER CODE BEGIN ADC1_Init 1 */

    /* USER CODE END ADC1_Init 1 */
    /** Common config
    */
    hadc1.Instance = ADC1;
    hadc1.Init.ScanConvMode = ADC_SCAN_DISABLE;
    hadc1.Init.ContinuousConvMode = DISABLE;
    hadc1.Init.DiscontinuousConvMode = DISABLE;

```

```

hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;
hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
hadc1.Init.NbrOfConversion = 1;
if (HAL_ADC_Init(&hadc1) != HAL_OK)
{
    Error_Handler();
}
/** Configure Regular Channel
 */
sConfig.Channel = ADC_CHANNEL_2;
sConfig.Rank = ADC_REGULAR_RANK_1;
sConfig.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
{
    Error_Handler();
}
/* USER CODE BEGIN ADC1_Init 2 */

/* USER CODE END ADC1_Init 2 */

}

/**
 * @brief TIM1 Initialization Function
 * @param None
 * @retval None
 */
static void MX_TIM1_Init(void)
{
    /* USER CODE BEGIN TIM1_Init 0 */

    /* USER CODE END TIM1_Init 0 */

    TIM_ClockConfigTypeDef sClockSourceConfig = {0};
    TIM_MasterConfigTypeDef sMasterConfig = {0};

    /* USER CODE BEGIN TIM1_Init 1 */

    /* USER CODE END TIM1_Init 1 */
    htim1.Instance = TIM1;
    htim1.Init.Prescaler = 0;
    htim1.Init.CounterMode = TIM_COUNTERMODE_UP;
    htim1.Init.Period = 0;
    htim1.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
    htim1.Init.RepetitionCounter = 0;
    htim1.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
    if (HAL_TIM_Base_Init(&htim1) != HAL_OK)
    {
        Error_Handler();
    }
    sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
    if (HAL_TIM_ConfigClockSource(&htim1, &sClockSourceConfig) != HAL_OK)
    {
        Error_Handler();
    }
    sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
    sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
    if (HAL_TIMEx_MasterConfigSynchronization(&htim1, &sMasterConfig) != HAL_OK)
    {
        Error_Handler();
    }
    /* USER CODE BEGIN TIM1_Init 2 */

    /* USER CODE END TIM1_Init 2 */

}

/**

```

```

* @brief TIM2 Initialization Function
* @param None
* @retval None
*/
static void MX_TIM2_Init(void)
{
    /* USER CODE BEGIN TIM2_Init 0 */

    /* USER CODE END TIM2_Init 0 */

    TIM_ClockConfigTypeDef sClockSourceConfig = {0};
    TIM_MasterConfigTypeDef sMasterConfig = {0};

    /* USER CODE BEGIN TIM2_Init 1 */

    /* USER CODE END TIM2_Init 1 */
    htim2.Instance = TIM2;
    htim2.Init.Prescaler = 0;
    htim2.Init.CounterMode = TIM_COUNTERMODE_UP;
    htim2.Init.Period = 0;
    htim2.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
    htim2.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
    if (HAL_TIM_Base_Init(&htim2) != HAL_OK)
    {
        Error_Handler();
    }
    sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
    if (HAL_TIM_ConfigClockSource(&htim2, &sClockSourceConfig) != HAL_OK)
    {
        Error_Handler();
    }
    sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
    sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
    if (HAL_TIMEx_MasterConfigSynchronization(&htim2, &sMasterConfig) != HAL_OK)
    {
        Error_Handler();
    }
    /* USER CODE BEGIN TIM2_Init 2 */

    /* USER CODE END TIM2_Init 2 */

}

/**
* @brief USART1 Initialization Function
* @param None
* @retval None
*/
static void MX_USART1_UART_Init(void)
{
    /* USER CODE BEGIN USART1_Init 0 */

    /* USER CODE END USART1_Init 0 */

    /* USER CODE BEGIN USART1_Init 1 */

    /* USER CODE END USART1_Init 1 */
    huart1.Instance = USART1;
    huart1.Init.BaudRate = 9600;
    huart1.Init.WordLength = UART_WORDLENGTH_8B;
    huart1.Init.StopBits = UART_STOPBITS_1;
    huart1.Init.Parity = UART_PARITY_NONE;
    huart1.Init.Mode = UART_MODE_TX_RX;
    huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
    huart1.Init.OverSampling = UART_OVERSAMPLING_16;
    if (HAL_UART_Init(&huart1) != HAL_OK)
    {

```



```

    Error_Handler();
}
/* USER CODE BEGIN USART1_Init 2 */

/* USER CODE END USART1_Init 2 */

}

/**
 * @brief GPIO Initialization Function
 * @param None
 * @retval None
 */
static void MX_GPIO_Init(void)
{
    GPIO_InitTypeDef GPIO_InitStruct = {0};

    /* GPIO Ports Clock Enable */
    __HAL_RCC_GPIOC_CLK_ENABLE();
    __HAL_RCC_GPIOA_CLK_ENABLE();

    /*Configure GPIO pin Output Level */
    HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_RESET);

    /*Configure GPIO pin : PC13 */
    GPIO_InitStruct.Pin = GPIO_PIN_13;
    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
    HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

    /*Configure GPIO pins : PA2 PA3 PA6 PA7 */
    GPIO_InitStruct.Pin = GPIO_PIN_2|GPIO_PIN_3|GPIO_PIN_6|GPIO_PIN_7;
    GPIO_InitStruct.Mode = GPIO_MODE_ANALOG;
    HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

/* USER CODE END 4 */

/**
 * @brief This function is executed in case of error occurrence.
 * @retval None
 */
void Error_Handler(void)
{
    /* USER CODE BEGIN Error_Handler_Debug */
    /* User can add his own implementation to report the HAL error return state */

    /* USER CODE END Error_Handler_Debug */
}

#ifdef USE_FULL_ASSERT
/**
 * @brief Reports the name of the source file and the source line number
 * where the assert_param error has occurred.
 * @param file: pointer to the source file name
 * @param line: assert_param error line source number
 * @retval None
 */
void assert_failed(uint8_t *file, uint32_t line)
{
    /* USER CODE BEGIN 6 */
    /* User can add his own implementation to report the file name and line number,
    tex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
    /* USER CODE END 6 */
}

```

```
#endif /* USE_FULL_ASSERT */
```

```
/****** (C) COPYRIGHT STMicroelectronics *****END OF FILE*****/
```