Slow Wave Sleep Enhancement System Final Paper

By

Aidan Stahl

Kavin Bharathi

Vikram Chakravarthi

Final Report for ECE 445, Senior Design, Spring 2026

TA: Hossein Ataee

6 May 2026

Project No. 36

Abstract

This report presents the design and implementation of a Slow-Wave Sleep (SWS) Detection and Enhancement system. The system addresses the widespread problem of degraded slow-wave sleep quality, which is associated with neurological conditions such as Alzheimer's disease, depression, and impaired memory consolidation. The device records EEG brain signals using a headband worn by the user, amplifies and digitizes those signals on our ADS1299-based custom PCB, transmits the digitized data wirelessly via Bluetooth to a host computer, and uses open-source sleep analysis libraries to detect slow-wave sleep in real time. Upon detection of a slow-wave's up-state, the system plays a carefully timed burst of pink noise to enhance slow-wave activity and extend its duration. The final system successfully demonstrated real-time EEG signal transmission, accurate slow-wave detection on publicly available EEG datasets, and pink noise playback within the required 300 ms latency. Battery life was estimated at 18–20 hours, well exceeding the 10-hour requirement. Lastly, the headset was considered comfortable enough to wear while sleeping.

Contents

1. Introduction

1.1 Problem and Solution

Many common neurological conditions like Alzheimer's disease, depression, and memory issues are associated with patients receiving lower quality sleep. Specifically, these issues often stem from a lack of a specific type of sleep known as slow wave sleep (SWS). As individuals age, sleep disorders and other sleep-related issues lead to a lack of overall sleep. As a result, the amount of time an individual spends in SWS and the quality of SWS they experience typically declines with age, contributing to many of the issues mentioned above. Our team is trying to improve sleep quality using a wearable device that is non-invasive and cost effective. This device will record EEG waves and then detect when the user is in Slow Wave Sleep (SWS) using the aid of specialized software. Once the user enters SWS, the system emits carefully timed bursts of pink noise through an auditory interface to enhance slow wave activity and extend its duration. The team that we're working with, Team 05 - Acoustic Stimulation to Improve Sleep, has cited studies that back up the effectiveness of pink noise bursts during SWS in improving overall memory.

1.2 Subsystem Overview

As shown in *Figure 1*, our project is divided into five main parts: the power subsystem, ADC subsystem, microcontroller subsystem, computer subsystem, and user subsystem. The power subsystem is responsible for powering our STM microcontroller and analog to digital converter (ADC). The ADC subsystem converts analog EEG signals into digitized values, which are then sent to the microcontroller subsystem as 24-bit values via SPI communication. The microcontroller subsystem sends the digitized values to the computer subsystem for processing. Data is sent via Bluetooth Low Energy (BLE) or UART communication. The computer subsystem converts the 24-bit values to microvolts and plots the data in the time domain. Additionally, the data is analyzed to determine whether the user is in slow-wave sleep. If slow-wave sleep is detected, the data is sent to a peak detection algorithm, where an adaptable threshold is used to determine where negative peaks of the slow-wave sleep waves occur, and pink noise is subsequently played along the upstate of these waves. The user subsystem describes the user of the device.

1.3 Outcome/Functionality

In our initial design review document, we outlined specific high-level requirements for our project. Pink noise should be played within 300 ms of detecting a slow-wave sleep wave, the battery should last at least 10 hours to support usage for a full night of sleep, and if random people are surveyed, the headset should receive a comfort rating of at least ⅘ , so it does not disrupt a user's sleep cycle. Additionally, we described more detailed requirements in the requirements and verifications section of the design review. In summary, we intended to measure a user's EEG brain waves while a user is sleeping, accurately measure and plot the data, classify slow-wave sleep correctly, detect the upstate of each slow wave and play pink noise within 300

ms of the start of the upstate. We were able to meet the vast majority of our functionality requirements. Our final design measured EEG signals properly and was able to detect slow-wave sleep and play pink noise within 300 ms of a detected slow wave upstate. Our block-level design remained roughly the same with the only change being replacing Bluetooth with UART communication due to bluetooth antenna design issues and implementation problems within the STM IDE.

1.4 Block Diagram

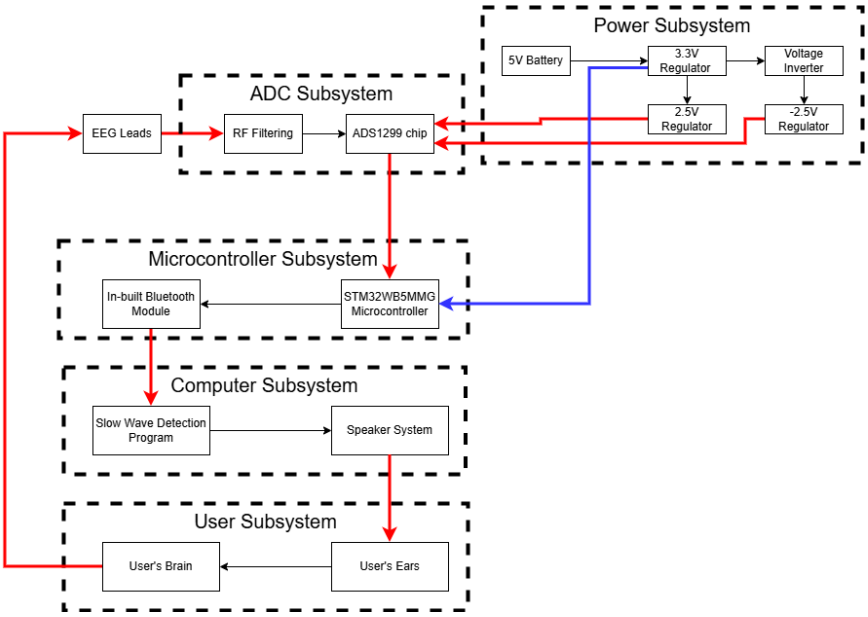

Figure 1: Block Diagram

1.5 Visual Aid

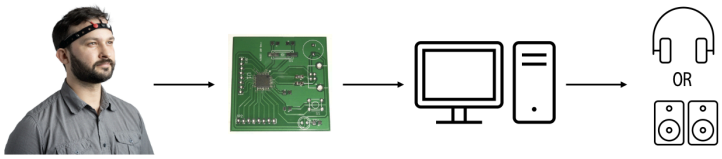

Figure 2: Visual Aid

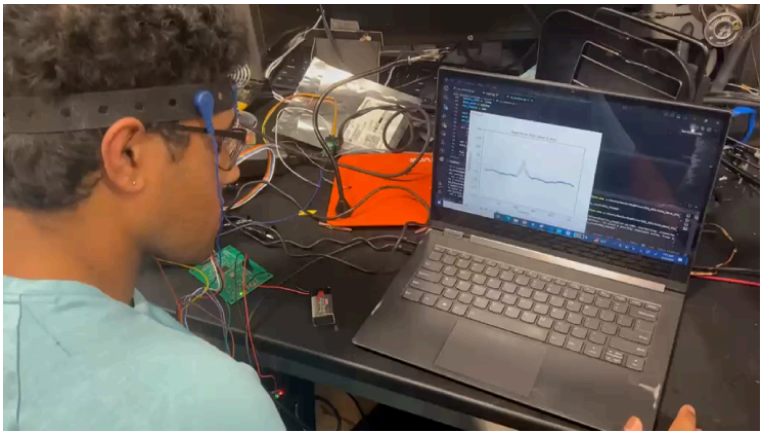

Figure 3: Real Life Representation of Visual Aid

2. Design

2.1 Design Procedure

We started our design process by constructing an overview of the subsystems that would be required for the project, and they broadly came to be an amplifier for the low voltage EEG signals (ADC Subsystem), a microcontroller to receive and transmit data (microcontroller subsystem), a computer program to analyze the EEG waves from the user (computer and user subsystem) and a long lasting battery to power the entire system (power subsystem).

We looked at various schematics, including the OpenBCI Cyton Board schematic. We included filtration for higher frequencies through a hardware filtration circuit before the ADC subsystem so that the EEG signals have lesser noise. We were initially planning on using a separate amplifier and analog to digital conversion subsystem, however, we realized that it would become too complicated and actually end up saturating the signal if we did so because of the multiple components.

Instead, we chose to use a specialized ADS1299 chip for the ADC subsystem which handles both amplification and digital to analog conversion for bio signals. For the power constraints, we chose to use regulators on the board that would directly convert the input voltage to the respective voltages needed for the microcontroller and the ADC subsystem. For the microcontroller, we chose to use the STM32WB5MMG because it used less energy (satisfying our power constraints) and also had an in-built low energy bluetooth BLE capability for communication. We also needed to procure an EEG wearable headset so that we could measure signals from our head. For the computer subsystem, we use publicly available EEG data and visualize them on an EDF file in order to verify if our program was working as intended.

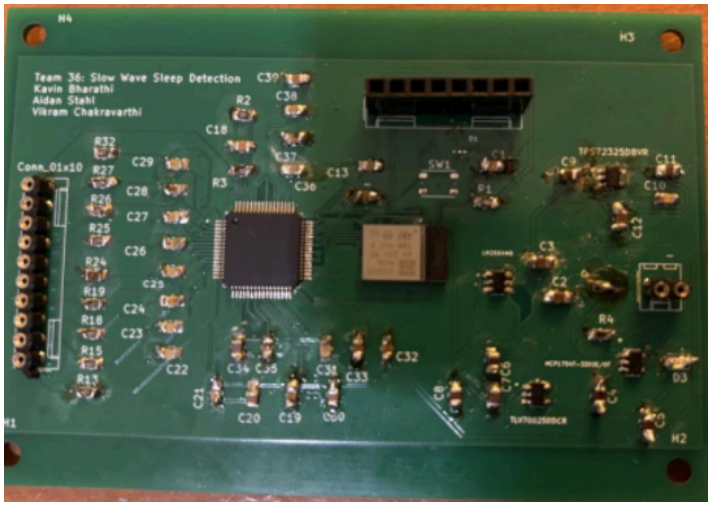

Figure 4: Our Custom PCB

2.2 Detailed Subsystem Overview

2.2.1 Power Subsystem

The power subsystem is responsible for generating the regulated voltage for the rails required by the STM32WB5MMGH6TR microcontroller and the ADS1299IPAGR analog front end. The MCU we used (STM32WB5MMGH6TR) accepts a power input voltage between 1.71 and 3.6 volts with a typical voltage of 3.3V. The 3.3V regulator is connected directly to the Vdd pin of the MCU. The ADC we used (ADS1299IPAGR) also utilized this 3.3V signal to power the digital logic (DVDD), which accepts a voltage between 1.8 and 3.6 volts. Additionally, we used -2.5V and 2.5V regulators to power the analog power supply pins of the ADC, providing a total range of 5 volts of input. The 2.5V regulator is powered by the 3.3V regulator. Additionally, we passed the 3.3V signal through an inverter to power the -2.5V voltage regulator with a -3.3V signal.

We considered using a single 3.3 V supply, but this would not provide enough voltage to the bipolar rails of the ADS1299 chip through the regulators. So based on these specifications, we chose to use a 9 volt lithium-ion battery to power a 3.3V voltage regulator. We also decided to keep the battery detachable and connected through a power input on the board. This allows for it to be changed if they are issued with the battery. Also our board, due to the regulators that we used, is able to accept batteries with voltages between 5V to 16V providing a flexible range for the user to choose from.
We also needed to make sure that the battery is able to last for at least ten hours, which is the time during which a user would be sleeping and using the circuit. For this, we performed a basic calculation to determine the amount of time our battery of 9V would last for. The ADS1299 draws about 8.14 mA of current, and the STM32WB5MMG draws about 5 mA of current. So, the total current I is:

$$I = 8.14mA + 5\ mA = 13.14\ mA$$

For ten hours of operation, the ideal battery capacity C is:

$$C = It = 13.14\ mA \times 10\ h = 131.4\ mAh$$

So, accounting for additional possible capacity due to bluetooth, our selected 9V battery had a higher capacity of 1000 mAh well exceeding the requirement.

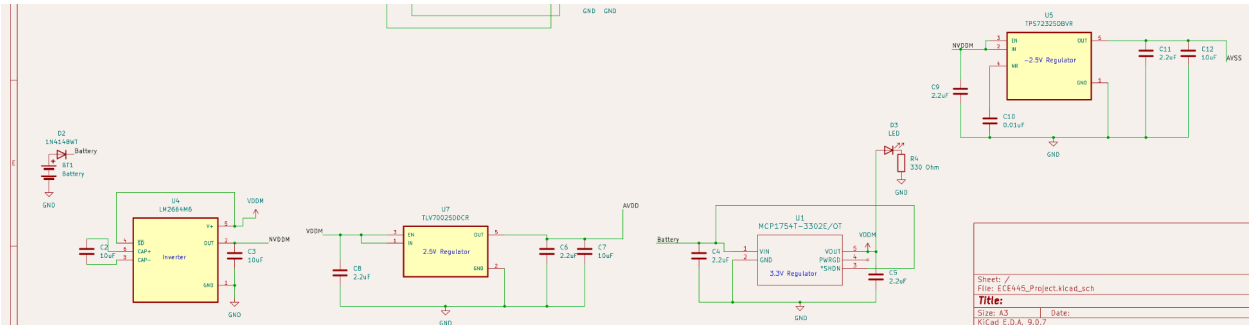

Figure 5: Power Subsystem Schematic

2.2.2 ADC Subsystem

This subsystem includes both RF filtering and Analog-to-Digital Signal Conversion. RF filtering conditions the electrode input signals so only valid EEG frequencies reach the EEG. We found that the electrode leads that are being used from the headset were acting as antennas and picking up radio-frequency signals. So, in order to filter these higher frequency signals out, we designed a low-pass filter consisting of a 1k Ohm resistor and 2.2nF capacitor. We chose these values based on researching the best components to filter out much higher frequency signals. With these components, using the following equation, we were able to find our cutoff frequency fc.

$$f_c = \frac{1}{2\pi RC} = \frac{1}{2\pi \cdot 100000 \cdot 1 \cdot 10^{-9}} = 1591.549\ Hz$$

Since the cutoff frequency represents the frequency corresponding to an output power of -3 dB or ½ relative to the input power, we decided that this was an ideal cutoff frequency for a number of reasons. First, the typical frequencies of EEG leads are around 100-200 Hz at a maximum. This cutoff frequency is high enough where we won't be at risk of accidentally filtering out signals we want and the original EEG signals at this frequency will be maintained. Bluetooth transmission operates in the 2.4 GHz range, which is much higher than our cutoff frequency. Based on the following calculations using the transfer function, we were able to determine that at bluetooth's frequency of 2.4 GHz, the input voltage seen at the beginning of the filtering circuit will realize an output voltage of around 6.631e-7*Vin or -123.568 dB relative to the input voltage.

$$|H(\omega)| = \frac{Vout}{Vin} = \frac{1}{\sqrt{(1\cdot 10^{-4})^2 (2\cdot\pi\cdot f)^2 + 1}} = \frac{1}{\sqrt{(1\cdot 10^{-4})^2 (2\cdot\pi\cdot 2.4\cdot 10^{9})^2 + 1}} = 6.631 \cdot 10^{-7}$$

In the dB range this is:

$$20log_{10}(6.631 \cdot 10^{-7}) = \quad -123.568\,dB$$

The main reason why we chose the ADS1299 chip was its Analog-to-Digital Converter that converts the conditioned analog EEG signals into synchronized digital samples. The ADS1299 has a high 24-bit resolution as well. The device is optimized for biomedical applications for microvolt measurements especially in the case of low-amplitude EEG signals with programmable gain instrumentation amplifiers (PGA), a multiplexer, simultaneous delta-sigma ADCs, internal references generation, and digital filtering. The delta-sigma ADC increases the resolution and the signal-to-noise ratio which is ideal for an application such as measuring EEG where the values are in microvolts and frequency bandwidth small in between 0-100 Hz. The PGA can be set to gains of 1, 2, 4, 8, 12, 24. This means that external amplifiers for the microvolt EEG voltages are not needed. Additionally, the channels are sampled simultaneously so as to maintain phase alignment so that it is more precise for applications such as working with brain activity and other medical applications. This ensures that there is no misalignment in the channels which could give inaccurate readings of EEG.

Some alternatives that we considered was using a separate amplifier and ADC subsystems. We thought that we would amplify the signal by a 10,000 gain and then it would be within the sufficient voltage range for the in-built STM32 ADC to plot the EEG. However, we noticed that there was a lot of saturation as shown in the image below.

One alternative design that we tried, used a separate analog amplifier followed by the STM32's internal ADC. In this approach, the EEG input would be amplified by 10,000 times so that the microvolt signals could be digitized by the STM32 internal ADC. However, this gain made the analog stage highly sensitive to motion artifacts, electrode offset and noise. As shown in Figure 6, the amplified waveform saturated when we tried this circuit with the EEG headset as input. That solidified our decision to use the ADS1299 for the reasons mentioned before.

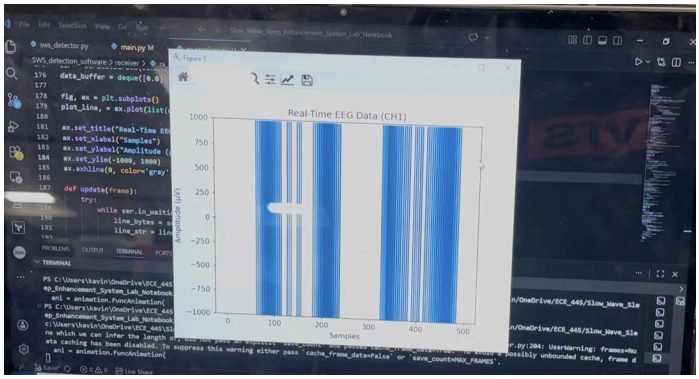

Figure 6: Saturated EEG input with alternative circuit

This solidified our decision to use the ADS1299 chip, which provided the accurate plotting of EEG as shown in Figure 7 below:

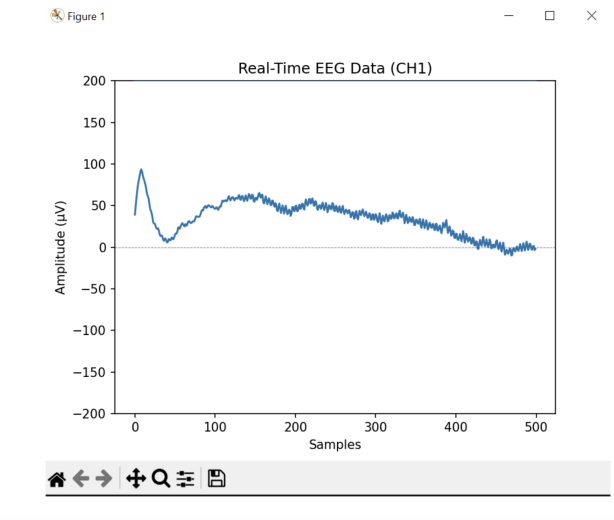

Figure 7: Accurate plotting of EEG

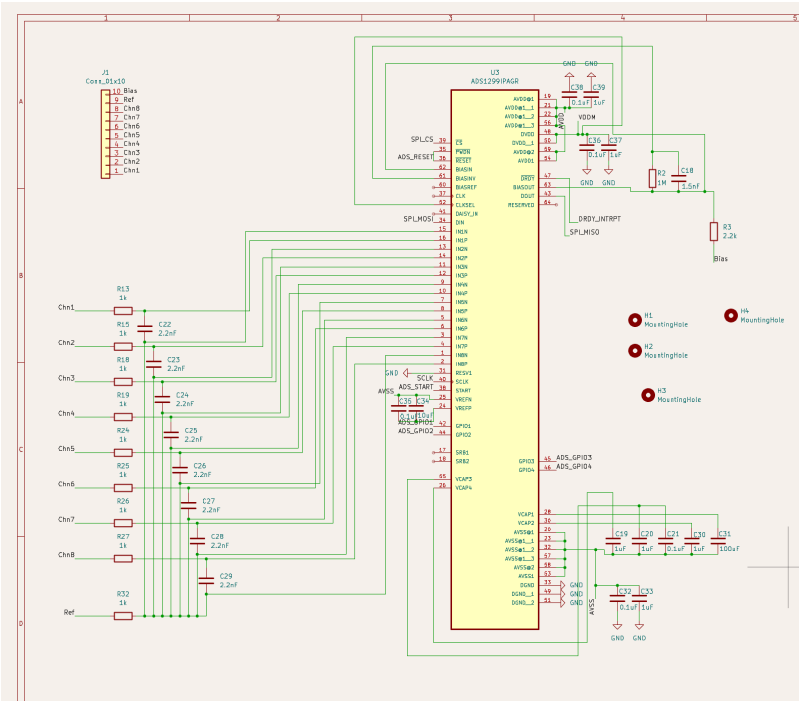

Figure 8: ADC Subsystem Schematic

2.2.3 Microcontroller Subsystem

The STM32 microcontroller subsystem receives ADC samples from ADS1299 from the SPI interface. Its main function is to configure the ADS1299 registers, manage startup sequencing, maintain sampling synchronization, package the multi-channel EEG samples into timestamped packets and transmit the data to the host computer. The BLE protocol provides low-power periodic wireless transmission making it suitable for overnight use.

The STM32WB5MMG was selected because it supports low power operation and has integrated Bluetooth communication. It has an Arm Cortex-M4 up to 64 MHz, a dedicated Cortex-M0+CPU for radio communication, and an integrated BLE radio. This dual core architecture is useful because ADC acquisition and wireless communication can be handled separately, reducing the likelihood of packet transmission interference with EEG sampling.
For each SPI frame sample frame, the ADS1299 outputs 24 status bits and eight EEG channels with 24 bits per channel. Therefore, each frame contains:

$$24 \text{ bits} + 8(24 \text{ bits}) = 216 \text{ bits} = 27 \text{ bytes}$$

At a sampling rate of 250 Hz, the required data throughput is

$$27 \text{ bytes/sample} \times 250 \text{ samples/s} = 6750 \text{ bytes/s} = 54 \text{ kb/s}$$

This throughput is well below the BLE 2 Mb/s physical-layer rate. The sampling period is

$$\frac{1}{250} = 0.004 \text{ s} = 4 \text{ ms}$$

So the ADS1299 DRDY signal asserts every 4 ms. With an 8 MHz clock, the time required to transfer one 216 bit frame is

$$\frac{216}{8 \cdot 10^6} = 27 \ \mu\text{s}$$

That is much smaller than the 4 ms sampling period, allowing the SPI transfer to complete before the next sample is ready. During the implementation, the STM32 communication architecture was validated through a UART debug path instead of BLE. The first verification was to see if the correct device ID of the ADS1299 was being read and if registers were being able to be written to or read from. However, we ended up using the UART for communication instead of BLE. This is because we ran into multiple issues with the soldering of the chip. The pad spaces were too close together, which caused multiple bridging issues. Additionally, we weren't able to see the pads underneath because they were not visible under the STM32. Figure 9 shows the pad spacing which caused the issues.

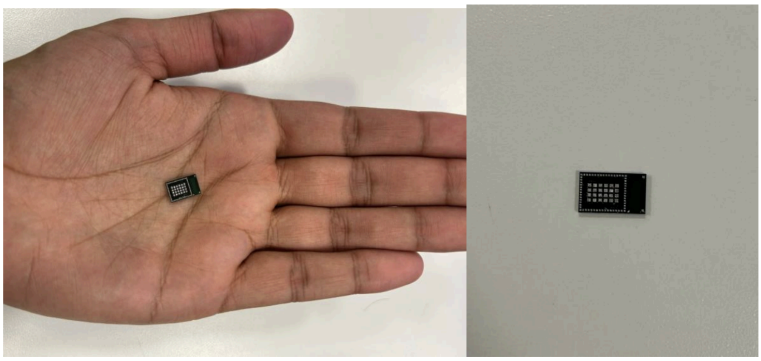

Figure 9: STM32 in use with pads underneath

A design alternative that we considered, but did not have enough time to implement was using the ESP32 instead. This is because it is easier to solder. Additionally, it has a more user-friendly interface, Arduino. The SMT32 CubeIDE which was used for programming was not user friendly and it was a big learning curve for us to begin to use the platform. Also, we were not able to use the BLE communication because we made a mistake in placing the antenna in a copper zone. It was not very clear in the documentation, but it was mentioned that the antenna should be placed in a no-copper zone. Due to the traces underneath the antenna, BLE communication was affected which made us UART wired communication instead. On the positive side, this improved our latency. Figure 10 shows the incorrect placement of the antenna.

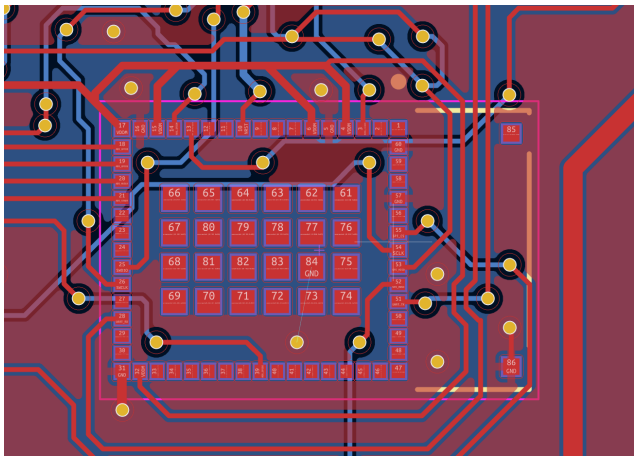

Figure 10: Traces underneath and near antenna affecting BLE

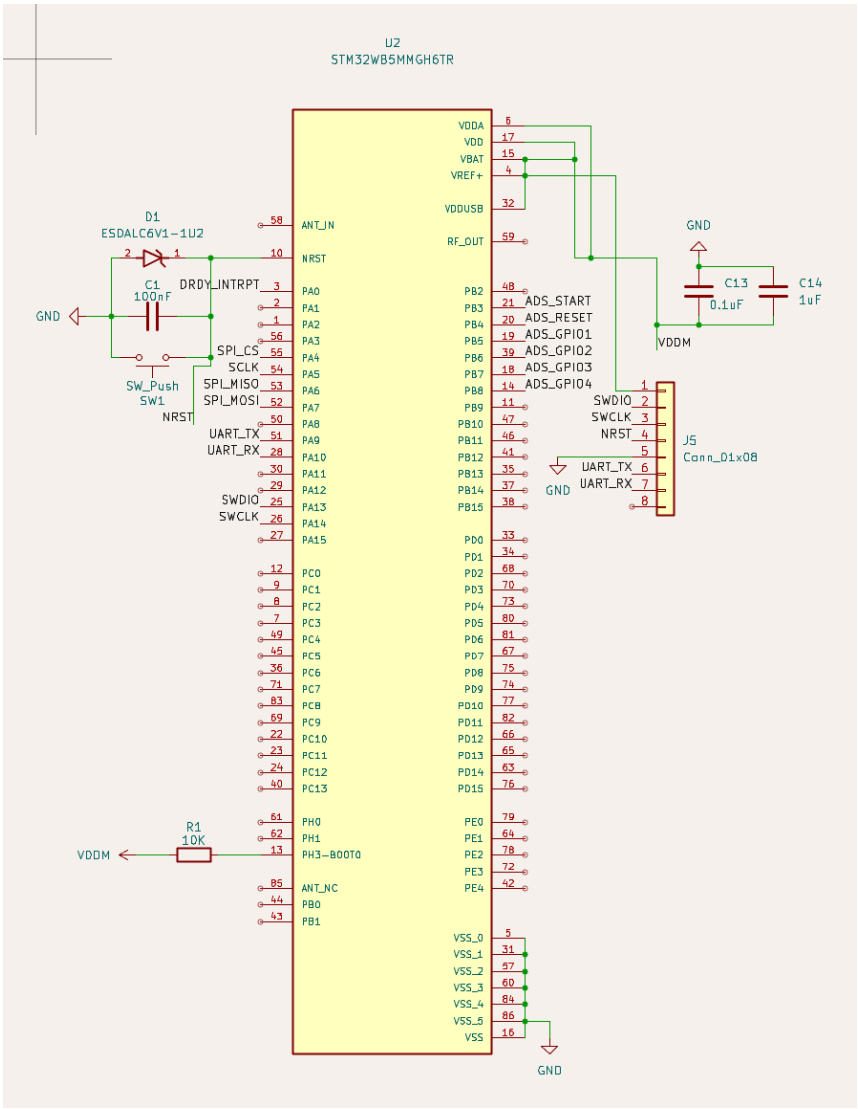

Figure 11: Microcontroller schematic

2.2.4 Computer Subsystem

The computer subsystem consists of three main parts: converting raw values from our ADC subsystem into valid readings in microvolts, identifying when the user is in Slow Wave Sleep, and playing pink noise whenever a slow wave is detected through a negative peak detector.

The data sent from the microcontroller subsystem to the computer is not in microvolts, so we had to use a specific formula to convert our raw ADC values back to microvolts.

$$\text{EEG Reading in } \mu\text{V} = \frac{Raw\ ADC\ Value * V_{Ref}}{Gain * 2^{(\#\ of\ Data\ bits)-1}}$$

$V_{Ref} = 2.5\,V$ and refers to Voltage reference. Gain specifies the amplitude of amplification for the EEG signals. The gain value can be specified when programming the ADS1299. With higher

gain, we experienced increased noise, so we had to test different values. We saw that Gain=4 served our purpose the best. The number of data bits in each sample of data is 24 as shown in the previous section.

While livestreaming our EEG data, we use an open source Command-line sleep analysis tool called YASA (Yet Another Spindle Algorithm) to read digitized EEG data and determine if the user is experiencing slow-wave sleep. We learned that YASA needs at least 30 seconds of EEG data to run a machine learning algorithm for identifying slow waves. To account for live data, we decided to use a sliding window that updates constantly with 30 seconds of data. Later, we realized that slow waves were still being detected when the user was awake or in light sleep because . To combat this, we decided to run the detector only when we have confirmed that we are in deep sleep (Official N3 sleep stage). In addition to slow wave detection, YASA also has another tool to estimate the possible sleep stage a person may be in. This tool takes in another sliding window of 5 minutes and guesses what sleep stage we might be in. Once we have confirmed that we are in slow wave sleep, we run the detector.

Next, our peak detection algorithm uses an adjustable max voltage cutoff. Since slow waves are defined by low-frequency high amplitude waves, if our peak falls below this cutoff, it's classified as a slow-wave peak and pink noise is subsequently played. The pink noise is played within 300 ms of the upstate of the slow-wave. Since the negative peak defines where the slow-wave upstate begins, we use the max voltage cutoff as an estimation to find where the minimum peak of the slow-wave is since actually detecting this peak in a live setting is very difficult. The base limit for our voltage cutoff is -80 uV. So, the highest our voltage cutoff can be is -80 uV. The voltage cutoff is calculated by taking the last five seconds of data and computing the voltage that falls in the bottom 15th percentile. Therefore, if a large amount of the data falls significantly below -80 uV, such that the bottom 15th percentile is less than -80 uV, the cutoff will be adjusted to a lower value (-90 uV for example). The adjustable cutoff is updated every 2 seconds. We settled on using the bottom 15th percentile of data with a base value of -80 uV because we tested different values and found that this returned the most accurate detection of slow-waves. Once the wave crosses the cutoff, the pink-noise is played. Since the cutoff is based on where the lowest portion of the wave falls, the peak is very close to the cutoff, so our pink noise is played very close to the actual peak. The reason we chose to use an adjustable cutoff was because we found that slow waves differ from person to person. Some people might have slow-waves that cross the -100 uV range or lower. As a result, keeping a constant cutoff would result in inaccurate detection of slow wave peaks and misclassification of slow waves. Additionally, slow waves are often much higher upon initially falling asleep. As the user sleeps, the slow wave amplitudes seen during slow-wave sleep become lower. Therefore, the adjustable cutoff, once again, keeps slow wave classification and peak detection accurate.

Finally, the computer subsystem is responsible for playing pink noise. Once the peak is detected using the method described above, our algorithm signals for pink noise to play and plays a 50 ms burst of pink noise. The 50 ms burst was found to be the most effective at preventing

sleep-related disorders and memory consolidation issues by the studies our sponsored group provides, and is therefore the value we chose.

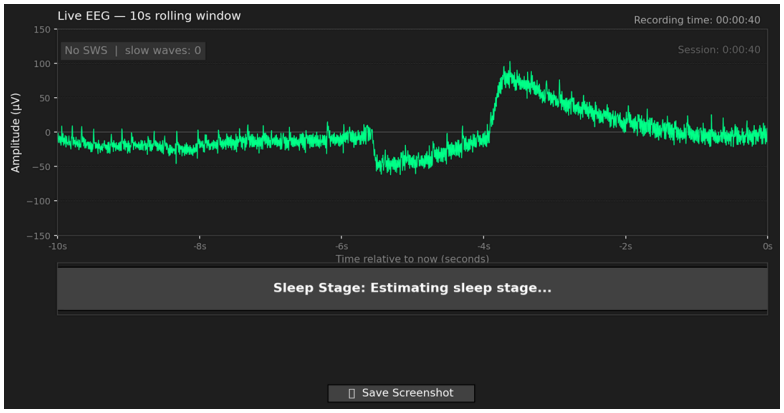

Figure 12: GUI Showing Calibration Before Slow Wave Detection

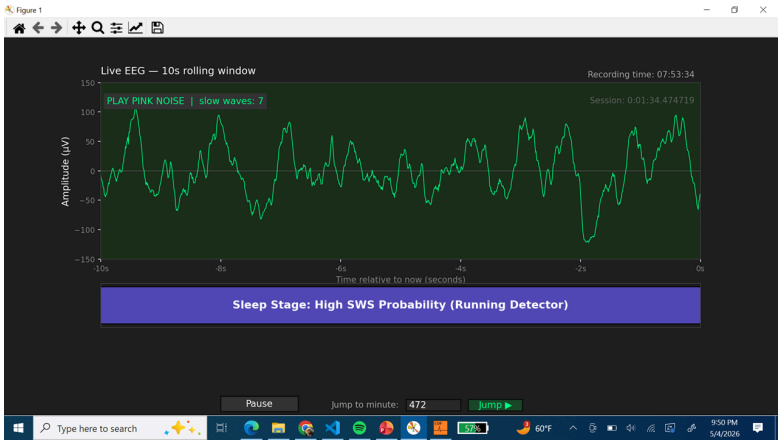

Figure 13: Running Slow Wave Detector on Sample EEG Data

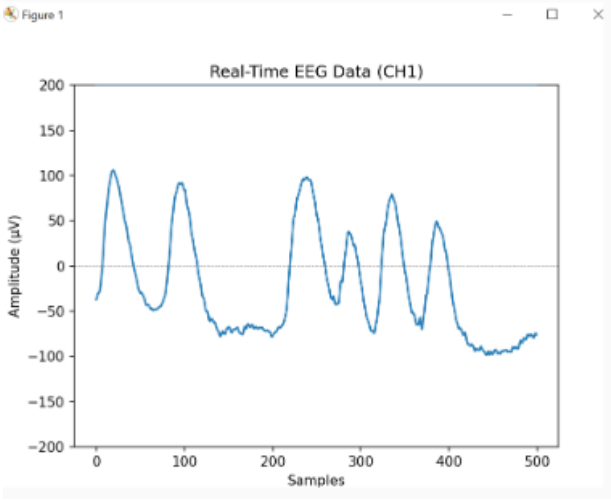

Figure 14: Testing Responsiveness to Blinking

2.2.5 User Subsystem

The user subsystem is the final interface between the device and the user. It consists of the wearable EEG headset, the electrode contact points, and the audio output device (the computer/laptop) that plays the pink noise during detected slow-wave sleep. The headset continuously supplies EEG signals to the ADC subsystem, while the audio output delivers stimulation after the computer subsystem detects the correct sleep stage.

This subsystem is important because the device must operate during sleep without disturbing the user. Therefore, the headset must maintain stable electrode contact while remaining comfortable enough for overnight use. If the headset shifts, loses contact, or becomes uncomfortable, the EEG signal quality decreases and the user may wake up, preventing the system meeting its purpose. We noticed that when you move or touch the headset the signal quality gets affected. To help combat this, we reduced the number of total channels to one channel. Additionally, the bias electrode should be connected to the head. As an alternative, we would recommend using electrodes with better connectivity and less sensitivity to noise. Additionally, wearing a gel on the forehead between the electrodes would help in improving the signal quality.

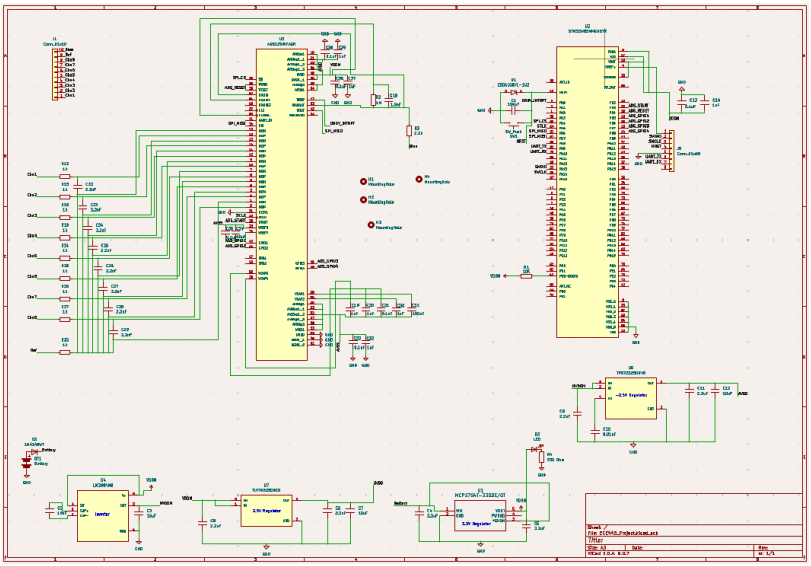

Figure 15: Full KiCAD Schematic

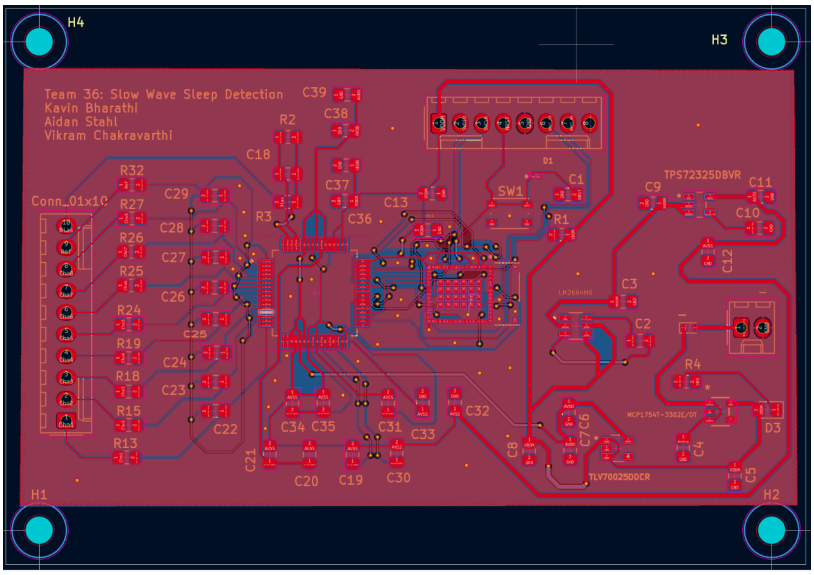

Figure 16: EEG Acquisition PCB Layout

3. Design Verification

3.1 Computer Subsystem

3.1.1 Requirement

Slow-wave sleep is detected correctly. Pink noise is played within 300 ms or less after the up-state of a slow-wave sleep wave is correctly identified.

3.1.2 Verification

Use sample EEG data to test if the algorithm is successful at identifying slow-wave sleep. If the algorithm identifies slow-wave sleep during the provided slow-wave sleep sample data and successfully identifies other sleep cycles or not sleeping, when non-slow-wave sleep sample data is provided, the algorithm is working as intended. Use sample EEG data to test if the software correctly identifies when a slow-wave sleep up-state is occurring by making the program print a line indicating the up-state wave has been detected and look at the EEG sample data at this specific time to determine if an up-state is truly occurring. After the up-state of the wave has correctly been identified, make sure the pink noise MP3 file is played within 150 ms of this detection.

As can be seen in Figure 17 below, pink noise is played at the timestamp 7:30:45.410. The peak is detected in the rectangle at 7:30:45.205. This is within 300 ms of detecting the slow wave peak thus showing that slow wave sleep is detected correctly and pink noise is also played at the appropriate time.

Figure 17: Timestamp of Pink Noise Versus EEG data

3.2 Microcontroller Subsystem

3.2 Requirement

Able to see digital signals corresponding to EEG on a receiver with sampling rate of 250 Hz and resolution of 24 bits. SPI communication works as expected at 8 MHz with no missed DRDY interrupts, no bit errors.

3.2.2 Verification

Use a signal generator to generate a specific frequency within the range of 1-100 Hz. Set the voltage at a low range, similar to what would be expected from an EEG lead. Check amplitude and received signals to see if they are the same as the signal from our SG. Validate that amplitude corresponds to gain scaling. If Bluetooth communication is not working, try to see if EEG signals are received through wired UART connection. Test with multiple different frequencies from the SG within a range of 1-100 Hz and change the amplitudes as well to simulate representing multiple different voltage peaks.

We used a signal generator to supply a low frequency sine wave to our PCB. The frequency was set to 1 Hz and amplitude at 1 mV. Our BLE was not working, so we verified it with a wired connection through UART instead. We were able to see correct values being plotted and signals adjusting to amplitude and frequency changes as seen in Figure 18 below.

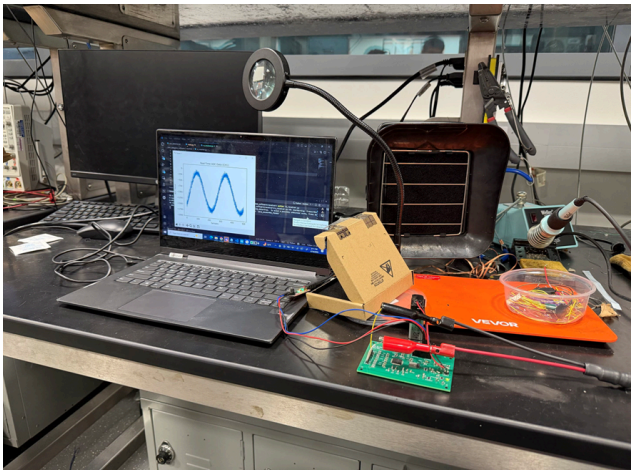

Figure 18: Signal generator being plugged into board and sine wave plotting

3.3 Power Subsystem

3.3.1 Requirement

The microcontroller and ADC must receive between 1.8V and 3.6V (ideally around 3.3V). The analog positive power supply pins (AVDD) must receive a voltage between 2.475V and 2.625V (ideally around 2.5V). The analog negative power supply pins (AVSS) must receive a voltage between -2.475V and -2.625V (ideally around -2.5V).

3.3.2 Verification

Power the regulators with a 6V lithium-ion battery and measure the voltage received at the Vdd input pin of the MCU and the DVDD pins of the ADC. Confirm that the values are around 3.3V and within the required range of 1.8V and 3.6V. Power the regulators with a 6V lithium-ion battery and measure the voltage received at all AVDD pins of the ADC. Confirm that the values are around 2.5V and within the required range of 2.475V and 2.625V. Power the regulators with a 6V lithium-ion battery and measure the voltage received at all AVSS pins of the ADC. Confirm that the values are around -2.5V and within the required range of -2.475V and -2.625V.

We used a multimeter to measure the voltages across each of the points on the PCB for the STM32 and the ADS1299 chip. We were able to measure 2.49 V and -2.49 V for the ADS1299, and 3.29V for the STM32. Additionally, we had an LED on the board which would power on when the STM32 is being supplied with 3.3V (and therefore the ADS1299 is being supplied with 2.5V and -2.5V) which is lit up in the Figure 19 below.

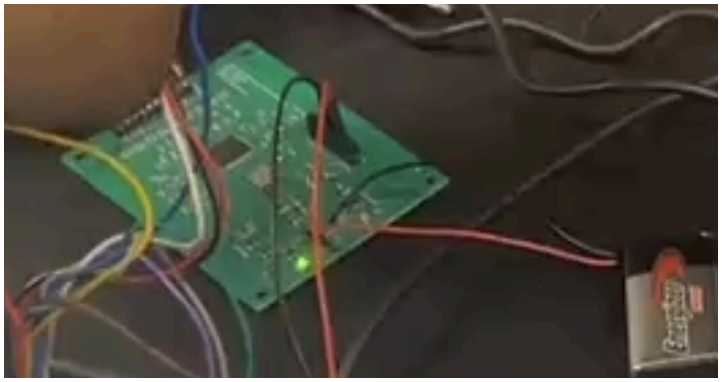

Figure 19: LED on board being lit up when powered

3.4 ADC Subsystem

3.4.1 Requirement

High-frequency signals, such as bluetooth's 2.4 GHz frequency, should be filtered out.

3.4.2 Verification

Use a signal generator to generate a high-frequency signal in the range of 2-3 GHz with a low voltage similar to what an expected EEG voltage would be. Ensure the ADC's digitized output is very close to zero with at least a suppression of - 90 dB or greater (for frequencies >= 2.4 GHz). Test with multiple frequencies in the range of 2-3 GHz and multiple voltages to ensure signals remain suppressed.

We inputted a sine wave signal with a frequency of 3 GHz and saw that the input was suppressed as shown in Figure 20 below. This shows that we were not able to measure the sine wave meaning that the ADC was successfully suppressing noise in that frequency range.

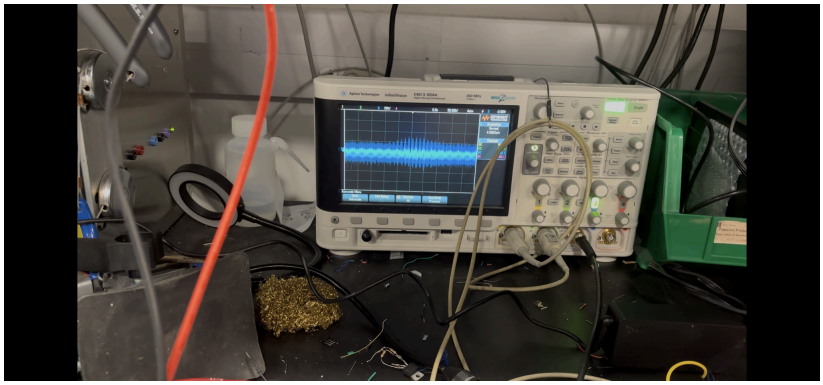

Figure 20: High frequency input into board being suppressed

When inputting a low frequency signal with 1 Hz, we were able to get a proper reading of the sine wave as shown in Figure 21 below. This shows that we are successfully able to see signals with a low frequency.

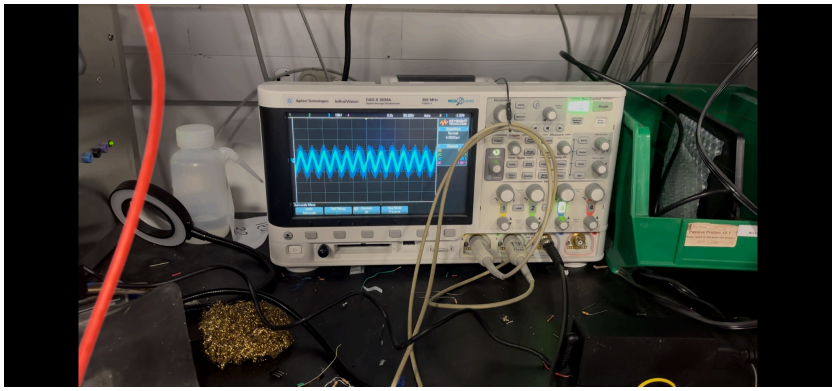

Figure 21: Low Frequency input being read

3.5 User Subsystem

3.5.1 Requirement

If given a survey to multiple users for how comfortable the device is, the average rating should be 4 out of 5. (1 is least comfortable, 5 is most comfortable)

3.5.2 Verification

We will just have multiple users test the finished product. Each user will complete an anonymous survey about their experience. One of the metrics will be the comfortability of wearing the headset while sleeping

We had multiple users try the EEG headset and had a good response to the comfort rating with an average rating of 4.6. Figure 22 shows two users trying the headset who reported good comfortability ratings of 5/5 each.

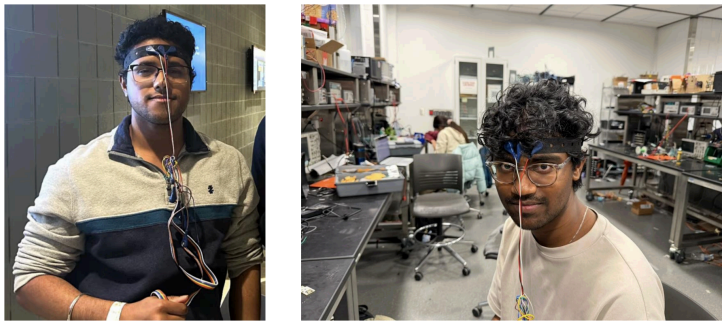

Figure 22: Multiple users testing the EEG headset

4. Costs

4.1 Parts

Part	Manufacturer	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
Analog to Digital Converter (ADS1299IPAG (8-ch AFE, TQFP)) x 2	Texas Instruments	73.30	65.151	146.6
Microcontroller (STM32WB5MMGH6 TR (BLE module)) x 2	STMicroelectronics	12.14	9.53	24.28
MCP1754 (3.3V regulator) x 2	Microchip	0.54	0.46	1.08
LM2664 (Voltage Inverter) x 2	Texas Instruments	1.08	0.776	2.16
TLV70025DDCR (2.5V Regulator) x 2	Texas Instruments	0.23	0.13760	0.46
TPS723 (-2.5V Regulator) x 2	Texas Instruments	3.45	2.03	6.9
GRM21BR60G107ME 11L (100 uF capacitor) X 10	DigiKey	0.29	0.29	2.9
CC0805KKX5R9BB22 5 (2.2 uF cap) X 20	DigiKey	0.13	0.13	2.64
Total				$187.02

Table 1: Cost Analysis/Bill of Materials

4.2 Labor

Team Member	Hours of Work	Hourly Rate ($)	Total ($)
Kavin	10	16.00	160.00
Vikram	10	16.00	160.00
Aidan	10	16.00	160.00
Total	10	16.00	480.00

Table 2: Weekly Labor Costs

4. Conclusion

4.1 Discussion of Societal Impact, Engineering Standards, Ethics, and Safety Considerations

The Societal impact of the Slow Wave Sleep Detector is improving widespread treatment for sleep and mental disorders. It has been proven through the studies discussed in this document that pink noise has many beneficial effects for improving sleep quality. The restorative outcomes from improved sleep are essential to improving widespread treatment. The non-invasive nature of this project allows it to be applied across a large group of users freely.
This project comes with several ethical considerations that we must take into account when designing. First, we must add voltage surge protection connected to the EEG output leads because if the voltage were to surge, the user could be harmed. Another potential ethical concern is patient information. Our device is going to be used in a medical/healthcare application. HIPPA cites established guidelines for protecting patient information that must be followed. Any use of patient information in our development and/or testing stages must abide by these guidelines. These safety and ethical guidelines are further enforced by following IEEE and ACM standards. More specifically, this project follows the IEEE Code of Ethics Principles 1 and 3, and aligns with the Association for Computing Machinery Code of Ethics Principles 1.2, 1.6, and 2.5. IEEE Principle 1 prioritizes the health, safety, and welfare of the public through ensuring that the EEG headband is non-invasive and ensuring that safe levels of pink noise are played. IEEE Principle 3 emphasizes being transparent with how effective this device is for sleep-enhancement. ACM Code of Ethics is followed by securing healthcare data according to widely accepted standards.

Electrical and mechanical safety have been considered in this project. We made sure that the system follows through with low-voltage battery operation. This allowed for an easier time to create circuit protection that will also protect circuit components. To address mechanical safety, the PCB will be situated in a protective, 3D printed enclosure. The purpose of this enclosure is to prevent accidental contact with outside environments that could damage solder joints or traces.

4.2 Accomplishments, Uncertainties, and Future Work/Design Alternatives

Our group was successful at creating a PCB to measure analog EEG brain wave voltages, convert them to digital values, and plotting and processing the data in order to detect slow-wave sleep and play timed bursts of pink noise at the upstate of slow waves through a minimum peak detection algorithm. We were able to verify proper functioning with multiple methods. We supplied a sine wave from a waveform generator with a voltage peak to peak amplitude of 1 mV and verified that the sine wave was maintained on our plots with the correct voltage amplitude. We cross referenced this with an oscilloscope and confirmed that the plots matched. Additionally, we changed the parameters of the waveform generator, including the amplitude and frequency, and confirmed that the changes reflected accurately on our plots.

Unfortunately, due to hardware problems and soldering issues, our circuit which was initially functioning without issue, encountered a problem, which we were unable to resolve in the remaining course time. We attempted to debug the issue by replacing the STM32 microcontroller and initially were able to get the PCB functioning again, but ultimately we encountered another problem, which we once again attributed to soldering problems. Additionally, we replaced the ADS and unfortunately the issue prevailed. *Figure 19* shows the issue we were encountering. The plots were extremely noisy and showed invalid data.

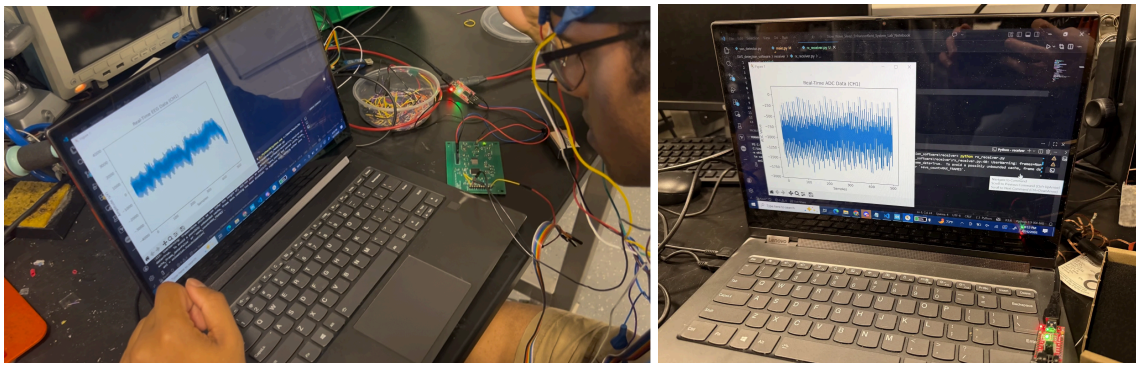

Figure 19: Demonstration of Circuit Issue

This project has several areas that can be improved in the future. Currently, our design involves measuring one EEG signal and a reference. In the future, one thing we could focus on is expanding this to measure eight EEG signals for more accurate slow wave detection. Additionally, we can focus on software improvements to minimize calibration time and detect slow wave sleep with higher accuracy and in less time. Potential design alternatives include using an ESP32 in place of the STM32 model we used due to a more user-friendly IDE and larger pins for easier soldering.

References

[1] *Hong-Viet V. Ngo, Thomas Martinetz, Jan Born, Matthias Mölle, Auditory Closed-Loop Stimulation of the Sleep Slow Oscillation Enhances Memory, Neuron, Volume 78, Issue 3, 2013, Pages 545-553, ISSN 0896-6273, https://doi.org/10.1016/j.neuron.2013.03.006.*

[2] *H.V. Ngo, & B.P. Staresina, Shaping overnight consolidation via slow-oscillation closed-loop targeted memory reactivation, Proc. Natl. Acad. Sci. U.S.A. 119 (44) e2123428119, https://doi.org/10.1073/pnas.2123428119 (2022).*

[3] *Marina Wunderlin, Marc A Züst, Elisabeth Hertenstein, Kristoffer D Fehér, Carlotta L Schneider, Stefan Klöppel, Christoph Nissen, Modulating overnight memory consolidation by acoustic stimulation during slow-wave sleep: a systematic review and meta-analysis, Sleep, Volume 44, Issue 7, July 2021, zsaa296, https://doi-org.proxy2.library.illinois.edu/10.1093/sleep/zsaa296*

[4] *"Cyton Data Format." OpenBCI Documentation, OpenBCI, 16 July 2025, docs.openbci.com/Cyton/CytonDataFormat/.*

[5] *Vallat, R., & Walker, M. P. (2021). An open-source, high-performance tool for automated sleep staging. eLife, 10. https://doi.org/10.7554/eLife.70092*

[6] *OpenBCI. (n.d.). OpenBCI EEG Headband Kit [Product page]. OpenBCI Shop. Retrieved Month Day, Year, from https://shop.openbci.com/products/openbci-eeg-headband-kit*

[7] https://www.build-electronic-circuits.com/printed-circuit-board-guide-beginners/ (image used from website)

[8] U.S. Department of Health and Human Services. (2025, March 14). *Summary of the HIPAA privacy rule*. HHS.gov; U.S. Department of Health and Human Services. https://www.hhs.gov/hipaa/for-professionals/privacy/laws-regulations/index.html

[9] Bluetooth SIG. *Part B — Link Layer Specification*, Core Specification (BLE), Section "connInterval". https://www.bluetooth.com/wp-content/uploads/Files/Specification/HTML/Core-54/out/en/low-energy-controller/link-layer-specification.html?utm_source=chatgpt.com.

[10] STMicroelectronics. (2024, October). *STM32WB55xx STM32WB35xx: Multiprotocol wireless 32-bit MCU Arm-based Cortex-M4 with FPU, Bluetooth 5.4 and 802.15.4 radio solution* (Datasheet DS11929 Rev 17). https://www.st.com/resource/en/datasheet/stm32wb55cc.pdf

[11] Garcia-Rill, E. (2009). Reticular Activating System. In L. R. Squire (Ed.), Encyclopedia of Neuroscience (pp. 137–143). doi:10.1016/B978-008045046-9.01767-8

Appendix

Requirements	Verifications	Verification Status
• Microcontroller and ADC must receive between 1.8V and 3.6V (ideally around 3.3V)	• Power the regulators with a 6V lithium-ion battery and measure the voltage received at the Vdd input pin of the MCU and the DVDD pins of the ADC. Confirm that the values are around 3.3V and within the required range of 1.8V and 3.6V.	Verification satisfied. Have an LED to indicate whether the battery is supplying voltage to all regulators. Voltage supplied to Microcontroller is 3.29V and 2.49V to ADS
• The analog positive power supply pins (AVDD) must receive a voltage between 2.475V and 2.625V (ideally around 2.5V).	• Power the regulators with a 6V lithium-ion battery and measure the voltage received at all AVDD pins of the ADC. Confirm that the values are around 2.5V and within the required range of 2.475V and 2.625V.	Verification satisfied. ADS1299 AVDD pin gets 2.49 V consistently.
• The analog negative power supply pins (AVSS) must receive a voltage between -2.475V and -2.625V (ideally around -2.5V).	• Power the regulators with a 6V lithium-ion battery and measure the voltage received at all AVSS pins of the ADC. Confirm that the values are around -2.5V and within the required range of -2.475V and -2.625V.	Verification was satisfied. AVSS was measured at -2.49V on average

Table 3: Power Subsystem Requirements and Verifications

Requirements	Verifications	Status
• At low-frequency inputs, the ADS must output the correct	• Use a signal generator to generate a low-frequency signal	Verification satisfied. Live feed was responsive to

digitized measured voltage values within a standard error of ±1%.	in the range of 1-100 Hz with a very low voltage and ensure the ADC's digitized output corresponds with the voltage set on the signal generator within a ±1% standard error. ● Test with multiple frequencies in the range of 1-100 Hz and multiple voltages to ensure accuracy remains.	frequency and amplitude changes. Values imputed on signal generator matched plotting window.
● High-frequency signals, such as bluetooth's 2.4 GHz frequency, should be filtered out.	● Use a signal generator to generate a high-frequency signal in the range of 2-3 GHz with a low voltage similar to what an expected EEG voltage would be. Ensure the ADC's digitized output is very close to zero with at least a suppression of - 90 dB or greater (for frequencies >= 2.4 GHz). ● Test with multiple frequencies in the range of 2-3 GHz and multiple voltages to ensure signals remain suppressed.	Verification Satisfied. Took a video of a high-frequency signal being inputted and filtered out.
● The sampling frequency should be greater than or equal to 200 Hz meaning the Nyquist frequency is at least 100 Hz, which ensures that there is no aliasing for signals	● Use a signal generator to generate a sinusoidal wave. Test inputs at various frequencies from 10 Hz to 100 Hz and ensure the digitized output is the correct	Verification Satisfied. Programmable ADS has 250 Hz set as sampling frequency. Slow Wave detector matches same frequency

less than or equal to 100 Hz.	frequency within a ±1% standard error. ● Plot the ADS's output while changing the voltage at a constant frequency and make sure that the output graph matches the frequency and voltages expected with a standard error of ±1. ○ Test with several frequencies to ensure accuracy is maintained.	

Table 2: ADC Subsystem Requirements and Verifications

Requirements	Verifications	Status
● Able to see digital signals corresponding to EEG on a receiver with sampling rate of 250 Hz and resolution of 24 bits.	● Use a signal generator to generate a specific frequency within the range of 1-100 Hz. Set the voltage at a low range, similar to what would be expected from an EEG lead. ● Check amplitude and received signals to see if they are the same as the signal from our SG within a ±1% standard error. ● Confirm that the sampling rate is within ±1% standard error of actual sampling rate of 250 Hz. ● Validate that amplitude corresponds to gain scaling.	Verification satisfied. For both our breadboard demo and PCB, we had sampling code in our STM to send the digitized sine wave data. Utilized UART for this.

	• If Bluetooth communication is not working, try to see if EEG signals are received through wired UART connection. • Test with multiple different frequencies from the SG within a range of 1-100 Hz and change the amplitudes as well to simulate representing multiple different voltage peaks.	
• SPI communication works as expected at 8 MHz with no missed DRDY interrupts, no bit errors.	• Check if the digital information at the DRDY pin corresponds to the EEG received on the microcontroller output. • Check if proper MISO/MOSI transitions are made, SPI clock frequency matches. • Make sure that DRDY pins toggle at the correct interrupt times.	Verification satisfied. Initially checked data being transferred to my laptop's COM port on putty to double check data integrity. Conducted this test using UART connection from our PCB to laptop.
• Bluetooth operates with latency below 100 ms with packet loss rate less than 2% and throughput of 54 kbps.	• Timestamp the transmission and reception to compute actual latency. • Measure packet loss by streaming known data streams for a certain fixed time.	Verification not satisfied. We positioned our microcontroller's BLE antenna in an incorrect area of our pcb. This limited the functionality of BLE. We had to instead transfer EEG signals through UART.

Table 3: Microcontroller Subsystem Requirements and Verifications

Requirements	Verifications	Status
• Slow-wave sleep is detected correctly.	• Use sample EEG data to test if the algorithm is successful at identifying slow-wave sleep. If the algorithm identifies slow-wave sleep during the provided slow-wave sleep sample data and successfully identifies other sleep cycles or not sleeping, when non-slow-wave sleep sample data is provided, the algorithm is working as intended.	Verification successful. Ensured that detected slow waves were in the "Slow wave sleep" stage (N3). Analyzed the amplitude, frequency, duration, and other qualities of a slow wave to see if our detection was right.
• Pink noise is played within 150 ms or less after the up-state of a slow-wave sleep wave is correctly identified.	• Use sample EEG data to test if the software correctly identifies when a slow-wave sleep up-state is occurring by making the program print a line indicating the up-state wave has been detected and look at the EEG sample data at this specific time to determine if an up-state is truly occurring. • After the up-state of the wave has correctly been identified, make sure the pink noise MP3 file is played within 150 ms of this detection.	When running slow wave sleep detection, we had a log file detailing when pink noise was played. We compared that timestamp to the EDF file and saw that pink noise played within 300 ms of detecting a slow wave.

Table 4: Computer Subsystem Requirements and Verifications

Requirements	Verifications	Status
• If given a survey to multiple users for how comfortable the device is, the average rating should be 4 out of 5. (1 is least comfortable, 5 is most comfortable)	• We will just have multiple users test the finished product • Each user will complete an anonymous survey about their experience • One of the metrics will be the comfortability of wearing the headset while sleeping	Verfication successful. Asked multiple people to wear the headset, and they all said that it is comfortable.

Table 5: User Subsystem Requirements and Verifications