UNIVERSITY of SOUTH FLORIDA
College of Engineering
Department of Electrical Engineering

EEL4914 Fall 2024

Functional Near Infrared Spectroscopy Headset (fNIRS)

FINAL REPORT

- Tristan Valenzuela
- Shovan Shakya
- Thang Pham
- Dr. Ashwin B. Parthasarathy (Advisor)

Table of Contents

1. Executive Summary

1.1 Project Overview

The fNIRS-Based Brain-Computer Interface (BCI) Headset project, in collaboration with the USF Translational Optics Imaging and Spectroscopy Lab, aims to develop a portable, user-friendly device that translates brain activity into actionable outputs using Functional Near-Infrared Spectroscopy (fNIRS). Designed for research and assistive applications, the headset features a lightweight, ergonomic design, real-time data acquisition at 100 Hz, and high spatial resolution with multi-channel configurations. Its modular software, built on STM32, integrates seamlessly with an external PC for visualization and computation via a Python-based GUI.

1.2 Purpose and Scope

Purpose

- Provide an open-source, portable, and cost-effective BCI solution for neuroscience, biomedical, and bioelectronic research and assistive technology development.
- Design a lightweight, ergonomic headset with flexible materials for prolonged and comfortable use.
- Enable applications in cognitive monitoring, rehabilitation, assistive technologies, and user-interface research.

Scope

- The design of the electronic and mechanical hardware of the headset.
- The design of the software and communication interface.
- Performance requirements such as timing, maximum operating time, and data transfer rate.

1.3 Definitions, Acronyms, and Abbreviations

- BCI: Brain-Computer Interface - a technology that enables communication and control between the human brain and external devices.
- fNIRS: functional Near-Infrared Spectroscopy - fNIRS (functional near-infrared spectroscopy) is a non-invasive imaging technique that measures brain activity by detecting changes in blood oxygenation using near-infrared light.
- LED: Light Emitting Diode
- LS: Long Separation
- SS: Short Separation
- IR: Infrared
- HbO: Oxyhemoglobin
- HbR: Deoxyhemoglobin
- ADC: Analog to Digital Converter
- PCB: Printed Circuit Board
- MCU: Microcontroller Unit

1.4 References

[1] E. Jeong, M. Seo, and K.-S. Kim, "Guide for wavelength selection of LEDs for fNIRS systems," in *2022 The 22nd International Conference on Control, Automation and Systems (ICCAS 2022)*, Busan, Korea, Nov. 27–Dec. 1, 2022, pp. 1819–1822.
[2] A. L. Gropman and A. Anderson, "Novel imaging technologies for genetic diagnoses in the inborn errors of metabolism," *Journal of Translational Genetics and Genomics*, vol. 4, pp. 429-445, Nov. 2020. DOI: 10.20517/jtgg.2020.09.

2. Product / Service Description

2.1 Product Context

The main method for reading brain signals is through a modality known as the Electroencephalogram (EEG). This modality directly converts the electrical potential discharged by the brain through the skull into a readable signal that can be sent to a host computer for interpretation. However, several alternatives have their respective benefits over EEG, one of which is known as functional Near-Infrared Spectroscopy (fNIRS). fNIRS has the added benefit of higher spatial resolution, portability, and tolerance to user movement which creates a more comfortable experience for the user. fNIRS utilizes near-infrared light to capture brain signals due to neurovascular coupling. fNIRS uses the absorption of light due to oxygenated (HbO) and deoxygenated hemoglobin (HbR) to analyze the activation of signals from the brain.

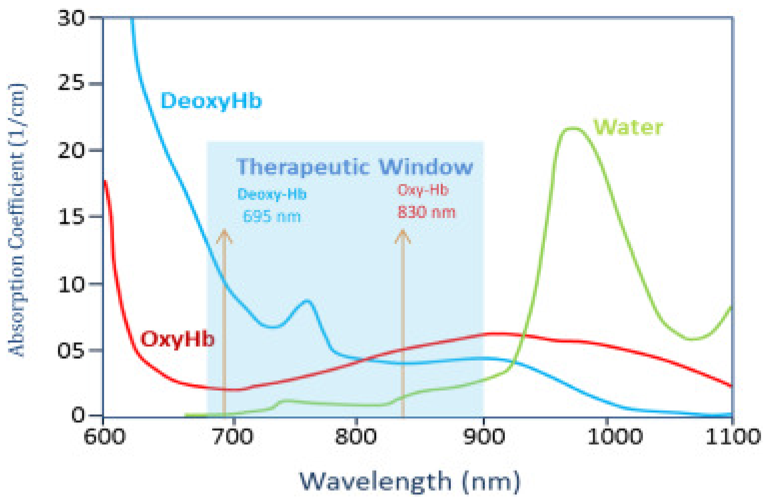

Figure 1. fNIRS Absorption Coefficient vs Wavelength indicating optimal optical window. [2]

An fNIRS BCI would utilize multiple source LEDs and photodiode detectors to create a map of brain signals which can be used to interpret specific areas of the brain that are being activated. The detected brain signals can then be used to control an external system such as software running on a host PC, or a robot.

The typical implementation of an fNIRS system utilizes a near-infrared light source such as a 730nm to 850nm wavelength LED [1]. The LED is placed on the skin while a photodiode which is sensitive to near-IR wavelength is placed near the source LED. This setup is able to detect slight variations in the HbO and HbR on the point of contact due to a relationship with light absorption.

This source-detector configuration can be placed on a user's head to detect changes in HbO and HbR due to neuronal activity which allows the possibility of a BCI system based on this modality.

A Python-based visualization program is then used to display the fNIRS-acquired brain signals.

2.2 Assumptions

Hardware Assumptions

- Price and cost optimization is not a priority, as this product is aimed at being an open-source research project and not one for mass production.
- Fundings will be sufficient to cover all material costs, including parts meant for prototyping (such as development boards).
- The design will be based on STM32 to be able to meet the software requirements.

Software Assumptions

- STM32 CubeIDE will be available for software development, compilation, flashing, and debugging.
- STM32 HAL libraries and functions will remain consistent. Any changes will be updated in the fNIRS software accordingly.
- An external PC is available to run a separate Python / MATLAB program for GUI and computation. The PC is also fast enough to run the software scripts.
- The external PC has a USB slot available for the USB to UART FTDI chip so the fNIRS device can communicate serially to the PC.

2.3 Constraints

Hardware Constraints

- Due to the large number of photodiode detectors, and the high cost of a single TIA, to meet the low-cost requirements of this design, the photodiodes must be individually multiplexed into TIAs.
- The MCU's clock frequency should be sufficient to handle the total functionality required of the headset which involves high GPIO switching speed, high ADC sampling rate, and high output communication baud rate. The selected STM32H7 meets each requirement, however, to meet its maximum clock frequency there must be active/passive cooling present.
- The headset must be flexible to conform to the shape of the wearer's head. Creating a flexible PCB will resolve this requirement.

Software Constraints

- Due to the tight timing constraint of the device, the MCU will have very little time to collect ADC data before they are gone. The MCU must do this while also performing background tasks such as communicating data to an external PC. All tasks must be executed promptly to get an accurate 100Hz reading.
- Ideally, the cost of the MCU should be as low as possible without compromising performance. Since flash and RAM size, as well as processing speed is associated with cost, it is important to optimize the software to reduce memory and speed requirements and keep only the necessary functionalities.
- Since the microcontroller cannot exceed 350MHz internal clock speed to prevent overheating, certain workarounds such as ping-ponging buffers must be done to ensure the 100Hz sampling rate is met.

2.4 Dependencies

Hardware Dependencies

- The SMD 735/850 LED will be used to match the requirement of both fNIRS wavelength light.

Software Dependencies

- STM32 CubeIDE will be available for software development, compilation, flashing, and debugging.
- The software duties will be performed promptly as long as the system clock is of sufficient speed.
- Python-based libraries such as numpy, matplotlib, pyqtgraph.
- MATLAB will be able to interpret the CSV file as long as it follows a specified format.

3. Requirements

3.1.1 User Interface Requirements

- The fNIRS headset can show a complete scan at a frequency of 100Hz.
- The band that wraps around the forehead should be made of flexible cloth, and the underlying PCB should also have the flexibility to make sure the sensor and source make good contact with the skin.
- The headband and the biosensor design should also be comfortable for the user to allow for prolonged usage.
- A data visualization program should also be developed to provide a graphical perspective on the sensor data, along with UI elements such as recording data to CSV and putting markers.

3.1.2 Performance

Hardware Requirements

- The system must have 8-16 channels with a minimum of 4 source LEDs. This should be accomplished by maximizing the number of source-detector pairs on the headset.
- The headset must be under 2 lbs.
- The system must have a communication interface between the host computer and the main processor.
- The design must have adjustable gain at the Transimpedance Amplifier to account for the varying separation distances of the detectors.
- The LED must emit light at 735nm and 850nm to meet the necessary optical window.

Software Requirements

- The firmware must successfully extract all analog readings at an appropriate time to meet the 100Hz sampling rate. The software must not miss any analog readings within a scan cycle.
- The communication interface implemented (UART) must be sufficiently fast to leave as much time as possible for the data collection yet is implemented in such a way that the number of error transmissions is <5% of the total transmissions.
- The firmware must successfully perform its duties such as toggling the trans-impedance amplifiers, IR LEDs, and multiplexer inputs at the specified time.
- A real-time data visualization and recording program needs to be developed to display the real-time data and also have features such as saving data to CSV or other appropriate formats and putting markers on data.
- The software should be modular and scalable for maintainability and future integrations.

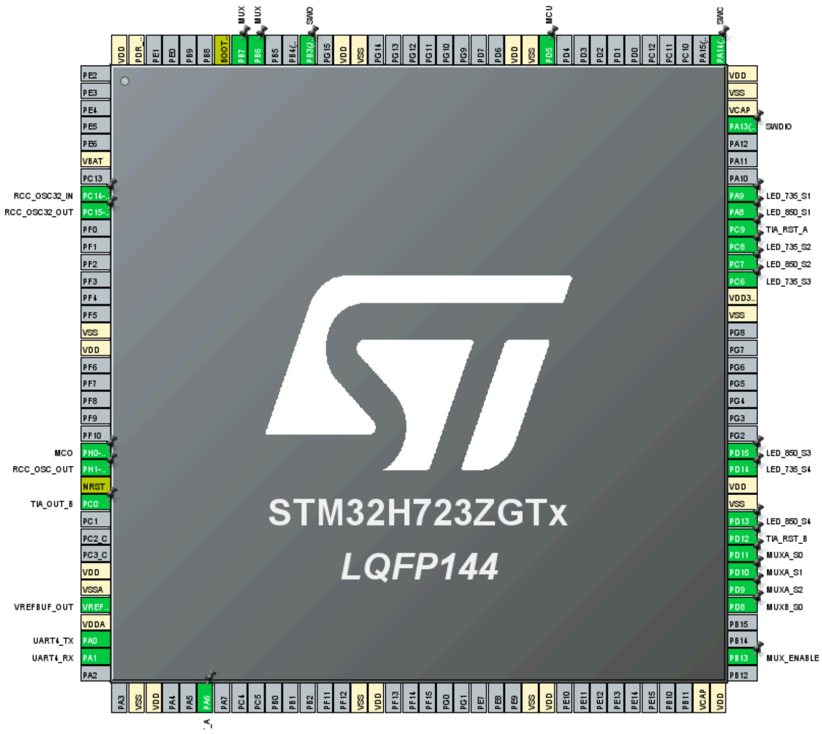

Figure 2. STM32H723 peripheral configuration.

3.1.3 Capacity

- The system can generate 100 samples per second of fNIRS readings.
- The Python script can collect samples from the 40 channels and store them in a CSV file for at least three hours.

3.1.4 Availability

- The system should run without maintenance between 6 months to 1 year.

3.1.5 Latency

- The system must provide 100 continuous fNIRS readings every second.
- The real-time data visualization software must keep up with the acquired 100 Hz readings.
- Any software action must occur within a 1% deviation to ensure that actions are occurring at the specified time. This importance is due to the real-time nature of the requirements.

3.1.6 Manageability/Maintainability

- The software is modular and scalable to ensure any future changes can be implemented easily without severely affecting the functionality of other nearby functions.
- All software is documented on GitHub for storage and version control.

- All hardware must be modular to ensure design scalability and versatility depending on the user's desired use case.

3.1.7 Monitoring

- The hardware must have indicator lights to display ON/OFF, Error states, and Flashing.

3.1.8 Maintenance

Hardware Maintenance

- The hardware must be modular in such a way that it is possible to replace the source-detector assembly if damaged.
- The hardware must support flashing and debugging of the main processor.

Software Maintenance

- The firmware must be updated accordingly if the STM32 HAL libraries it uses ever change.
- Git is used as a version control tool to ensure that the software is not only documented, but easily revised.

3.1.9 Systems Interfaces

- ADC to Source-Detector Assembly
- MCU to ADC
- MCU to TTL-USB Converter (UART)
- TTL-USB Converter to Host PC

3.2 System Requirements Matrix

Req#	Function	Requirement	Date Rewd	SME /Faculty Reviewed / Approved
1	Software	Extract all analog readings at an appropriate time to	3/17/2024	Dr. Ashwin Parthasarathy

		meet the 100Hz scan rate.		
2	Software	Number of error transmissions is <5% of the total transmissions.	3/17/2024	Dr. Ashwin Parthasarathy
3	Software	Software tasks will NOT be starved during runtime.	3/17/2024	Dr. Ashwin Parthasarathy
4	Software	Software must be modular by following common coding conventions.	3/17/2024	Dr. Ashwin Parthasarathy
5	Hardware	The PCB must be flexible to wrap around the head.	3/17/2024	Dr. Ashwin Parthasarathy
6	Software	The MCU can communicate with an external PC.	3/17/2024	Dr. Ashwin Parthasarathy
7	Software	The software functions must perform their intended purpose without interrupting critical tasks such as ADC sampling or UART communication.	3/17/2024	Dr. Ashwin Parthasarathy

4. User Scenarios/Use Cases

Measuring the health of the human brain allows healthcare workers to provide patients with more accurate diagnoses. The primary method for measuring brain activity is through a test known as the Electroencephalogram (EEG). The EEG directly reads the brain signals through the charge discharged by neurons in the brain. However, EEG may miss crucial information such as blood flow and tissue health which can be measured using fNIRS. To best gain a better understanding of the brain, it would be beneficial to perform both fNIRS and EEG scans to gain a full-picture understanding of the patient's medical state.

Additionally, Brain-Computer Interfaces are becoming much more prevalent in the form of implants such as the recently tested Neuralink, or EEG alternatives such as those from OpenBCI. However, unlike these alternatives, an fNIRS BCI is capable of collecting brain activity without surgical implantation and can gather readings with much higher spatial resolution as opposed to EEG. These specific benefits allow for fNIRS BCI to measure brain activity from specific areas of the brain at a high resolution without surgical implantation.

5. Analysis Models

5.1 Sequence Diagrams

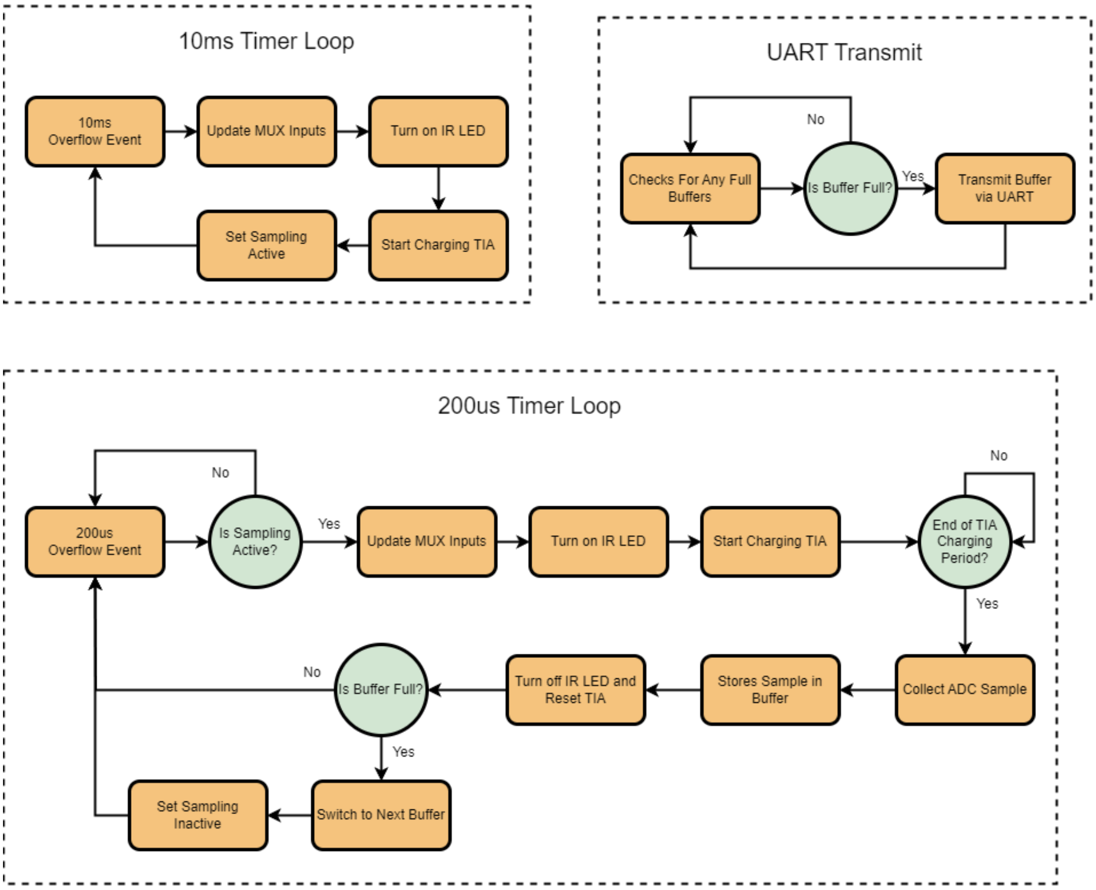

Figure 3. Sequence diagram demonstrating the flow of the control software.

5.2 Data Flow Diagrams

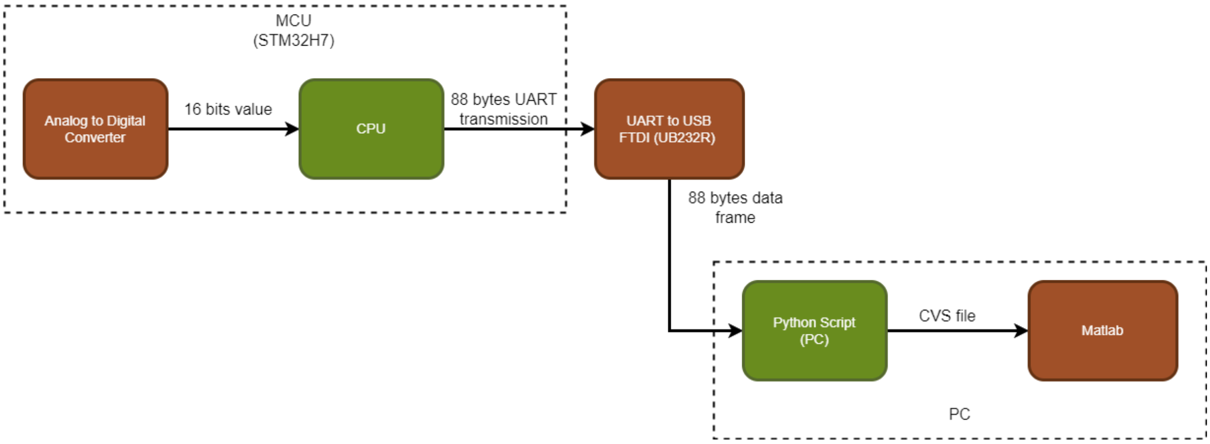

Figure 4. Data flow diagram showing the different hardware components communicating with each other and the different forms of data they exchange.

5.3 State-transition Diagrams

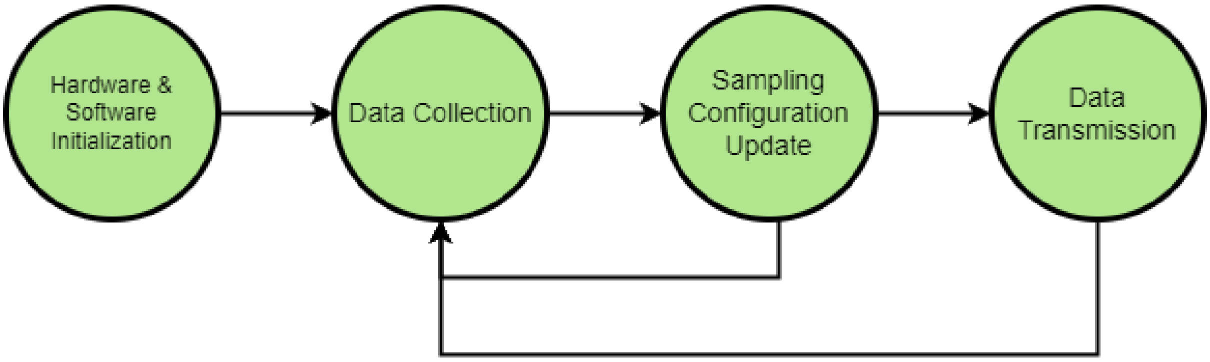

Figure 5. State transition diagram showing the different software stages that the microcontroller goes through.

5.4 SWAP

5.4.1 Size

Headset: 7 x 1.5 inches

MCU Housing: 1.8 x 1.8 x 1 inches

Source-Detector Spacing: 1 cm SS & 3 cm LS

Maximum Headband Circumference: 28 inches

5.4.2 Weight

Total Weight: 1.4 lbs

5.4.3 Power

Nominal Power Consumption: 550mW

5.5 System Performance

Up to 100 Hz sampling frequency (all 40 channels 100 times/s) possibility to increase sampling frequency at the cost of reducing detector sensitivity.

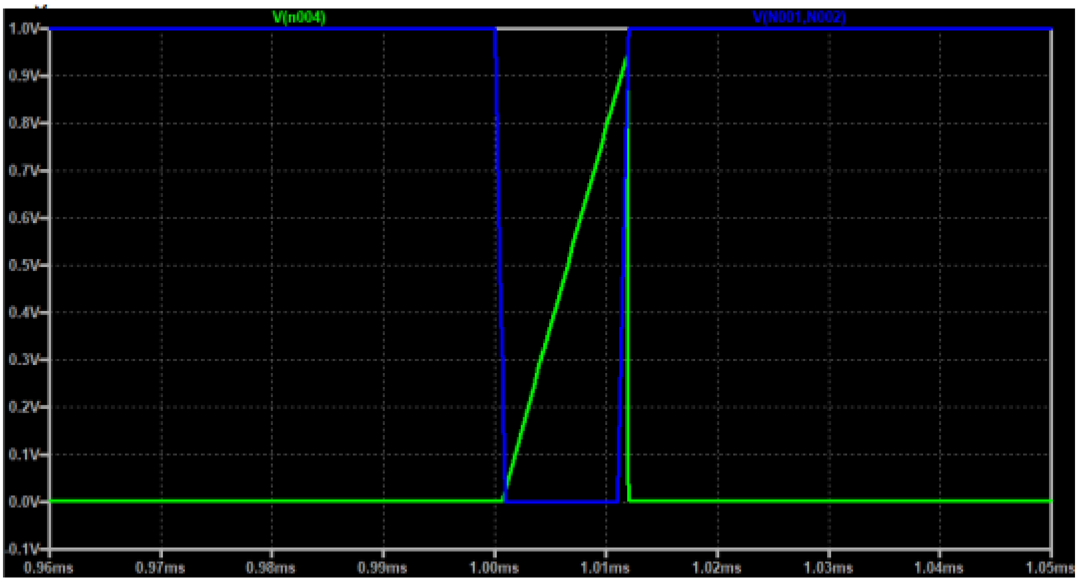

Figure 6. Transimpedance Amplifier (TIA) Integration Time Estimation.

6. Project Risk

The following is a list of risks and solutions if encountered.

- Critical components have long lead times or have become obsolete (eg. Dual-Wavelength IR LED)
    - Find possible drop-in replacements or re-design while ensuring equivalent performance.
- Data sent to the host PC shows significant noise
    - Readjust and tighten the headset to ensure sources and detectors are contacting the skin.
- Electrical discharge to the wearer.
    - Design a quick disconnect and fault detection to eliminate harm to the user.

7. Standards

UART Standards: https://ieeexplore.ieee.org/document/9227663

8. Engineering Ethical Responsibility

The fNIRS BCI project adheres to engineering ethics and prioritizes safety, environmental sustainability, and long-term reliability throughout all stages of design and testing.

8.1 Public Health

The system is designed to ensure no adverse effects on the user's health by integrating safety measures such as voltage and current protection. This approach minimizes risks related to electrical malfunctions or improper operation, ensuring the well-being of users. No leaded solder has been used to manufacture the design.

8.2 Safety

The product ensures safety to the user through current protection diodes, LED power limiting, protective mechanical casing of the microcontroller board, as well as foam placed against the flexible PCB and the skin. The head strap is also easily removable. The electronics contacting the user's skin are also manufactured with gold immersion to ensure biocompatibility.

8.3 Global, Social & Cultural Factors

The design aims to be inclusive and accessible across diverse user groups. By focusing on safe and reliable components, the system is adaptable to a wide range of applications and cultural contexts, ensuring usability and acceptance on a global scale.

8.4 Economic Factors

The system uses USB-bus power, which is cost-efficient and widely available, reducing the financial barrier for users. High-quality components were selected to maximize reliability and longevity, reducing long-term maintenance costs and improving economic viability.

8.5 Environmental Factors

Efforts have been made to minimize environmental impact through low power consumption and the use of easily disposable electronics. The system's reliance on USB-bus power reduces energy waste, while durable components extend its service life, decreasing electronic waste and associated emissions.

9. Conclusions

Overall, the project implemented a modular, open-source, low-cost fNIRS headset allowing users to utilize and adjust the functionality of the designed Hardware and Software MCU platform for their specific needs. Additionally, this project allowed us to deeply explore and learn about bioelectronics design, analog circuitry, firmware and embedded systems design, software design, and hardware-software interfacing. The final product allows for the advancement of fNIRS and BCI research by keeping design scalability and modularity as the driving design considerations, allowing users a low barrier of entry to designing experiments utilizing this platform.

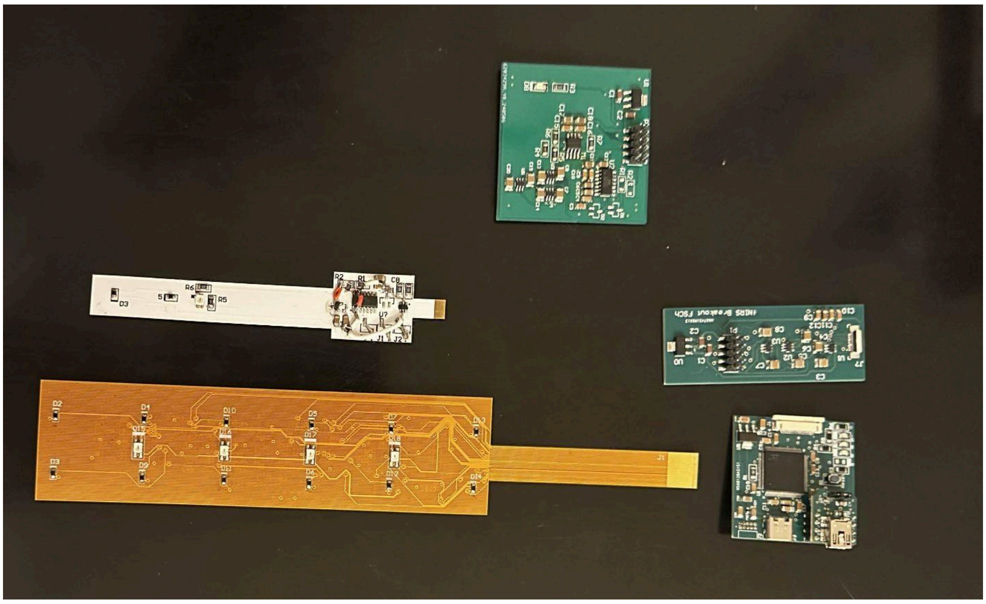

Fig 7: From top to bottom, left/right. The first single-channel prototype breakout board, single-channel flex-PCB, single-channel breakout board, multi-channel board, and MCU board.

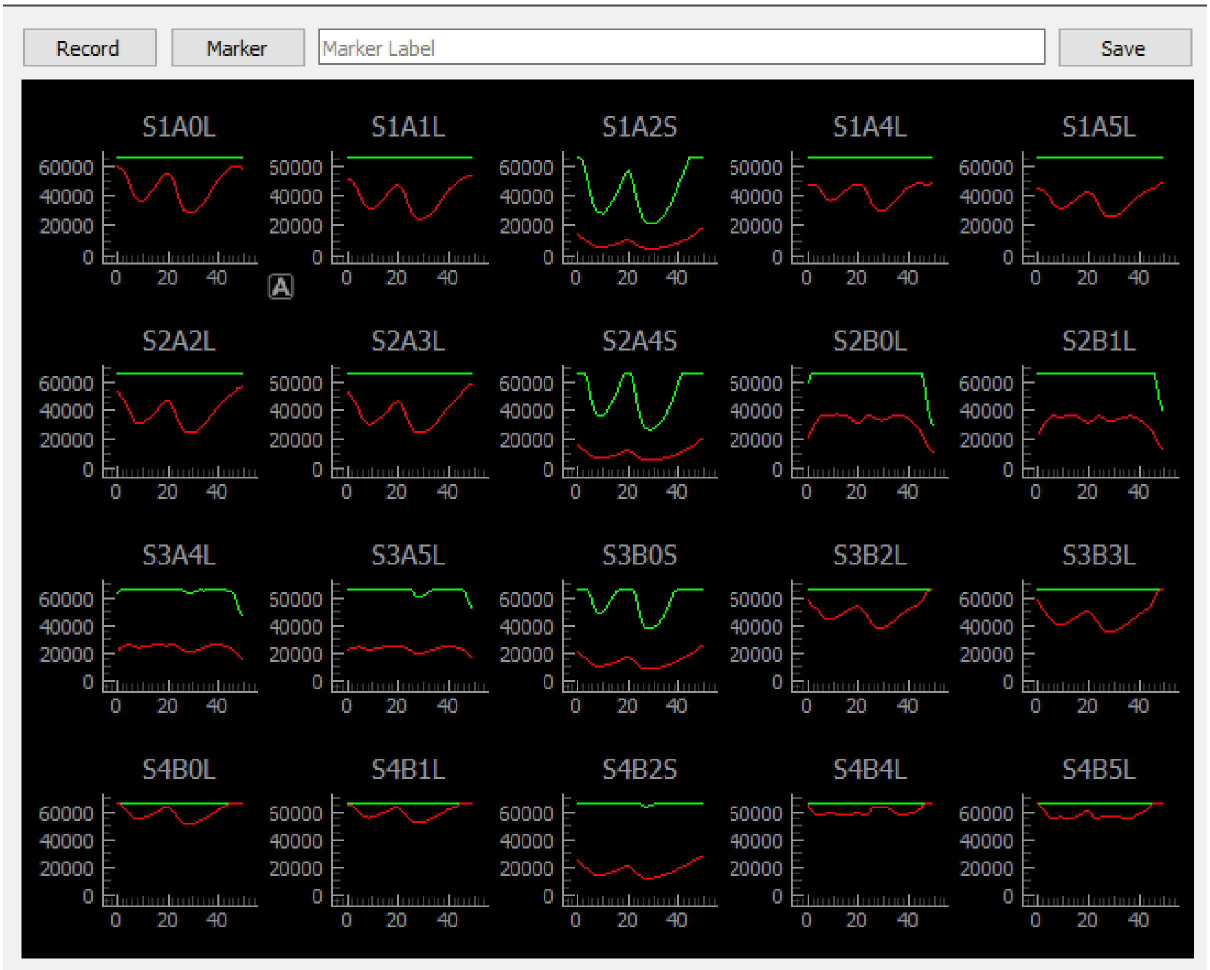

Fig 8: Custom Python-based real-time data visualization program

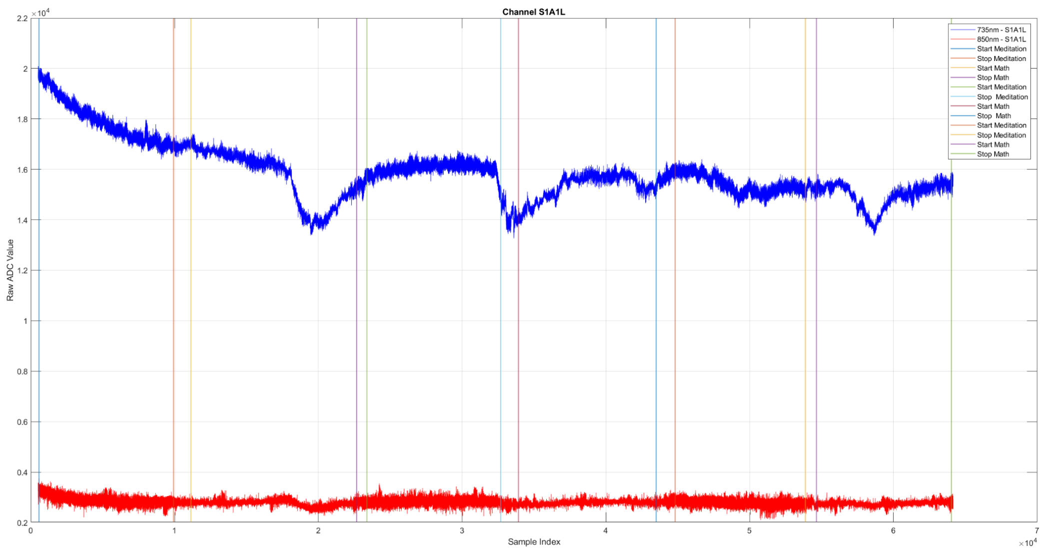

Fig 9. MATLAB data parsing program showing data with real-time markers.

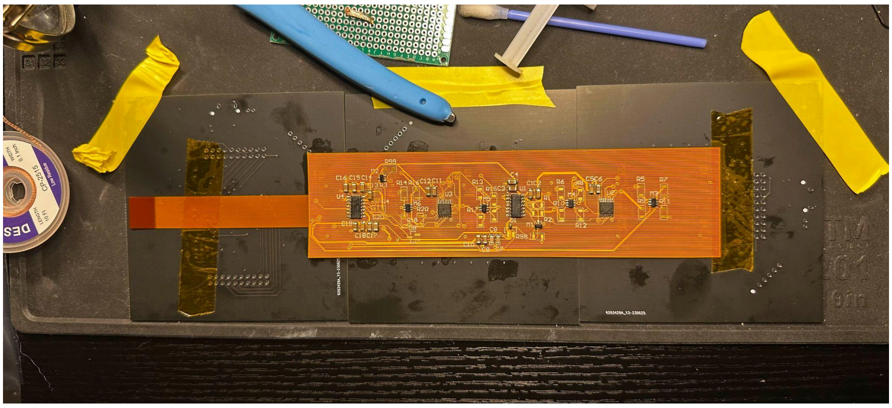

Figure 10. Assembly of multi-channel flex PCB.

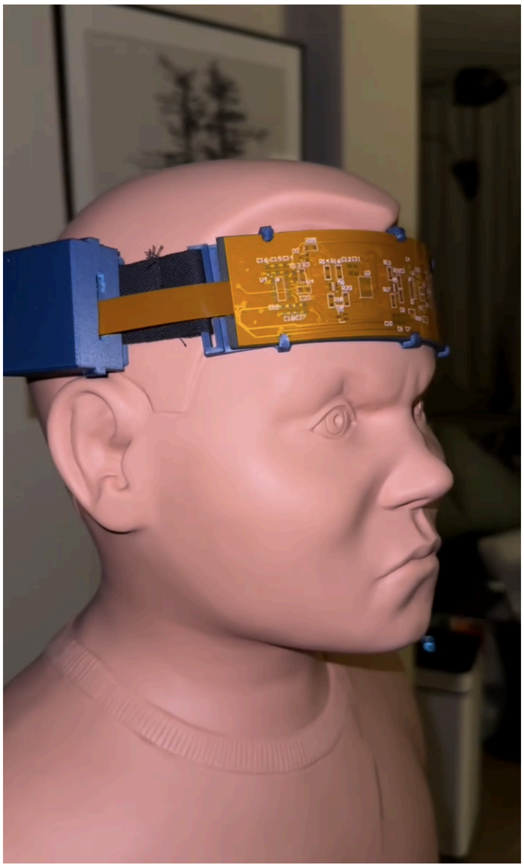

Figure 11. Example fNIRS headset placement.

Figure 12. Final electronic and mechanical design.

10. Work Division

Everyone contributed to both the software and the hardware design and debugging of the project in various ways.

Hardware Lead: Tristan Valenzuela
Firmware Lead: Thang Pham
Data Software Lead: Shovan Shakya