fNIRS Based Brain-Computer Interface Headset

System Requirements Document (SRD)

Version 1.1

April 24, 2024

Team Member 1 Name: _____Tristan Valenzuela_____

Signature:_____  **_____, date: ____04/24/2024_____**

Team Member 2 Name: _____Thang Pham_____

Signature:_____  **_____, date: _04/24/2024____**

Team Member 3 Name: _____Shovan Shakya_____

Signature:_____SS_____ , date: _____04/24/2024_____

Faculty Advisor Name: _____Ashwin Parthasarathy_____

Signature:_____ , date: _____

Table of Contents

1. Executive Summary

1.1 Project Overview

A Brain-Computer Interface (BCI) is a computer interface which translates signals from the brain to control an external device such as a computer. Brain signals can be detected in several ways, one of which is known as Functional Near-Infrared Spectroscopy (fNIRS). The purpose of this project is to implement a Brain-Computer Interface utilizing Functional Near-Infrared Spectroscopy for a multi-purpose system which can be used to analyze brain behavior or control external systems. This project is in partnership with the USF Translational Optics Imaging and Spectroscopy Lab.

1.2 Purpose and Scope of this System Requirements Document

In 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.

Out of Scope

- A comprehensive explanation of the operation of Brain-Computer Interfaces (BCIs) and functional Near-Infrared Spectroscopy (fNIRS).
- A detailed description of applications or use cases for the headset.
- Specific software interface descriptions.

1.3 Abbreviations

BCI: Brain-Computer Interface
fNIRS: functional Near-Infrared Spectroscopy
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
FPGA: Field Programmable Gate Arrays

1.4 References

https://www.usb.org/document-library/usb-20-specification

2. Product/Service Description

2.1 Product Context

Brain-Computer Interfaces are a growing method for user-interface of computer-controlled systems for various applications such as robotics and adaptive game controllers. 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, there are several alternatives which 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.

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 which 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. The LED is placed on the skin while a photodiode which is sensitive to near-IR wavelength is placed in close proximity to the source LED. This set up 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 BCI system utilizing fNIRS-acquired brain signals translates a user's neuronal activity into actions or controls that are utilized by the external system. Typically, the translation process requires a training step which trains the user to consistently activate the same areas of the brain for each specific control or action. This training process can be done through a feedback-based approach which shows the user the brain signal being measured. The user can then, over time, learn to activate specific areas of the brain through mental tasks such as counting or imagery to control the BCI system.

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.
- Due to a lack of available skillset and development time, an MCU is used instead of an FPGA due to most team members being more knowledgeable with MCU than FPGA development.
- Fundings will be sufficient to cover all material costs, including parts meant for prototyping (such as development boards).

Software Assumptions

- FreeRTOS will continue to be under MIT license, free to use and distribute with acknowledgement. RTOS implemented in software will use FreeRTOS's libraries.
- The software will be developed on the STM32 ARM platform.
- STM32 CubeIDE will be available for software development, compilation, flashing, and debugging.
- An external PC is available to run a separate Python program for GUI and computation.

2.3 Constraints

Hardware Constraints

- The fNIRS device is responsible for capturing a complete scan of the brain at a frequency of 100Hz. This means 10ms is split between obtaining the data and sending the data to a PC for processing. It is important to choose a high-speed communication method to leave as much time as possible to obtain the data. This remaining time is further split between each LED source (as there are five measurements for each LED source, of which there are four). With all this in consideration, the microcontroller that is responsible for obtaining the data must operate in microseconds. Therefore, it is important to pick an MCU which has a sufficiently quick enough processor clock to meet this timing constraint.
- The onboard ADC's conversion time must be adequately high to meet the tight timing constraints.
- There must be sufficient flash and RAM memory to harbor the implementation of RTOS.
- The product must be light, portable, and fit to wear on one's head.
- The hardware must support software debugging features such as JTAG or SWD.

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 in a timely manner to get an accurate 100Hz reading. To ensure this, an RTOS will be implemented.
- 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. This also means utilizing only what is needed from the FreeRTOS libraries, discarding the rest.
- Since the product will be powered by a battery, the software must be architected in such a way to conserve power. This means interfacing the MCU with only the necessary peripherals and offloading as much functionality as possible to an external PC.

2.4 Dependencies

- The hardware module must be built before the software can be completely developed and tested. This dependency can be alleviated through the usage of a development board, but not completely avoided.
- The fNIRS headset must be used within an isolated environment with minimal noise.
- The fNIRS headset requires an external computer to interface with via USB to output data and offload computation.
- The Python program used for computing and displaying data to the user will exist on an external PC as a separate program.

3. Requirements

3.1 Hardware Requirements

- The fNIRS headset is able to 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 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.

3.2 Software Requirements

- The product's software will be under the MIT license, free to distribute due to the open-source nature of the project.
- The communication between the MCU and the external PC should allow for real-time processing and applications.

3.3 User Interface Requirements

- There is a GUI generated by a Python program displaying the readings obtained by the fNIRS device. The Python program exists on an external PC separate from the fNIRS device. The GUI will display data in real-time, with minimal lag behind the MCU.

3.4 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 system must have a minimum battery life of 4 hours.
- The headset must be under 0.5 lbs.
- The system must have a communication interface between the host computer and main processor.

Software Requirements

- The software must successfully extract all analog readings in an appropriate time to meet the 100Hz scan rate. The software must not miss or skip any analog readings within a scan cycle. The exact time available for the MCU to extract the ADC readings is TBD and is unknown until the specific ADC and communication interface speed is known.
- The communication interface implemented 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 software/RTOS is architected so that the software tasks will NOT be starved during runtime. All software tasks must be executed within a pre-defined worst-case time limit. Failure to do so will yield a critical error and alert the user.
- The software architecture and peripheral interfacing must be optimized for power conservation in such a way that the onboard LiPO battery is able to last and keep the device operational for at least 4 hours.

3.5 Capacity

- The system must be able to generate 100 samples per second of fNIRS readings.

3.6 Availability

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

3.7 Latency

- The system must provide 100 continuous fNIRS readings every second.

3.8 Manageability/Maintainability

- The software is modular, so that 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.

3.9 Monitoring

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

3.10 Software Requirements

- The software must be modular by following common coding conventions to facilitate ease of modification. The software must also be unit-testable and pass all unit tests developed for it.

3.11 Maintenance

Hardware Requirements

- 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.

3.12 Systems Interfaces

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

3.13 System and hardware Interface/Integration

- FreeRTOS
- Python Scripts (pyserial)

4. System Requirements Matrix

Table 1 Requirements

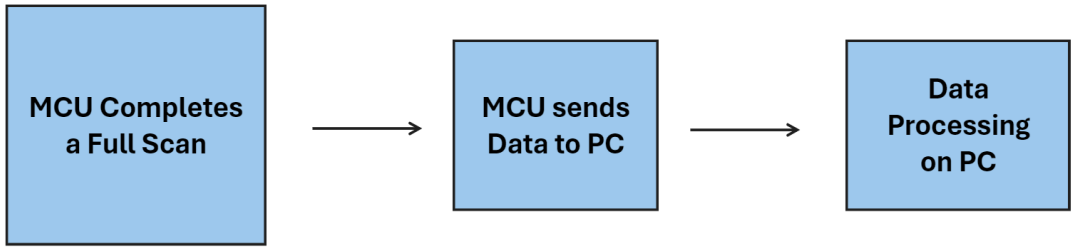
Req#	Function	Requirement	Comments	Date Rewd	SME /Faculty Reviewed / Approved
1	Software	Extract all analog readings in an appropriate time to meet the 100Hz scan rate.		3/17/2024	Dr. Ashwin
2	Software	Number of error transmissions is <5% of the total transmissions.		3/17/2024	Dr. Ashwin
3	Software	Software tasks will NOT be starved during runtime.		3/17/2024	Dr. Ashwin
4	Software	Software must be modular by following common coding conventions.		3/17/2024	Dr. Ashwin
5	Hardware	Performance is optimized so that battery lasts minimum of 4 hours.		3/17/2024	Dr. Ashwin
6	Software	The MCU can communicate with an external PC.	This can be accomplished by either UART to USB or SPI to USB.	3/17/2024	Dr. Ashwin
7	Software	The software functions must perform their intended purpose.		3/17/2024	Dr. Ashwin
8	Design	The fNIRS device is able to give a complete scan at 100Hz.		3/17/2024	Dr. Ashwin
9	Hardware	The PCB must be designed to minimize noise and interference.		3/17/2024	Dr. Ashwin

5. User Scenarios/Use Cases

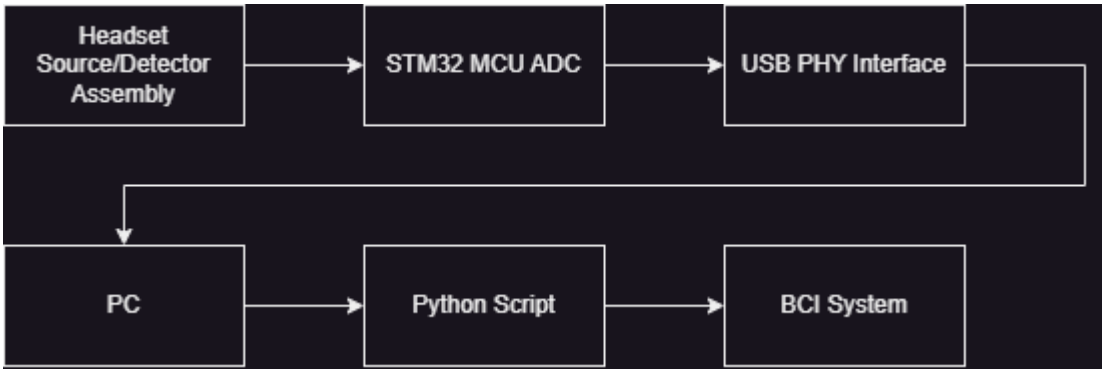
- Due to its safety, affordability, portability, and high temporal resolution, fNIRS holds the promise of widespread adoption as a research and clinical tool. Furthermore, it is especially well-suited for populations and measurement procedures where other imaging methods face limitations. It can be readily applied in infants, children, and agitated patients, as well as procedures involving mobility and interactivity, and in settings like the operating room or intensive care unit.
- Primary utilization domains encompass the study of behavioral and cognitive development in infants and children, examination of psychiatric conditions such as depression and schizophrenia, investigation of epilepsy, and assessment of stroke and brain injury.
- An essential intervention involves enabling communication for individuals with motor disorders like ALS and LIS using fNIRS-based BCI for patients to communicate with “yes” or “no” responses.
- Another important application of fNIRS - BCI is to restore movement in people with motor disabilities, which can be accomplished by controlling robotic arms/hands and wheelchairs.
- fNIRS can be used to monitor neuroergonomics parameters such as mental workload and estimating mental conditions in real time.

6. Analysis Models

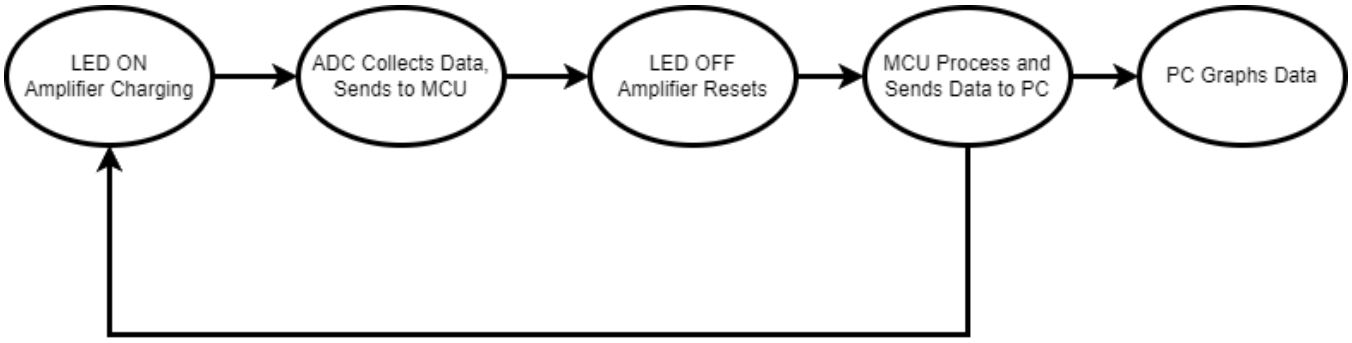
6.1 Sequence Diagrams



6.2 4.3 Data Flow Diagrams



6.3 4.2 State-Transition Diagrams



6.4 SWaP

6.4.1 Size

Headset: 5.5 x 1.5 inches
Source-Detector Spacing: 1 cm SS & 3 cm LS

6.4.2 Weight

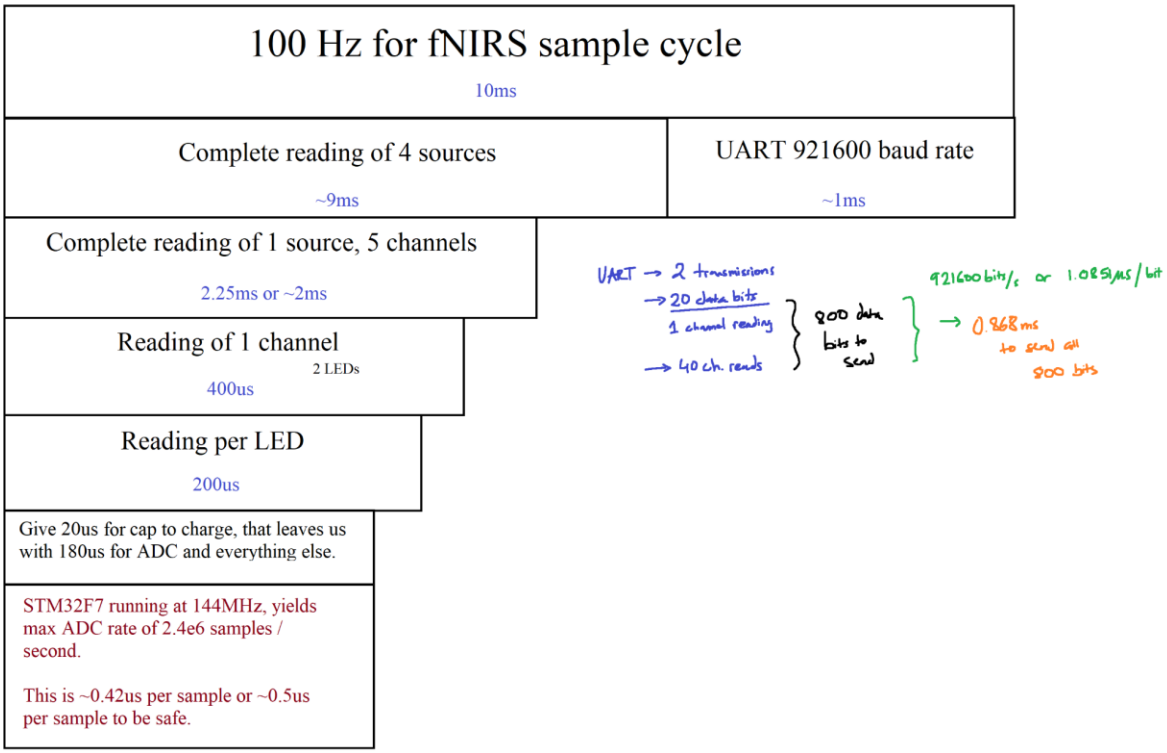
0.25 lbs total
Detector PCB: 0.06 lbs
MCU PCB: 0.04
Headset 0.15 lbs

6.4.3 Power/Performance

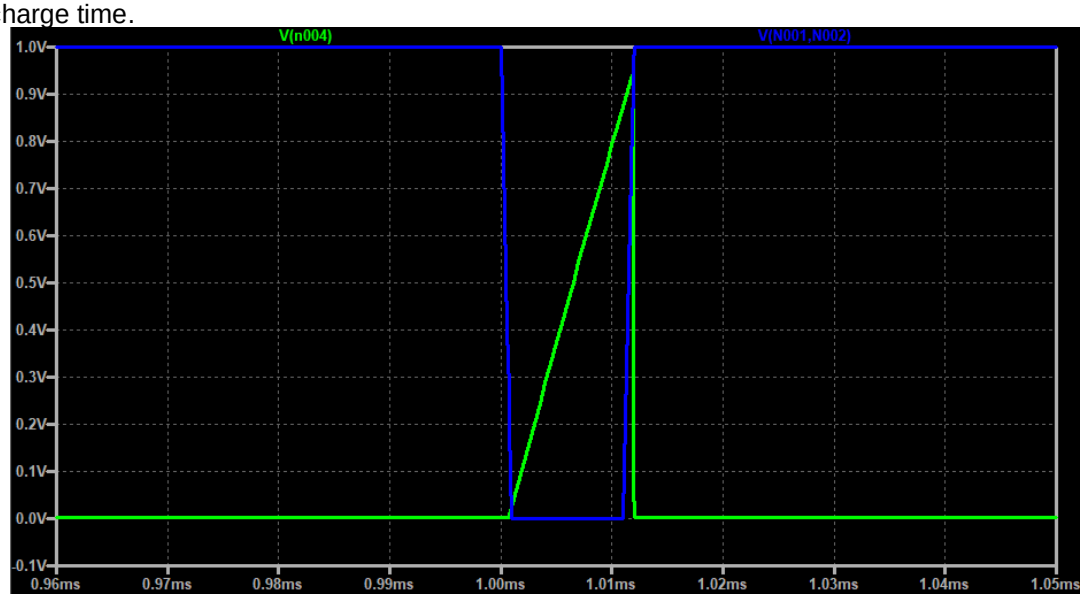
220mW IR LED (Non-continuous)
350mW MCU
60mW Amplifier
630mW Maximum
75 - 100 Samples/s
1 ms Latency

6.5 System Performance

Performed timing analysis of system using worst-case scenario values.



Integration time of TIA (Transimpedance amplifier) estimated through simulation to be 10us using worst-case capacitor charge time.



7. Requirements Test Matrix

Table 2 Test Requirements Matrix

Req#	Function	Requirement	Test Method	Brief Test description	SME /Faculty Reviewed / Approved
1	Software	Extract all analog readings in an appropriate time to meet the 100Hz scan rate.	Software Error Exception	Once an analog reading is present, a timer will count down. Failure to obtain the reading before the timer reset will result in an error thrown in debug console.	Dr. Ashwin
2	Software	Number of error transmissions is <5% of the total transmissions.	Comparison of Dummy Data	A predefined set of data will be transmitted from the MCU. The program on the external PC will receive these values and report the number of values that do not match up with the predefined set of values.	Dr. Ashwin
3	Software	Software tasks will NOT be starved during runtime.	Software Error Exception	All tasks will be given a max waiting time. Failure to execute before the max waiting time will throw an error in the debug console.	Dr. Ashwin
4	Software	Software must be modular by following common coding conventions.	Analysis	The software will be compared to a set of good coding conventions and is peer reviewed to ensure that the convention standards are met.	Dr. Ashwin
5	Hardware	Performance is optimized so that battery lasts minimum of 4 hours.	Hardware Test	The device will be left to run for 4 hours.	Dr. Ashwin
6	Software	The MCU can communicate with an external PC.	Analysis	The MCU will transmit some predetermined dummy data. The external PC will receive it and compare the accuracy of the received data to the dummy data.	Dr. Ashwin
7	Software	The software functions must perform their intended purpose.	Unit Test	A set of unit tests will be used to isolate each function and ensure that they are performing as expected.	Dr. Ashwin
8	Hardware	The PCB must be designed to minimize noise and interference.	Analysis	The PCB layout must be analyzed to ensure best practices before manufacturing.	

8. 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 host PC shows significant noise
    - Readjust and tighten 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.

9. Standards

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

RTOS Standards
https://standards.ieee.org/ieee/2050/7178/

10. Engineering Ethical Responsibility

The fNIRS BCI project shall follow engineering ethics and safety through all stages of design and testing. During the design process, great significance will be placed on designing the system to adhere to safe practices. The design will include safety indicators and modes to shut down in case of electrical short circuits to prevent harm to the user. Additionally, the headset will incorporate an easily removable head strap to be used in case of emergency.

This system will have little to no impact on the environment as emphasis will be made during the design process to ensure low power consumption and easily disposable electronics. The system will be designed to run on USB-bus power which requires low power consumption through all modes of use. Additionally, high quality and high reliability components will be selected to ensure a long service life and reduce waste and emission.

The data gathered during all research studies will remain confidential and adhere to the rules and regulations of the institutional review board (IRB).

11. Change Management Process

We are documenting all our ideas, design choices, progress, responsibilities, and tasks on a cloud-based platform called Atlassian. Within Atlassian, there exists Confluence and Jira. Confluence is where we document our research and design ideas. Jira is where we delegate tasks and document responsibilities. Therefore, any change to any ideas, design choices, and such is made by updating the articles in Confluence. Any change in tasks will be made in Jira by updating the story's description.