Functional Near-Infrared Spectroscopy (fNIRS) Headset

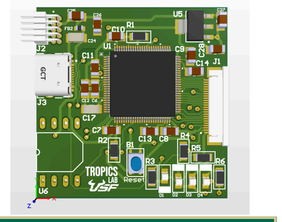
A Modular fNIRS Platform

Shovan Shakya, Tristan Valenzuela, Thang Pham
Advisor: Dr. Ashwin Parthasarathy
EEL4914: EE Senior Design 2 Fall 2024
Department of Electrical Engineering

Website

Problem Definition

Functional-Near Infrared Spectroscopy (fNIRS) is an emerging biomedical optics technology used for analyzing brain function and cognitive activity by using light. The current methods involve utilizing high-cost, and proprietary technology to perform research using fNIRS. Our project aims to solve this hurdle by creating a low-cost, open-source platform for future fNIRS researchers.



Motivation

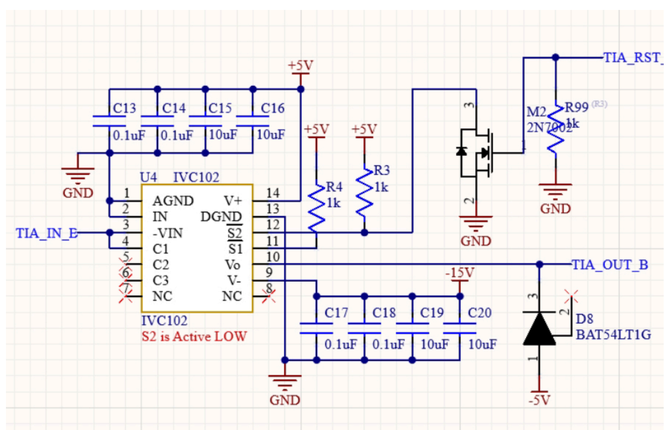
We wish to develop a **low-cost** and **open-source** fNIRS headset to establish a platform and lower the cost of entry for research. We aim to make the design **plug-and-play**, with modularity in mind to foster customizability.

Proposed Design

To handle the varying source and detector locations, the design uses a **multiplexed architecture** and a **unique source-detector geometry** to meet the design requirements and ensure modularity.

Design Considerations / Methodology

To account for varying source-detector separations, the system requires **adjustable gain**. This is largely accomplished by choosing a **switched-integrator TIA** which has a gain proportional to a defined integration time.



Requirements & Specifications

Hardware

- The headset must have **20 source-detector** channels with **735nm** and **850nm** source LEDs.
- The headset must have both **long (3cm)** and **short (1cm)** source-detector separations for each source.
- The system must communicate with a host PC and support a constant data stream at a baudrate of **230400 bit/s**.

Firmware

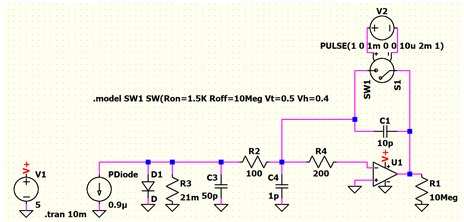
- The firmware is **real-time**. It must execute code by a specified time deadline.
- The program must collect **ADC samples** and transmit them to the PC via **UART** within 10ms.

User Interface

- The **Python** program must receive the incoming data and **display it graphically** in real time. It must have a UI where the user can put **timestamps**. It must export the data as a **CSV file**.
- The **MATLAB** program must take the CSV file and perform **post processing** on it, resulting in a graph of oxygenated and deoxygenated concentrations.

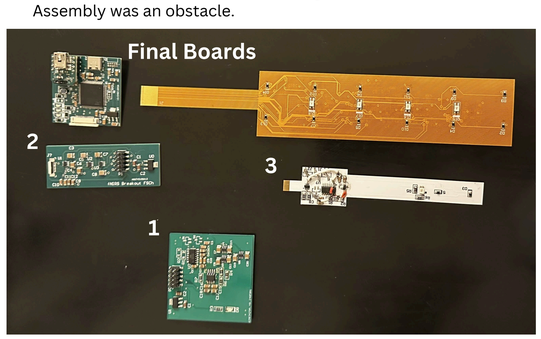
Simulation Results

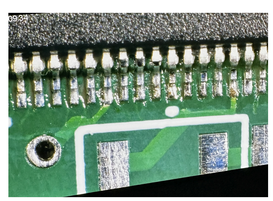
Major circuit design changes, such as introducing a **multiplexer** to multiplex different **photodiode channels** into a **single TIA**, were simulated before implementation to observe effect on accuracy of photodiode readings

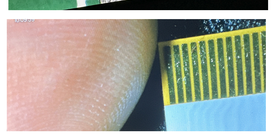


Prototyping

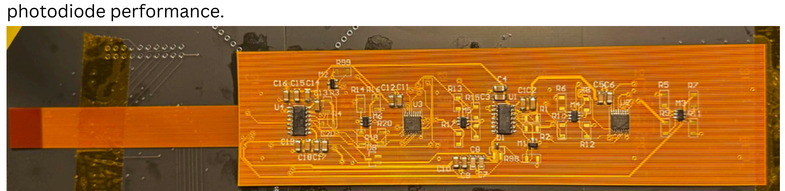
Three prototype boards were designed to **isolate** and **test** various applications of **theory**. Assembly was an obstacle.







1) **Single-Channel fNIRS board:** tests and select appropriate photodiode
2) **Single-Channel breakout board**: interfaces with off-the-shelf STM32 MCU, tests timing and transimpedance amplifier (TIA) selection.
3) **Single-Channel flexible PCB:** Test capabilities and limitations of flexible PCB, and selected photodiode performance.



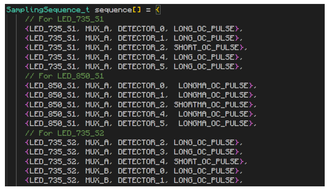
Testing

Hardware

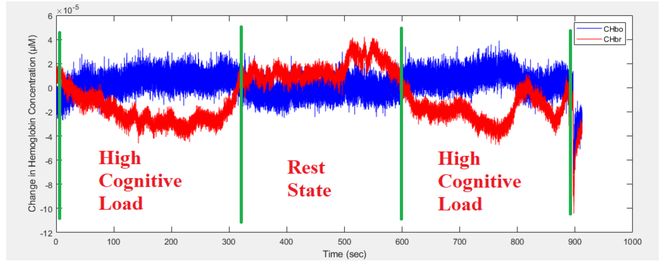
- Functional testing using **Oscilloscope** and **Logic Analyzer** to **debug** Hardware-Software interfacing

Firmware

- Software debugging was done using a Segger JLink EDU and **STMCubeIDE**.
- Microcontroller **GPIO timings** were verified using Sigilent Logic Analyzer.

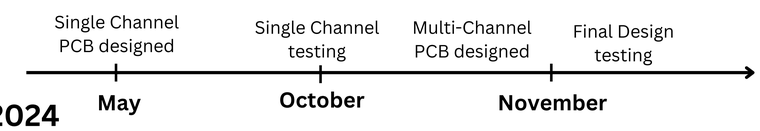


Analysis



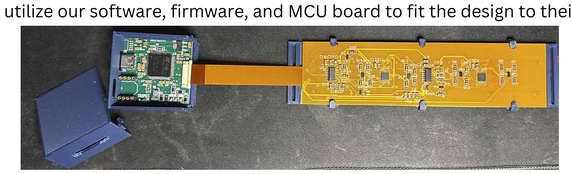
Gathered brain signal data while user was under different levels of cognitive load

Timeline



Conclusion

The system operates within the desired requirements and is capable of gathering data for fNIRS research purposes. The USF TROPICS lab and future fNIRS researchers can use this design as a plug-and-play method for testing. Alternatively, researchers can create custom layouts and utilize our software, firmware, and MCU board to fit the design to their needs.



Work Divison

Tristan Valenzuela: Hardware Lead | Full Hardware Design
Shovan Shakya: Software Lead | User Program & Flex PCB Design
Thang Pham: Firmware Lead | Interfaced with STM32