

Personal Data Acquisition Prototype

Team 16

ECE 442

Team: Preston Hang, Mingqian Xu, Qunyi Pan

Project Partner: Chris Patton

Section 1

1.1 Executive Summary

With an ever-expanding technological age, individuals can find their hobbies and activities intersecting with technology. One of the most important functions that users may desire include the ability to collect numerical data. This idea of a Data Acquisition system is present today; however, not very accessible for the average person. Currently, there is a gap between DIY and professional devices of this nature that we are trying to fill.

We are going to design a personal data acquisition device that has several different functions related to position and motion data. Our products will be biased towards those who like outdoor sports and sports data collection, with the main direction being for mobile data collection for hiking and cycling. The purpose of this project is to provide a reliable mid level device that non-technical people can use.

The device is tailored to be a wearable device that an individual can use to record live data from the sensor. The project incorporates an LSM9DS1 sensor, an STM32 Development board, a Raspberry Pi 4 Model B, a CANBus data communication protocol, and a Flask app user interface.

In the project [1], Our products will be biased towards those who like outdoor sports and sports data collection, with the main direction being for mobile data collection for hiking and cycling. Our goals include:

- An affordable product
- A application (on phone or laptop) in which recorded data can be live streamed
- A small, durable, wearable

For the current progress of the project, we have completed the basic project planning and divided the product into eight components. The group members started to research and analyze their respective components.

1.2 Team Contacts and Protocols

TABLE I
TEAM MEMBER INFORMATION

Name	Email	Work preference	Availability
Dakotah Rivers	riversda@oregonstate.edu	Hardware/software interface	Mondays: after 6 Tuesday: after 6 Wednesday: after 6 Thursday: after 6
Preston Hang	hangp@oregonstate.edu	Hardware and software programming/integration and UI	M: After 6 T: After 6 W: After 7 TH: After 6/7
Mingqian Xu	xuming@oregonstate.edu	More hardware and include part of software	Monday: after 6 Tuesday: after 2 Wednesday: after 6 Thursday: after 3 Friday: after 2
Qunyi Pan	panq@oregonstate.edu	Hardware	Monday - Friday: after 6

Times that work for all of us: Tuesday and Thursday after 6pm, so there are two group meetings on these days per week. On Tuesday at 6 pm, the group meeting is in discord or zoom. On Thursday at 5:30 pm, the group meeting is in Dearborn or the library.

Project partner: During the communication with the project partner, it was understood that he wanted to get some test data results from this project and could add some of the project content as inspiration to his current work. The project partner was more inclined to understand how to make the sensor translator work more effectively and to collect test data through different sensors. During the development of the project, the project partner can provide some help in compiling the Raspberry Pi, as he has previous experience working with it. This was a great help to our group. We have a video conference with our project partner every Tuesday at 6pm on discord or zoom. Our group would share with him what we had done in the previous week and then ask questions for discussion and analysis.

TABLE II
TEAM PROTOCOLS AND AGREEMENTS

Topic	Protocol	Standard
-------	----------	----------

Weekly meeting	An online meeting is held every Tuesday to discuss problems encountered and the latest tasks completed	In the share folder of google drive, write questions and what was done last week on the weekly meeting prep sheet in advance.
Contact	Team uses discord to contact each other and post job schedule	Summary job schedule, and identify the degree of task completion. Create a weekly task form in the share folder of google drive, and each person will write a message on it after completing the task.
Record	Put detailed and well-documented notes/work in Google share drive	After the meeting, summarize the meeting report and skim it for each team member. Notes from each meeting will be collected in a special folder for review.

1.3 Gap Analysis

From talking with our project partners, we learned that there is a gap in the market for a small, multifunctional motion data collector. This product is not only for sports data collectors, but also for people who like outdoor sports. This product can help users to know their real-time location and various sports data. Its competitive advantage lies in the small size, easy to carry, a variety of data collection, and real-time transmission of information to the mobile terminal.

A reliable consumer level device that allows non technical people who are interested in the performance of their activities to analyze statistics and data of their movements. This device is biased toward use in hiking and cycling, and can help users who want to collect human movement data or who like to travel outdoors by collecting movement information in real time and sending it to mobile terminals. The feature of collecting movement information can help the relevant users to understand their own movement condition when hiking or cycling, and to analyze and adjust it. We wanted to bring together various sensors, such as GPS and accelerometers, and use them to achieve multi-functionality. In addition, for the small size design, we will refer to a variety of small instrument housing frameworks and add our own design ideas. We want our device to be affordable, practical, and easy to use for an everyday user without having to deal with device programming or complicated programs. We want our product to have an easy to use interface for the customer.

1.4 Timeline/Proposed Timeline

Dakotah's preferred role: software/hardware interface

Preston's preferred role: Software implementation + User Interface (website, app)

Mingqian Xu's preferred role : enclosure, hardware, and software.

Qunyi Pan's preferred role: Hardware(sensors), PCB design.

TERM 1 TIMELINE

Smartsheet Tip → A Gantt chart's visual timeline allows you to see details about each task as well as project dependencies.

PROJECT TITLE	Personal Data Acquisition Prototype	TEAM MEMBERS	Preston Hang, Dakotah Rivers, Mingqian Xu, Qunyi Pan
PROJECT PARTNER	Chris Patton	DATE	10/27/22

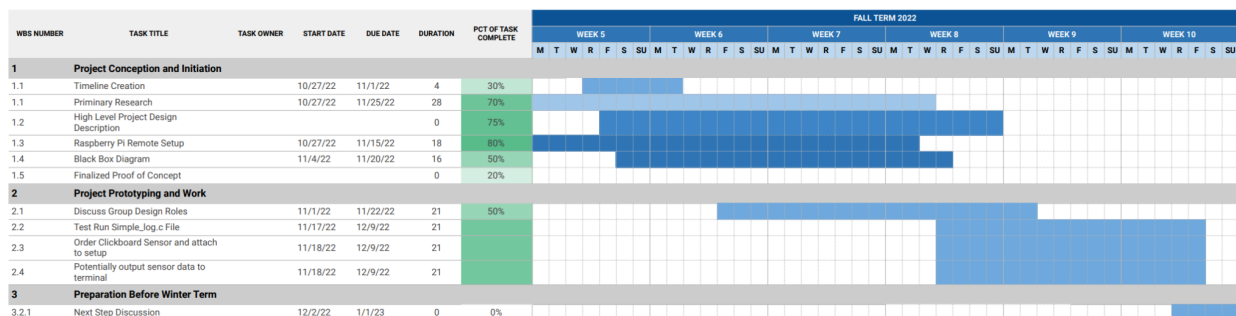


Fig.1. Proposed Term 1 Timeline

1.5 References and File Links

- [1] "Personal Data Acquisition Prototype," *EECS Project Portal*. [Online]. Available: <https://eeecs.oregonstate.edu/capstone/submission/pages/viewSingleProject.php?id=8X8BrEPHpj64KOt1>. [Accessed: 14-Oct-2022].

1.6 Revision Table

11/17/2022	Mingqian Xu: follow the feedback to edit the information of project partner and section 1.1, 1.3.
11/01/2022	Mingqian Xu: follow the feedback to edit references format and create table 2
10/14/2022	Preston: Added touch ups, added cover page

10/11/2022	Preston: Document draft created Team: Section 1 drafts (1.1 - 1.6)
------------	---

Section 2: Impacts and Risks

2.1 Design Impact Statement

2.1.1 - Introduction

With the digital age advancing, the world is finding technology being integrated more and more with our daily lives, and the general public is becoming more interested in personal data collection. Whether that be recording maximum speed when skiing, or monitoring moisture levels and plant health for a garden, data collection is important in today's environment. Our Senior Capstone project will be a Personal Data Acquisition device, designed to monitor a variety of data (primarily movement), including but not limited to positional data, motion, GPS, etc. To prepare for potential impacts that this project can pose, we will be assessing a number of effects our Data Acquisition device can pose to Public Health, Culture and Society, the Environment, and the Economy. This paper will provide our team a deeper understanding of benefits and consequences that may arise from the development and implementation of this technology, and supply other engineers with perspectives that they may not have considered for their own projects.

2.1.2 - Public Health, Safety, and Welfare Impacts

- Mental Wellbeing

Our device could be considered a fitness tracker, as it records movement data for the user to analyze to improve their performance in the given application (snowboarding, skiing, mountain biking, ect). While wearable fitness trackers can have numerous benefits for the individual, for some users it could have negative psychological effects as well.

A recent poll done by Joybird, a fitness organization, asked 1800 people who had fitness trackers several questions. Some results include: "do *you feel guilty about days you do not wear your fitness tracker?*" 51.6% said yes. "*Do you feel guilty about ignoring notifications and prompts to be more physically active?*" 48.5% said yes. "*Do you feel stress from not closing your rings or meeting daily goals?*" 38.5% said yes. These are not small numbers, it turns out that huge portions of the wearable fitness tracker using the population feel stress and anxiety because of their trackers. This could be a potential negative impact for almost half of users of our device [1].

CNN also conducted a survey with 200 women who use Fitbits. Even when it comes to physical exercise, the wearable device showed negative effects. 43% of the survey participants said they felt their workout was “wasted” if they weren’t wearing their Fitbit. Even more extreme, 22% said they felt less motivated in general to exercise [2].

A wearable main purpose is to aid a person’s physical wellbeing and performance, but can also instigate negative impacts on their mental wellbeing. Although our device may not be as frequently used as a fitness tracker, our device’s partnered app can create reminders to take a break when exceeding numerous hours of consecutive use, and (on the other side) not guilt users to use it with notifications. As a data collection service, creating a data “goal to meet” could be an option, but should not be required to emphasize that this isn’t for breaking records and daily goals, but for quantitative data measurements.

2.1.3 - Cultural and Social Impacts

- Privacy

The most important aspect of our project are the sensors to collect the data. Once the data has been collected, we also need to organize and store the data into a server so that the app can livestream it. With our project handling data collection and storage, we must carefully consider user privacy and data security. Nowadays, security is one of the most important features a device can have, as many sensors (such as a GPS or a microphone) grab personal information. It is a huge responsibility for companies and developers to protect these users’ privacy. And in fact, there are some scandals or bad impacts caused by the violation of privacy. One of them is the Facebook privacy scandal. Facebook’s corporate parent has reached a tentative settlement in a lawsuit alleging the world’s largest social network service allowed millions of its users’ personal information to be fed to Cambridge Analytica, a firm that supported Donald Trump’s victorious presidential campaign in 2016 [3].

- Security

Another infamous scandal is Yahoo’s hack in 2013, where 3 billion user’s account information were grabbed by a hacking group. The hackers took the names, dates of birth, telephone numbers and passwords of clients whose security could be handily broken. The gatecrashers additionally acquired security questions used to reset lost passwords and reinforcement email addresses, important data for somebody attempting to break into different records claimed by a similar client, and particularly helpful for programmers attempting to break into government PCs all over the planet [4].

2.1.4 - Environmental Impacts

- Electronic Waste

The environmental impact of our products is mainly related to how we handle waste. Since our products use a large number of electronic devices, the definition provided in the "BAN protects people and the planet from the toxic components within electronic waste" [5] is that e-waste is very broad and includes small electronic appliances. So when our products are discarded, they are considered e-waste. According to G. Gaidajis' research [6], e-waste is chemically and physically different from municipal or industrial waste because it contains hazardous and valuable materials. If people do not handle e-waste properly, they can have a negative impact on the environment, such as contaminating the soil with metals. Among our

products, all of them are made up of electronic devices except the casing. If the products are not recycled correctly, then they also have an impact on the environment. The solution we see is for companies to take the initiative to recycle, similar to the iPhone recycling. Discarded products recovered from users can likewise be sold to specialized e-waste recycling companies. Although this will raise the cost, it is a more environmentally friendly way.

According to an article published by sustainability CO-OP [7], when electronic products are manufactured, they tend to produce a lot of carbon dioxide as an accessory and consume a lot of water, which is a great pressure on areas where water is scarce. The pressure is on. Our products, too, are made up of electronics. They often face corresponding problems in the manufacturing process. Our solution is to use and promote electronic products that are manufactured with a national license and have third-party testing and certification. This can inhibit to some extent the irregular production by unscrupulous businessmen. In general we take into account two kinds of environmental impacts, one from waste and the other from the manufacturing process.

2.1.5 - Economic Factors

- Accessible Middle Platform

Looking through an economic perspective, there doesn't exist a middle ground in data recording systems between expensive professional systems, and small, single-functioning systems. For data acquisition in the industry, Tektronix, a large company that specializes in electrical engineering equipment, provides a few options on systems. Their Keithley series of data acquisition systems includes the DAQ6510 Logging and the 2700 Multimeter System. These devices can provide data acquisition for "environmental monitoring of DUT temperature...Burn-in testing, HALT and HASS accelerated life testing, [and] Failure analysis... [with a quantity of] 576 channels." [8], but cost upwards of \$3000. On the other side of the spectrum, smaller data recorders, such as a speedometer for a bike, can be just \$30 on Amazon, yet only provide data collection for a single function. Currently, we are aiming at a \$300 device, providing a platform for individuals seeking a system with various functions that won't cost them a fortune. With this, the market for a higher functioning data collection device would allow accessibility for people of lower socioeconomic status. There do exist a few middle-cost wearables, such as a \$200-\$300 Apple Watch; however, a majority of the marketplace doesn't emphasize the "data analysis" function that we are aiming for. The project will implement an application that live streams user's recorded data for digestible analysis, and allow users to do more with what they've collected.

- Hidden Costs

However, some smaller data acquisition devices can have "unseen" costs for hardware and software. In the article "The Hidden Costs of 'Low-cost' Data Acquisition Systems," Fluke Calibration goes over how many different systems can plague customers with additional costs for hardware and software. They mention that devices are usually evaluated with a "cost per channel" which can give the illusion that a system may be a good deal; however, higher end systems with 20 channels can multiply the cost unfairly, since a customer may not even need all 20 channels. Another hardware type are "Plug-and-play" DIY data recorders, which allow a user to connect established modules to fit their needs, but these require additional hardware purchases to make work. On the other hand, consumers may not realize that they have to purchase the packaged software, which can range from "a few hundred dollars to several thousand dollars and still not address all of your specific needs" [9]. A device like this, if manufactured and developed professionally, could negatively impact those of lower income, ambushing them with hidden costs and accumulate to an outrageous price. Our solution to this

is to include all that needed to function from the get go. This results in a usable product without additional hardware and a fully developed software implementation. Firmware and GUI updates could be provided to improve the project, similar to a driver update, but shouldn't be needed if we program all required features.

2.1.6 - Conclusion

As more and more technology is developed, it is crucial for engineers to remember the technology's impact on the world around us. Although there may be beneficial impacts, example being the not understanding all of the potential byproducts can lead to unintended consequences, such as exploitation of the technology. Our Personal Data Acquisition device has many implications, including but not limited to:

- Stress and/or anxiety from the wearable
- Invasion of privacy
- Hacking of personal information
- Electronic waste
- Excessive resource consumption
- Potential hidden costs

After analyzing these issues, our group has come to realize just how significant these and other impacts can be. Some can be solved, such as being transparent with prices and not including hidden costs, but other times, impacts can't always be resolved by the engineer. As a team, we have to provide our best efforts in diminishing the negative impacts of our project. Although our device is only a small part of many electronic products, it is often the combination of small parts that can have a big impact.

2.1.7 - References

- [1] "Canvas - a blog by Joybird," *Joybird*. [Online]. Available: <https://joybird.com/blog/fitness-tracker-health-impact/>. [Accessed: 03-Nov-2022].
- [2] R. Duus, M. Cooray, and T. Conversation, "Research reveals the dark side of wearable fitness trackers," *CNN*, 01-Sep-2016. [Online]. Available: <https://www.cnn.com/2016/09/01/health/dark-side-of-fitness-trackers>. [Accessed: 03-Nov-2022].
- [3] "Facebook parent settles suit in Cambridge Analytica scandal," *CNBC*, 27-Aug-2022. [Online]. Available: <https://www.cnn.com/2022/08/27/facebook-parent-settles-suit-in-cambridge-analytica-scandal.html>. [Accessed: 03-Nov-2022].
- [4] "Facebook parent settles suit in Cambridge Analytica scandal," *CNBC*, 27-Aug-2022. [Online]. Available: <https://www.cnn.com/2022/08/27/facebook-parent-settles-suit-in-cambridge-analytica-scandal.html>. [Accessed: 03-Nov-2022].

- [5] “BAN protects people and the planet from the toxic components within electronic waste,” *Basel Action Network*. [Online]. Available: <https://www.ban.org/e-stewardship>. [Accessed: 03-Nov-2022].
- [6] G. Gaidajis*, K. Angelakoglou and D. Aktsoğlu. (2010, Oct 9) “E-waste: Environmental Problems and Current Management,” [review article]. Available: <https://pdfs.semanticscholar.org/0316/b2c1985fb04c47e8e9d0dcdb40198e547dd0.pdf>. [Accessed: 03-Nov-2022].
- [7] “Are Electronics Bad for the Environment?” *sustainability CO-OP*, March 24 2022. [online]. Available: <https://thesustainabilitycooperative.net/2022/03/24/are-electronics-bad-for-the-environment/#:~:text=Many%20toxic%20substances%20are%20used,can%20also%20harm%20human%20health>. [Accessed: 03-Nov-2022].
- [8] “Keithley switching and Data Acquisition Systems,” *Tektronix*. [Online]. Available: <https://www.tek.com/en/products/keithley/switching-and-data-acquisition-systems>. [Accessed: 04-Nov-2022].
- [9] “The Hidden Costs of ‘Low Cost’ Data Acquisition Systems,” *Fluke*. [Online]. Available: <https://us.flukecal.com/literature/articles-and-education/data-acquisition-and-test-equipment/application-notes/hidden-costs>. [Accessed: 04-Nov-2022].

2.2 Risks

TABLE III
RISK ASSESSMENT AND ACTION PLANS

Risk ID	Risk Description	Risk Category	Risk Probability	Risk Impact	Performance indicator	Action plan
R1	Run out of project budget	Financial	M	H	Spending upwards of \$300	Keep a financial ledger and keep track of purchase
R2	Difficulty with required technical skill and knowledge required	Technical	H	M	Slow progress on project	Consult professors, TAs, or project partner
R3	Safety concerns with enclosure design. Could hurt individual wearing device	Personal/Health	M	H	Unsafe or unsecure enclosure	Rigorous field testing of enclosure

R4	Accidentally break the PCB, Microcontroller.	Technical/Financial	L	H	The PCB, microcontroller cannot work anymore	Design or buy a new one
R5	Sensor translator does not properly communicate signal to EtherCAT device	Technical	M	H	Sensor information does not output	Ensure PCB is created without electrical error
R6	Easily damagable enclosure that is prone to harsh weather conditions	Technical	M	H	Any water/dirt/snow seepage, and damage occurring during reliability tests	Redesign enclosure
R7	PCB/Parts delay due to shipping, etc	Timeline	L	H	Expected shipping gets delayed	Order parts early/allocate additional time in timeline for delay
R8	EtherCAT redesigned PCB does not work	Timeline/Financial/Technical	M	H	No communication between Raspberry Pi and redesigned PCB	Ensure no electrical errors when designing, confirm with possible breadboard variation
R9	UI does not output correct data	Technical	L	M	Data from sensor is either not outputting to UI or data is incorrect/corrupt	Data analysis and software debugging

2.3 External Links

1. G. E. T. M. Editor, "Best practices for ensuring safety with electrical & electronic enclosures," *YONGU-Professional Manufactory of Aluminum Enclosures*, 09-Jul-2022. [Online]. Available: <https://www.yg-enclosure.com/article/best-practices-for-ensuring-safety-with-electrical-electronic-enclosures.html>. [Accessed: 03-Nov-2022].

2. A. of us at monday.com, “How to manage your project budget: A step-by-step guide,” *monday.com Blog*, 07-Sep-2022. [Online]. Available: <https://monday.com/blog/project-management/project-budget/>. [Accessed: 03-Nov-2022].

2.4 Revision Table

11/17/2022	Preston: Added 5 additions risks (R5-R9)
11/03/2022	Dakotah: Document draft created Team: Section 2 drafts

Section 3: Top-Level Architecture

3.1 Block Diagram

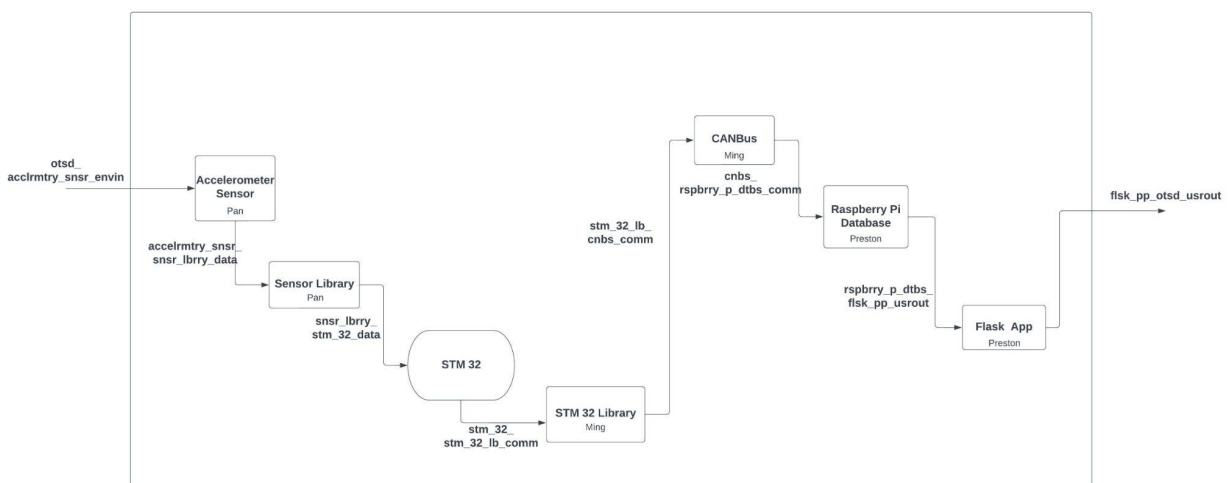


Fig.2. Top Level Block Diagram

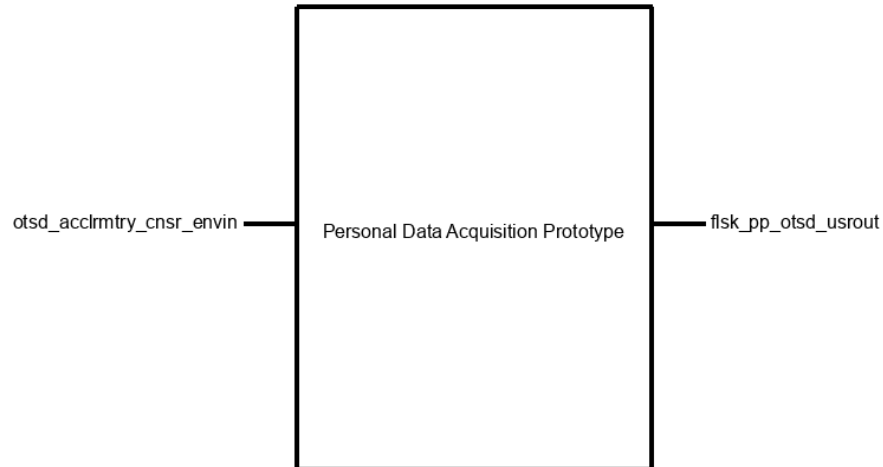


Fig.3. Top Level Black Box Diagram

3.2 Block Descriptions

3.2.1 - Accelerometer Sensor

In this block, we use LSM9DS1 as our accelerometer. This sensor could sense the change of the acceleration, then send the precise data to the STM32 microcontroller by I2C protocol. This block is to understand the hardware of the sensor, how to read the values, and what communication protocol to use.

Block Champion: Qunyi Pan

3.2.2 - Sensor Library

This is a software block between accelerometer and STM32 microcontroller, under this block, it can modulate and transfer the data from accelerometer into a readable format, and make these data be ready for sending to Raspberry Pi Database through CANBus. This block is to program the sensor to be readable by the microcontroller, and involves coding.

Block Champion: Qunyi Pan

3.2.3 - STM 32

This block is the heart of our project, but is not technically a block of work created by one individual system. The STM32-F103C8T6 Blue Pill is the microcontroller that takes in the sensor input and distributes it to the Raspberry Pi Database. For further clarification, this block will not make an appearance in Section 4, as it is only shown here to provide context to the libraries being installed.

3.2.4 - STM 32 Library

The block is designed to prepare the preparations needed to connect the Canbus module. The connection protocol for this block to connect to Canbus is CAN. The goal of this block is to configure and program the microcontroller to enable CAN communication, and send our data using the CAN pins of the microcontroller.

Block Champion: Mingqian Xu

3.2.5 - CANBus

The Canbus block is specifically designed to connect the STM 32 microcontroller to the Raspberry Pi and ensure that their data communication is not affected in any way. This goal of this block is to ensure good hardware is used to enable the CAN communication. There are two hardware components, one for the microcontroller, and one for the Raspberry Pi.

Block Champion: Mingqian Xu

3.2.6 - Raspberry Pi Database

The purpose of this block is to provide a location for the data to be stored. The database will be using SQLite, a simple structured query language (SQL) database that is hosted local to the device it is on. To enable this, SQLite3 will be installed onto the Raspberry Pi and the script that pulls data from the CANbus will have the ability to insert data into the SQLite files. The goal of this block is to enable database storage so that our sensor values can be saved.

Block Champion: Preston Hang

3.2.7 - Flask App

The purpose of this block is to incorporate a user interface for our project. The app will be stored locally onto the Raspberry Pi where it can access the database to plot and view the data coming from the sensor. The plan for the user interface is to use Flask, a Python framework for building back-end applications, with a combination of Chart.js, a Javascript library that specializes in data visualization with charts. The goal of this block is to allow an interactable output for the User to have, which displays the data coming from the sensor live.

Block Champion: Preston Hang

3.3 Interface Definitions

TABLE IV
INTERFACE DEFINITIONS

Name	Properties
otsd_acclrmtry_cnsr_envin	<ul style="list-style-type: none">. Other: z: $\pm 2g$ to $\pm 16g$. Other: y: $\pm 2g$ to $\pm 16g$. Other: Acceleration range: x: $\pm 2g$ to $\pm 16g$

acclrmtry_cnsr_snsr_lbrry_data	<ul style="list-style-type: none"> . Messages: XYZ accel data . Protocol: I2C
flsk_pp_otsd_usrout	<ul style="list-style-type: none"> . Other: Clear Chart Button . Other: Start/Stop Button . Type: Web Application . Type: Real time chart displaying data
rsprbrry_p_dtbs_flsk_pp_usrout	<ul style="list-style-type: none"> . Other: Multiple Tables in DB file . Other: SQLite Installed . Other: DB files . Type: Row value: Sensor reading . Type: Row value: Timestamp . Type: Row value: Sensor Name
cnbs_rsprbrry_p_dtbs_comm	<ul style="list-style-type: none"> . Messages: date or words . Protocol: I2C . Vnominal: 3.3V
stm_32_acclrmtry_cnsr_dcpwr	<ul style="list-style-type: none"> . Vmax: 5V . Vnominal: 3.3V
stm_32_stm_32_lb_comm	<ul style="list-style-type: none"> . Datarate: 1s . Messages: data or words . Vnominal: 3.3 V
snsr_lbrry_stm_32_data	<ul style="list-style-type: none"> . Messages: y-accel data . Messages: z-accel data . Messages: x- accel data
stm_32_lb_cnbs_comm	<ul style="list-style-type: none"> . Messages: data or words . Protocol: RS485 . Vnominal: 3.3 V

3.4. References and File Links

3.5 Revision Table

3/12/2023	Preston: Document draft created Team: Section 3 drafts (3.1 - 3.5)
-----------	---

Section 4: Block Validations

4.1 Canbus Block

4.1.1 Description

The purpose of this paper is to describe the role of the Canbus block in the ECE 442 Personal Data Acquisition Prototype project. The full name of the Canbus is "Controller Area Net-work Bus" and will describe how the Canbus block works, what is in the Canbus block, and how to verify that the Canbus block can be added to the system. The full name of Can-bus is "Controller Area Net-work Bus". It is one of the most widely used fieldbuses internationally. The main function of the Canbus block in this system is to handle the communication between the STM 32 microcontroller and the Raspberry Pi. The main reason for its selection is that it is a

serial communication bus with a multi-master approach and the basic design specification requires a high bit rate, high interference immunity and the ability to detect any errors generated.

Our plan is to design the Personal Data Acquisition Prototype project as a product that can be continuously upgraded. Our original design was to complete a multi-functional Personal Data Acquisition Prototype, but for various reasons we modified our idea. According to the latest design, the project now has only one motion sensor, but we are planning to reserve space for the Data Acquisition Prototype to be upgraded from one sensor to five or six sensors, for example (the sensors can be customized according to the customer's needs), which will make the communication between the STM 32 microcontroller and the Raspberry Pi. This will pose a huge challenge to the communication between the STM 32 microcontroller and the Raspberry Pi, so we designed to use Canbus to handle the transmission between the microcontroller and the microprocessor. So in the Personal Data Acquisition Prototype, the Canbus block is specifically designed to connect the STM 32 microcontroller to the Raspberry Pi and ensure that their data communication is not affected in any way.

In the next section, it will be shown how the design of the Canbus block connects the STM 32 microcontroller and the Raspberry Pi together. The reasons for choosing the Canbus block and its benefits will be explained in the next sections. At the end, we will discuss how to verify the Canbus block. An alternative component to the Canbus module, Ether Cat, will also be added to the discussion.

4.1.2 Design

The content of this module is to receive the data signal transmitted from the STM 32 microcontroller using the RS485 Canbus module, and then pass that signal to the Raspberry Pi. According to the figure 1 Black Box for Canbus block and figure 2 Block Diagram below, the Canbus block will be connected to two different blocks. One is the STM 32 microcontroller module that will transfer data through the `from_stm` interface. The other is a Raspberry Pi microprocessor module that will use the `to_pi` interface to send data.

RS485 Canbus Hat is an expansion board with RS485 and CAN communication function developed for Raspberry Pi, with RS485 and CAN communication function. According to the Figure 3 Canbus connection to Raspberry Pi, it is clear that RS485 Canbus Hat is very compatible with Raspberry Pi. And it uses SPI interface CAN controller MCP2515, with Canbus transceiver SN65HVD230 and RS485 transceiver SP3485, the overall size is 65 mm*30 mm, the diameter of the fixing hole is 3.0mm. Figure 4 Canbus Block Schematic is RS485 Canbus Hat's connection circuit diagram.

Two different ways will be used to connect the two connection ports for selection. Firstly, the RS485 Canbus Hat is an extension of the Raspberry Pi, so it will be connected to the SPI port, which is a good fit. The other connection port is for from_stm. Two jumper wires with 2.54 mm core and 20 cm outer length will be used to connect. The advantage of this is that the pins can be connected directly without soldering, and it is very easy to test and can be removed at any time. This type of jumper wire is ideal for connecting various microcontrollers and microprocessors, so I chose to use it as a way to connect the RS485 Canbus Hat to the STM 32 microcontroller.

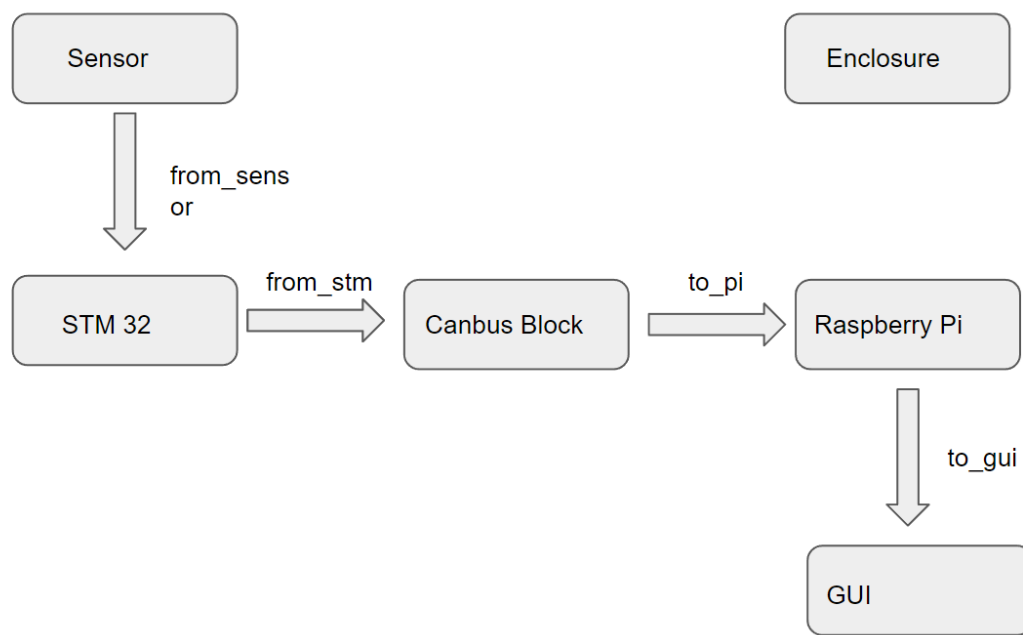


Fig.4. Top Level Block Diagram

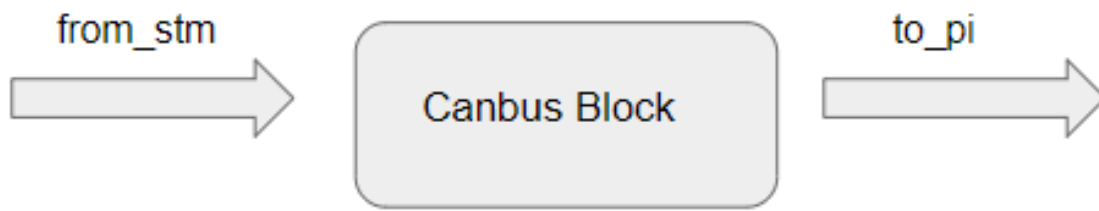


Fig.5. Block Diagram.

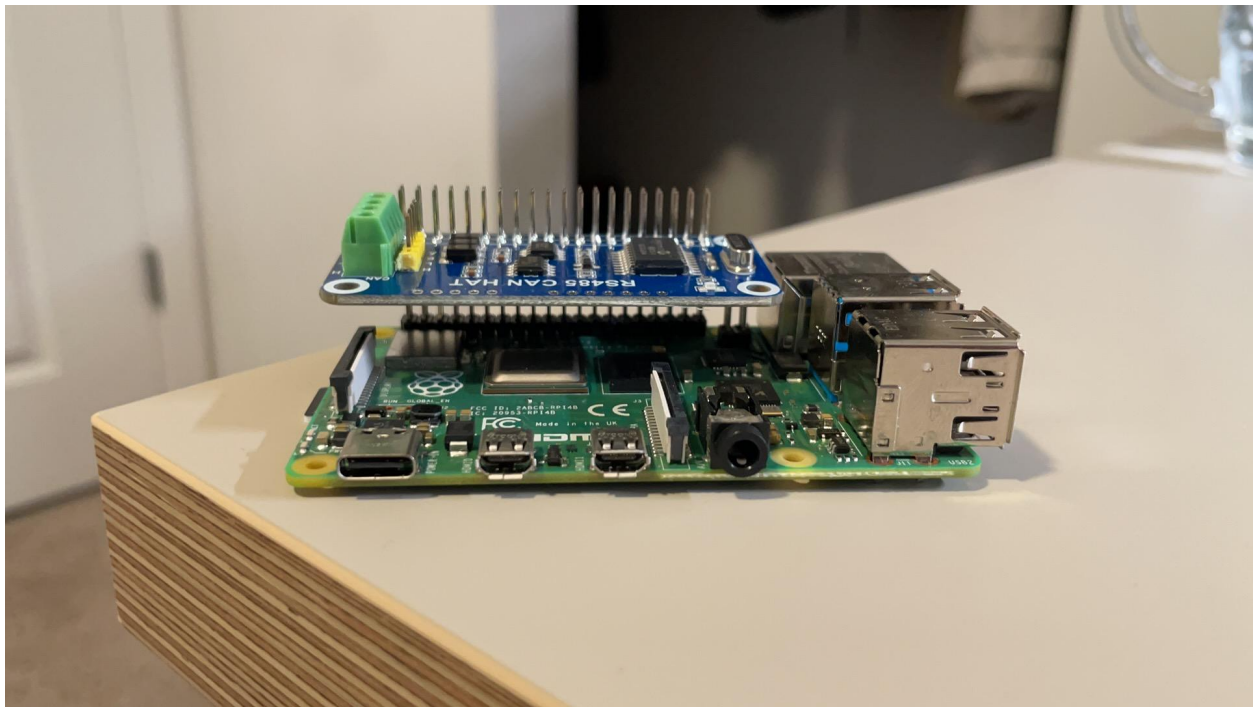
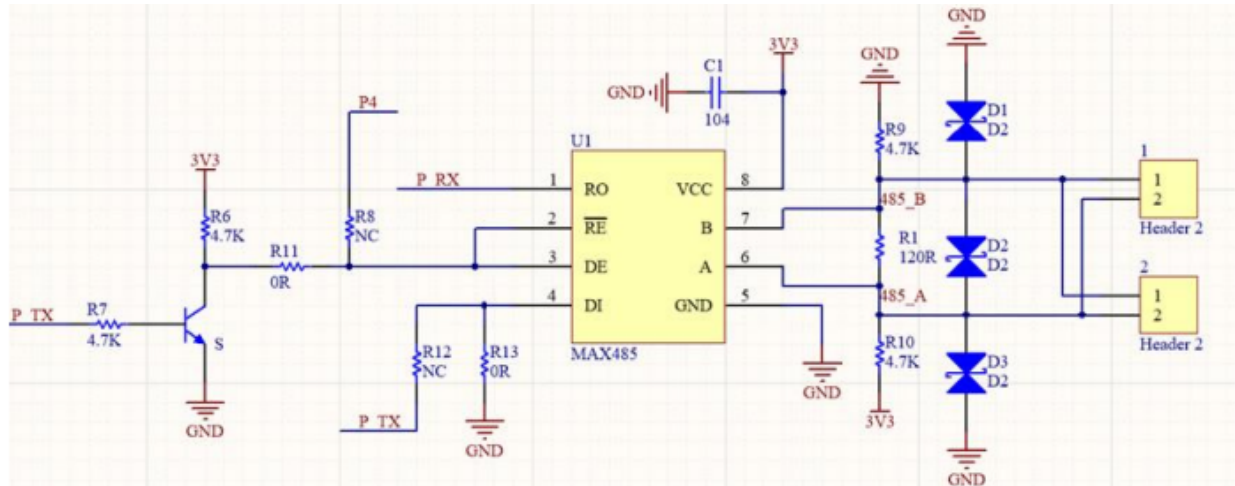


Fig. 6. Canbus connection to Raspberry Pi.



4.1.3 General Validation

I also did some comparison in terms of the cost of the materials chosen, and Canbus has a much better price advantage compared to other communication modules of the same type. For example, if you want to use Ether Cat, the cost will go up a bit. Communicating through a single CAN system, rather than directly over complex analog signal lines, reduces errors, weight, wiring and cost. The low cost gives Canbus an additional advantage in product design. The

Canbus is also a good choice for small modularity, as it is very small, measuring only 65 mm*30 mm overall, which fits well with the design of our Personal Data Acquisition Prototype, as our project is aimed at people who go hiking and free riding in the outdoors. Their needs are often small, light, and multifunctional. Therefore, the small size of the components will be more suitable for our Personal Data Acquisition Prototype. At the same time, Canbus has strong immunity to electrical and electromagnetic interference, making it ideal for applications with strict safety requirements. This also demonstrates that the Canbus is not easily damaged and has a better guarantee in terms of hardware. This is an advantage for people who like to explore the outdoors, as it is very difficult to repair damaged electronics outdoors, and having a sturdy and durable electronic product will add to the security. Another consideration is that the Canbus is relatively easy to program and control, and I can program it in C/C++ or Python. This allows me to complete the project modules with less engineering time.

Next I want to explore alternative blocks to the Canbus block. Our group discussed using Ether Cat as a replacement for Canbus because Ether CAT is the fastest industrial Ethernet technology and it provides nanosecond accurate synchronization. The Ether CAT system architecture typically reduces the CPU load by 25-30% compared to other bus systems with the same set cycle time. The excellent performance of Ether CAT allows system configuration with reduced need for network debugging. Due to the high bandwidth, additional TCP/IP can be transferred simultaneously with the control data. In addition, Ether Cat can support a variety of different interface protocols, giving us more direction in what we can design. If we want to give our Personal Data Acquisition Prototype more room for upgrade, Ether Cat would be a good direction to choose. However, we encountered a problem in the selection of Ether Cat, one is the size of the Ether Cat, and we designed the Personal Data Acquisition Prototype with the intention of making it portable and easy for people to carry around. If you want to buy a small Ether Cat, you will need to spend more money. The second point is that our partner, after understanding the hardware aspect of Ether Cat, understood that an official membership license was required for us to get access to the hardware modifications, but it was difficult to get this membership license. So we switched to Ether Cat as an alternative and chose Canbus as our current choice. This is the case when the carrier is an individual, but if you want to develop a similar product of Personal Data Acquisition Prototype, for example, for the group of scientific data loggers, the test object is a car or an aircraft, and other large equipment. There will be a good alternative to change the communication module to Ether Cat.

These are the reasons why Canbus was chosen as the communication connector. The compactness, affordability, and security of Canbus were among the modules that led to its inclusion in the Personal Data Acquisition Prototype project.

4.1.4 Interface Validation

TABLE V
INTERFACE VALIDATION

from_stm: input

Interface Property	The reason of value	Meet design requirement
I-Nominal: 330mA	This is based on the expected demand for canbus.	RS485 Canbus Hat User Manual [1]. Page 2 and 6. Since the design is rated at 3.3V and the current transfer to the Raspberry Pi needs to be at least 330mA.(Based on the ammeter test results.)
I-peak: 510mA	This may be higher than demand, but meets base expectations.	RS485 Canbus Hat User Manual [1]. Page 2 and 6. Since the design is rated at 3.3V and the current transfer to the Raspberry Pi needs to be at least 51mA.(Based on the ammeter test results.)
Logic-Level: 3.3V	This voltage is determined based on a single power supply for RS485 communication.	RS485 Canbus Hat User Manual [1]. Page 2 and 6. Rated voltage is 3.3V.

to_pi: output

Interface Property	The reason of value	Meet design requirement
I-Nominal: 330mA	This is based on the expected demand for canbus.	RS485 Canbus Hat User Manual [1]. Page 2 and 6. Since the design is rated at 3.3V and the current transfer to the Raspberry Pi needs to be at least 330mA.(Based on the ammeter test results.)
I-peak: 510mA	This may be higher than demand, but meets base expectations.	RS485 Canbus Hat User Manual [1]. Page 2 and 6. Since the design is rated at 3.3V and the current transfer to the Raspberry Pi needs to be at least 510mA.(Based on the ammeter test results.)

Logic-Level: 3.3V	This voltage is determined based on a single power supply for RS485 communication.	RS485 Canbus Hat User Manual [1]. Page 2 and 6. Rated voltage is 3.3V.
-------------------	--	---

4.1.5 Verification Plan

For the from_stm port test:

- 1) Use two core 2.54mm, 20cm outer length patch cables to connect.
- 2) Connecting the STM 32 to the breadboard.
- 3) Connecting the canbus to the computer using USB.
- 4) Use desktop GUI software to demonstrate the canbus interface.
- 5) Enter transmission commands.
- 6) Test, whether the Raspberry Pi receives the signal.

For the to_pi port test:

- 1) Use the SPI port to connect.
- 2) Connecting the Raspberry Pi to the computer using the USB connection cable.
- 3) Use desktop GUI software to demonstrate the canbus interface.
- 4) Enter transmission commands.
- 5) Test, whether the Raspberry Pi receives the signal.

4.1.6 References and File Links

[1] *RS485 Canbus Hat User Manua*. [Online]. Available:

<https://www.waveshare.com/w/upload/2/29/RS485-CAN-HAT-user-manuakl-en.pdf>

4.1.7 Revision Table

02/10/2023	Mingqian Xu: Rewrite all content
01/20/2023	Mingqian Xu: Create initial document

4.2 Accelerometer Sensor Block

4.2.1 Description

The purpose of our project is to design a personal data collection device that is ideal for outdoor sports enthusiasts, such as hikers and bikers. The goal of this project is to create a compact, mobile device that allows for easy and uninterrupted data collection during these activities. The focus is on making the device user-friendly, even for those who are not tech-savvy, and to provide access to the collected data through a companion app.

This document specifically focuses on the accelerometer sensor block, which is a 9DOF device (based on the LSM9DS1 chip). The accelerometer sensor block provides the power supply for the accelerometer, determines the range of acceleration it can sense, and contains the signal that holds the information on changes in position. This document also provides a step-by-step testing process to verify the design and to ensure that the properties of the block are compatible with other blocks in the project. The accelerometer sensor block is a crucial component of the data collection device and plays a vital role in ensuring that the data collected is accurate and relevant.

4.2.2 Design

In this block, the power supply for the sensor is provided through the `otsd_acclrmtry_cnsr_dcpwr` interface with a voltage of 3.3V. This dc voltage is supplied by the microcontroller that the sensor is connected to. The sensor is responsible for receiving environmental inputs, which is force or we also can say that as acceleration, through the `otsd_acclrmtry_cnsr_envin` interface.

Once the sensor receives this information, it processes it by converting it into XYZ position data. This is a crucial step in the data collection process as it allows the sensor to accurately determine the position of the object it is monitoring.

After the sensor has processed the data, it sends the 3 axis positions to the Printed Circuit Board (PCB) that contains the microcontroller through the `acclrmtry_cnsr_pcb_asig` interface. This allows the microcontroller to access the processed data and use it for further analysis and control.

The entire process of power supply, data collection, and data transfer is essential in ensuring that the sensor operates correctly and accurately. The 3.3V power supply provides the necessary energy for the sensor to function, while the environmental inputs allow the sensor to determine the position of the object. Finally, the transfer of the processed data to the PCB allows the microcontroller to use this information for further analysis and control.

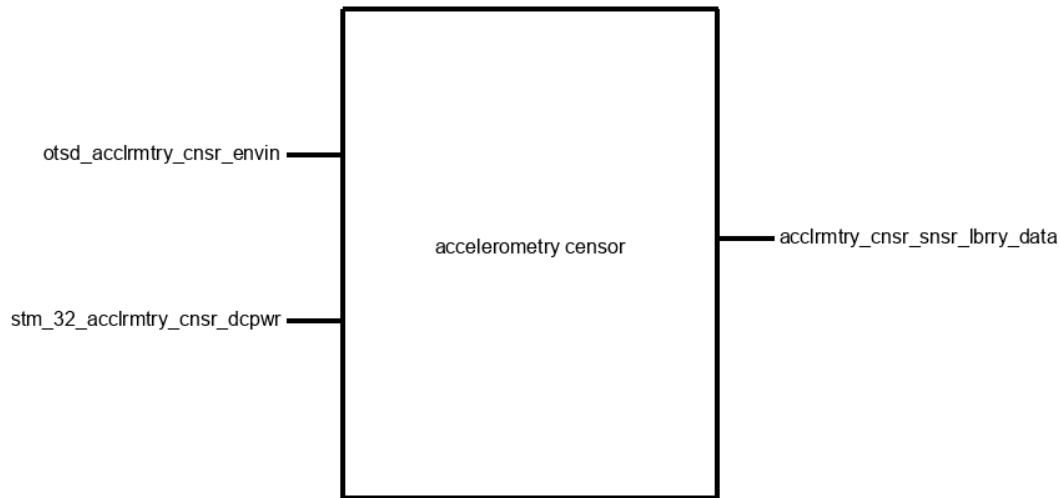


Fig.8. Accelerometer Sensor Block Diagram

4.2.3 General Validation

For the successful completion of our project, we have selected the 9DOF as our sensor of choice, and there are three interfaces that require validation. Firstly, the power supply of the sensor must be verified to ensure that it does not consume an excessive amount of power, which could potentially put undue pressure on the printed circuit board (PCB) being used. Typically, a microcontroller outputs a voltage of either 3.3V or 5V, and it is important to confirm that the 9DOF can meet these power requirements.

Given that the project is focused on outdoor exercise, the sensor must be highly sensitive to detect the motion of the user. Additionally, as the user will be changing altitudes, it is important that the sensor has the capability to detect changes in position along all three axes. To simplify the programming process, the sensor should also be able to send position information to the microcontroller in the form of XYZ coordinates.

This validation document serves to verify that the 9DOF 3 click will meet the requirements of our project by testing the various interfaces and features of the device. By carefully documenting and testing each aspect of the sensor, we can ensure that the 9DOF will provide accurate and reliable readings, making it the ideal choice for our project.

4.2.4 Interface Validation

TABLE VI
INTERFACE VALIDATION

Interface Property	Why is this interface this value?	Why do you know that your design details <u>for this block</u> above meet or exceed each property?
--------------------	-----------------------------------	--

Stm_32_acclrmtry_cnsr_dcpwr: Input

Vmax: 3.3V	The LSM9DS1 datasheet offers the value of the power supply	The voltage consumption is standard for the microcontroller output
------------	--	--

otsd_acclrmtry_cnsr_envin : Input

<p>Other:</p> <p>Acceleration range: ±2g, ±4g, ±8g ±16g</p> <p>Gyroscope range: ±125°/s, ±250°/s, ±500°/s, ±1000°/s, ±2000°/s</p> <p>Magnetometer: 1300μT (x,y), 2500μT (z)</p>	<p>The LSM9DS1 datasheet provides detailed information about the range of accelerometer, gyroscope, and magnetometer sensors included in the device. These sensors allow for the precise measurement of motion and orientation. The accelerometer measures linear acceleration along the three axes, the gyroscope measures angular velocity about the three axes, and the magnetometer measures the strength and direction of</p>	<p>Based on the sensitivity specifications provided in the datasheet, the LSM9DS1 has the capability to accurately detect a wide range of motion and orientation. The sensitivity of the accelerometer, gyroscope, and magnetometer sensors is precisely calibrated to provide high accuracy and resolution in detecting motion. This makes the device ideal for a wide range of applications as wearable devices. The ability to accurately detect</p>
---	--	---

	<p>magnetic fields. The datasheet specifies the exact range of measurement for each of these sensors, allowing for the user to fully understand the capabilities of the device. Additionally, the datasheet also provides information on the device's power requirements and communication interface. This information is crucial for ensuring that the device is properly integrated into a system and operates within its specified parameters. The datasheet is an essential resource for those looking to utilize the full potential of this device in their projects.</p>	<p>motion and orientation is critical for ensuring the device functions as intended and provides the necessary data for further processing and analysis. With its high sensitivity and accuracy, the LSM9DS1 is a powerful tool for those looking to measure and analyze motion in their projects.</p>
--	--	--

acclmtry_cnsr_snsr_lbrry_data : Output

<p>Other: data transfer via SPI and I2C protocols.</p>	<p>As stated in the datasheet, the LSM9DS1 is equipped to communicate with a host device as a slave using two serial digital interface protocols: SPI and I2C. These protocols allow for fast and efficient transfer of data between the sensor and the host device. This makes it possible to easily integrate the device into a wide range of systems and applications.</p> <p>The choice of communication protocol can be selected based on the specific requirements of the system and the application. SPI provides a fast, full-duplex communication link, while I2C allows for communication with</p>	<p>SPI and I2C are commonly used in lots of microcontrollers. Which also matches the protocol that our microcontroller uses.</p>
--	--	--

	multiple slave devices on the same bus. Both protocols offer advantages in different situations, and the LSM9DS1's support for both protocols allows for greater flexibility in selecting the best option for a given project.	
Position coordinate: X,Y and Z	The XYZ axis data can describe every position during movement, and can not only show the horizontal position, but also vertical position.	The microcontroller can use the XYZ coordinate to collect and record live motion data from the sensor with any direction and angle of movement, it totally meets the requirements of the project.

4.2.5 Verification Plan

Step 1 - Power Testing:

In order to properly test the power requirements of the 9DOF sensor, the first step is to connect the device to the microcontroller board. Once the connection has been established, the microcontroller will provide a 3.3V voltage to the sensor. The LED indicator on the 9DOF click serves as a key visual indicator of the device's power status. If the LED indicator is lit and functioning properly, this is a positive sign that the 9DOF sensor is receiving the required power to operate correctly. If the LED indicator is not lit or not functioning properly, it may indicate a power issue and further troubleshooting may be necessary. It's important to ensure that the 9DOF sensor has a proper power supply in order to perform optimally and provide accurate readings.

Step 2 - Data Transfer Testing:

In order to validate the data transfer capabilities of the 9DOF 3 sensor, the next step is to collect and display the XYZ axis position data. This can be achieved by writing a firmware using the STM32 CubeIDE and uploading it to a STM32 microcontroller. The firmware will be responsible for collecting the position data from the 9DOF sensor and displaying it in real-time on the screen.

It is important to verify that the XYZ coordinate data being displayed is reasonable and accurately represents the position of the sensor. If the data being displayed is within an acceptable range, this indicates that the 9DOF 3 sensor is fulfilling the data transfer requirement and transmitting the position information correctly to the microcontroller.

By testing the data transfer capabilities of the 9DOF sensor, we can confirm that the device is able to effectively communicate the position information to the microcontroller. This is a critical step in the validation process, as it ensures that the 9DOF sensor will be able to provide accurate readings and meet the requirements of the project.

Step 3 position change testing

In the third step of the process, we need to alter the position of the sensor to obtain accurate readings. The movements involved in this step are crucial in ensuring the accuracy of the data that is being collected.

The first type of movement that is necessary is the horizontal movement. This movement includes sliding the sensor from left to right, and from forward to backward. This will help us determine the accuracy of the readings in the X and Y directions.

Next, we have the vertical movement which involves moving the sensor upward and downward. This will help us determine the accuracy of the readings in the Z direction.

In addition to these simple movements, we also need to perform some complex movements like circular movements in different directions. This will help us determine the accuracy of the readings in all three dimensions, and ensure that the data collected is representative of the real world.

Once we have completed these movements and have confirmed that the XYZ coordinates are reasonable and consistent with the real world, we can be confident that this block has successfully fulfilled all of the requirements. This is an important step in ensuring the accuracy and reliability of the data being collected by the sensor.

4.2.6 References and File Links

“SparkFun Electronics.” [Online]. Available:

https://cdn.sparkfun.com/assets/learn_tutorials/3/7/3/LSM9DS1_Datasheet.pdf. [Accessed: 13-Mar-2023].

4.2.7 Revision Table

2/10/2023	Qunyi Pan: finish the final draft of the accelerometer sensor block
-----------	---

	validation
1/20/2023	Qunyi Pan: makes a accelerometer sensor block validation draft

4.3 - Flask App

4.3.1- Description

The User Interface block is the end result of the data system. In this block, the UI will extract stored data from a SQL (Structured Query Language) database that is hosted by the Raspberry Pi.

Project Partner requires this partnered UI to be:

- Easily accessible via phone, tablet, or computer
- Present live data from the sensor
- Read data coming from an SQL styled database

From a database, the UI will pull sensor readings and convert them to a chart so that a user can see the data being visualized. The UI should be simple, where it really only needs to display the sensor reading in approximate real-time, indicating to the user that what they're seeing is what is currently happening.

This block contains:

- The User Interface application (accessible via multiple devices)
- The database that the UI will read from
- A plot of the values taken from the database (Present live data coming from SQL database)

4.3.2 - Design

The block should contain two different programs of code: one specifically for the design of the application and handles the displaying of data, and another for the support of drawing in the information from the database.

Before this block occurs, it is assumed that the sensor has created a value, and this is passed into the database. An example of a row entry in the database is formatted as:

TABLE VII
EXAMPLE DATABASE ROW

Sensor Name; The type of sensor that is being read Ex. "Motion" Format - Text	Value; The value that is read by the sensor Ex. "45" Format - Integer	Timestamp The time in which the value was read by the sensor Ex. "2023-02-08 01:24:23.9700" Format - Datetime (YYYY-MM-DD H:M:S.f)
--	--	---

Once data is formatted into the above, this block interprets the database and pulls the data to then plot the values vs. timestamp. This translates to a graph with y-axis as the value (in units depending on the sensor) and an x-axis of time.

There are a few approaches to this UI that can be used:

- **Egui**
 - Egui is a Rust framework that provides an easy method of developing a GUI through the use of widgets [1]. Widgets are viewable/interactable by the user. This designed application will include a plot widget, for displaying the data. Rust is the programming language decided for this, as it is a fast processing language with safe memory management. It is also a language that our project partner has familiarity with. The GUI's input is the processed data, which should be organized in a way so that it can recognize the type of sensor, and it's values. This is important in the event that multiple sensors are modularly added to our system.
- **NodeRED**
 - NodeRED is an open source "flow-based" editor that runs with Node.js [2]. This software is usable in IoT products by reading values and then easily displaying them through a dashboard. Here, two flows can be created: one for reading the database, and one for displaying the data in live time.
- **Flask App**
 - Flask is an easy to use Python framework that allows for easy app development [3]. Using a virtual environment, you are able to deploy a

python app that outputs to a web browser (locally installed, so the browser URL is localhost:5000). Data can be plotted using Matplotlib, a library used for data visualization. The plot can be updated so that new data being written in can shown on the display. It may also be possible to push Flask to the web, so that you are able to access it from outside the local network.

For all approaches, data should be retrieved from an SQLite database. SQLite is a database application that is not located on the cloud, and is used for small data applications [4]. The project partner has stated that this platform should be sufficient for this project. SQLite requires an installation.

The UI's output should be a plot of the data for users to visually see what is being recorded. The UI may need to be hosted or accessible on the web, so that the UI is accessible through any device. Additionally, the UI should be interactable and include some kind of method to possibly download either the database file, or the chart itself. This feature would be implemented last and allow some interactions for the user.

The image below is the Black Box Diagram of the block is shown. There is an input of the Raspberry Pi database to the Block. This is included for a more technical perspective on the block.

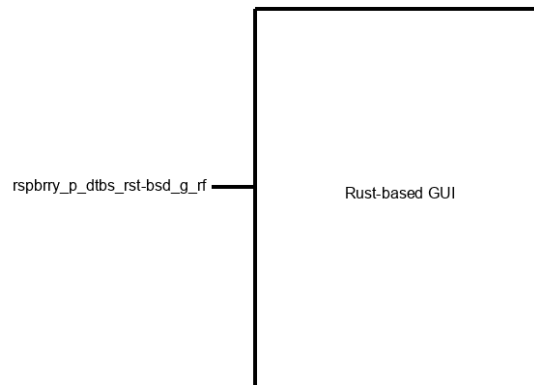


Fig.9. User Interface Block Diagram

The image below is a basic diagram of this block. Data from the Raspberry Pi will be the input to the UI, which will then output a chart of the data. This is included for a more general-purpose understanding of the block.

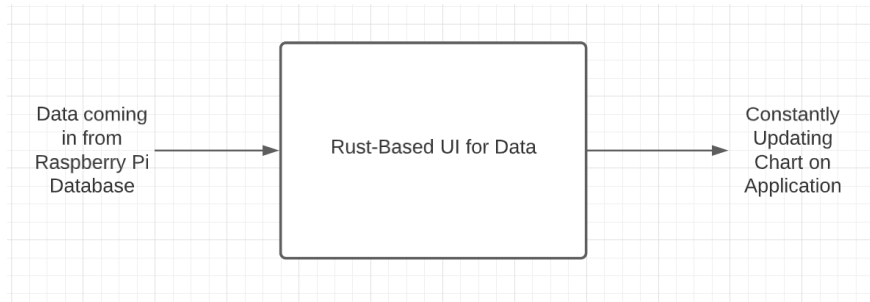


Fig.10. Basic Block Diagram

The image below describes a mock diagram of what the UI will look like. It will include a graphical representation of data being retrieved from the database, a title of the page indicating what the UI is for, and a possible method to export the data. This is included to understand the end goal of the UI as perceived by the user.

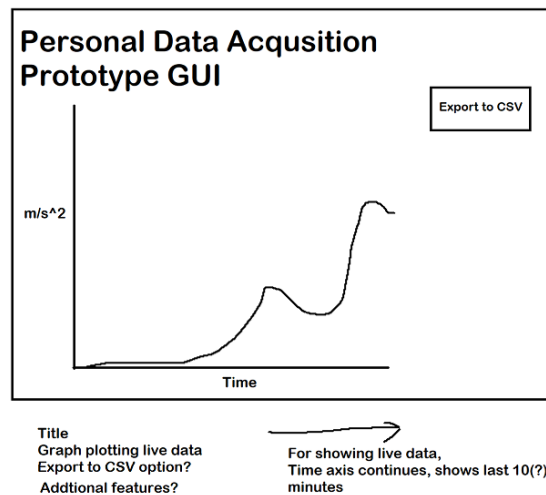


Fig.11. Basic Webpage Mock Design

The image below showcases how the data flows from one end of the block to the other. This is included to understand how data should be retrieved from the database as well as the data's destination.

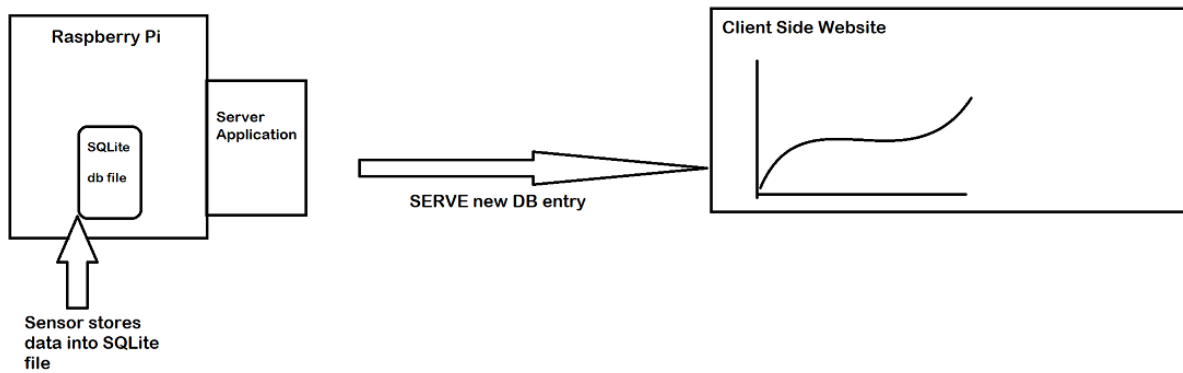


Fig.12. Basic Dataflow Diagram

Beginning Code:

The image below shows the entire app.py file, which is what is used to run the Flask application. When ran, a browser tab at URL localhost:5000 shows the application.

```

web-data-gui > flaskproject > webapp_env > app.py
1
2 from flask import Flask, render_template
3 import io
4 import base64
5 import sqlite3
6 import matplotlib.pyplot as plt
7 from matplotlib.figure import Figure
8 from matplotlib.animation import FuncAnimation
9 from datetime import datetime
10
11 # Create Flask App
12 app = Flask(__name__)
13
14 def connect_sqlite():
15     conn = sqlite3.connect("C:/Users/Preston/.vscode/web-data-gui/test3.db")
16     print("Database successfully connected")
17
18 def grab_new_row(conn):
19     curr = conn.cursor()
20     curr.execute("SELECT value FROM data")
21
22
23 @app.route("/data")
24 def data():
25     conn = sqlite3.connect("C:/Users/Preston/.vscode/web-data-gui/test4.db")
26     print("Database successfully connected")
27     cursor = conn.cursor()
28     cursor.execute('SELECT * FROM data')
29     data = cursor.fetchall()
30
31     fig = Figure()
32     ax = fig.subplots()
33     y_values = []
34     x_values = []
35     for row in data:
36         y_values.append(row[1])
37         timestamp = datetime.strptime(row[2], '%Y-%m-%d %H:%M:%S.%f')
38         x_values.append(timestamp)
39
40     ax.plot(x_values, y_values)
41     buffer = io.BytesIO()
42     fig.savefig(buffer, format='png')
43
44     image = base64.b64encode(buffer.getbuffer()).decode("ascii")
45     plot_url = f"<img src='data:image/png;base64,{image}'/>"
46     return plot_url
47
48 @app.route("/")
49 def home():
50     return render_template('main.html')
51
52 if __name__ == "__main__":
53     app.run()

```

Fig.13. Beginning Flask Code

In the image below, a showcase of the plotted SQLite database is presented using Matplotlib. This is the preliminary design of the UI and further features may be added.

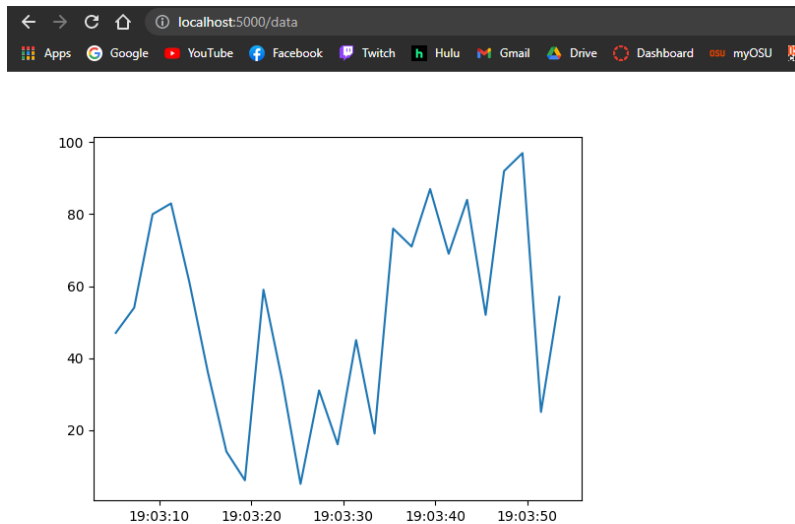


Fig.14. Embedded Graph of Database

The image below describes the SQL database and all values presented. This database is used to plot the graph in Figure 6.

SQL ▼		
sensor_name	value	time
motion	47	2023-02-12 19:03:05.295606
motion	54	2023-02-12 19:03:07.296241
motion	80	2023-02-12 19:03:09.296533
motion	83	2023-02-12 19:03:11.297465
motion	61	2023-02-12 19:03:13.298166
motion	36	2023-02-12 19:03:15.302428
motion	14	2023-02-12 19:03:17.309762
motion	6	2023-02-12 19:03:19.323872
motion	59	2023-02-12 19:03:21.338267
motion	34	2023-02-12 19:03:23.351811
motion	5	2023-02-12 19:03:25.362918

Fig.15. Example SQLite Database

Using Flask was much easier to implement, as there is a lot of helpful documentation on each specific library needed. The code currently involves multiple steps to arrive to the state that it is in:.

1. Flask App

- This is the basis of the application. It uses the Flask Python Framework to establish a Web server that when opened, displays a page. It's possible to incorporate standard Web Development programs, such as HTML, CSS, and JavaScript.
- 2. Reading SQL Database
 - In Flask, it is possible to open and query a database file. After creating a dummy database, I had made the Flask app open the database file and execute a "SELECT" query to pull data from the table.
- 3. Plotting Data
 - After retrieving the SQL data, it was then necessary to plot the data. I had grabbed each row of the SQL table, and translated the dummy sensor values as y coordinates, and the collected timestamp as x coordinates. This would then let me plot the two axis for the graph.
- 4. Embed Plot
 - Without this step, plots are created as separate windows. The next step was to embed the chart into the Flask Application. This is done by encoding the final chart into bytes and then passing it as an "img src" to then be used by the HTML.

4.3.3 - General Validation

The design features fit the need of our system as it provides all of the necessary requirements for displaying data. This is a pure software block that serves as the end of where the data must travel.

Alternate Approaches:

Egui is a free framework with a good amount of documentation, allowing for a low cost and approachable option for our display. Pushing the UI online is possible, as the repository for it has instructions on pushing it to the web. There are many concerns with this approach, including restraints due to understanding the complex programming language, then the egui template. While it would fit the job very well, I encountered multiple issues with just downloading libraries for sqlite (using rusqlite) and integrating it into egui. I attempted to use an eframe template (egui framework), provided by the creator of egui, but was unable to meet the requirements in time. With a constraint on time and technical skills, this approach may not be accomplishable in time.

An alternate UI that is possible is using NodeRED. This is web-based flow editor that allows for easy data manipulation through function blocks. I've personally used this

program in a project at my last internship. NodeRED is open source, with a lot of community support. It is also possible to install on the Raspberry Pi with a UI that presents live data. Upon further research, pushing NodeRED to the web contains many security concerns and additional requirements [5] that may prove too complicated to tackle for this project. A few restraints for this approach includes project partner requirement of a UI accessible by many platforms and security.

Chosen Approach:

Flask is a viable approach, with a considerable amount of documentation and resources for support. This seems to be the easiest approach with implementation, as there are plenty of libraries each with their own documentation. Some restraints to consider with this approach include a possible cost to push the Flask application to the web, and much smaller scalability compared to the other approaches.

With that being said, this approach is most likely the desired platform to use, considering restraints on time, cost, and technical skill. Creating a Flask application and reading SQLite data is a very accomplishable project, with a lot of guided resources to use for more complex interactions such as downloading plots, animating plots to create real-time updating plots, etc. Using this approach for the block will fit the needs of the system, as capabilities for reading SQL databases, plotting real-time data, and the ability to access the UI outside of locally are provided by external libraries. Flask is also a much less complex application than egui, and allows for a much more streamlined experience without the technical programming skills required for Rust.

The Project Partner initially requested a Rust-based platform, but after major project rescopes and other project setbacks, it may be necessary to use one of the other approaches.

4.3.4 - Interface Validation

TABLE VIII
INTERFACE VALIDATION

Rspbrry_p_dtbs__rf

Raspberry Pi Database to User Interface

Interface Property	Why is this interface property this value?	Why do you know that your design details <u>for this block</u> meet or exceed each property
Data Input: SQLite	<p>This database was chosen as it is local database application. Other applications were considered, such as online databases MySQL and MongoDB, however after discussion with the project partner, it was deemed that an online cloud-based database was unnecessary.</p> <p>SQLite was chosen for this small application as it is local and contains a lot of support for each UI approach.</p>	<p>The data is fairly simple, with a few columns of data per row. SQLite is perfectly capable of storing the data as it is a fully functioning SQL platform. SQL is easy to create tables and columns.</p>
Programming Language: Python/Flask	<p>This language is chosen for it's easy and capable features for this project. With a simple UI, Python can fulfill all necessary requirements.</p>	<p>There are a lot of documentation and capable features with Python. It is easy to read and use and can lead to quick results for a simple UI.</p> <p>There contains a library for each needed feature of the UI</p> <p>Database: sqlite3 Plotting: matplotlib</p>

		Web app: Flask
User Experience: Live data from the sensor	This is important here as the user should feel the data they are viewing corresponds to what is actually occurring.	While reading the sensor, there should not be too long of a delay in data being received. With the database being stored local to the Raspberry Pi, retrieving data should not take longer than 2 seconds.

4.3.5 -Verification Plan

To show the block passes interface properties and system needs, the data being present should be fast and correct to what the sensor is seeing. Since the UI is mostly for a viewing experience, users should be able to tell that the data they are seeing accurately corresponds to the sensor of the system.

TABLE VIV
VERIFICATION PLAN

Verification Plan	Purpose of Plan
<p>To show that data is being passed to the UI:</p> <ol style="list-style-type: none"> 1. Create SQLite database file with table and values 2. Import dummy data into the UI 3. Display the data with a plot that shows value vs. time 	<p>This plan verifies that the application is able to:</p> <ul style="list-style-type: none"> • Read data from an SQLite database • Display incoming data
<p>To show that the data is being presented live:</p> <ol style="list-style-type: none"> 1. Connect to the UI 2. Present the UI and system to 10 users outside of ECE 442 	<p>This plan verifies that the application is able to:</p> <ul style="list-style-type: none"> • Easily accessible from multiple devices • Ensure a user-friendly experience

3. Display data taken from a database for users to see 4. Poll user responses with the system and the UI 5. Ensure that 9/10 users felt the experience to be easy to understand and if they felt that the data was live	
---	--

4.3.6 - References and File Links

References:

- [1] E. Ernerfeldt. “egui: an easy-to-use GUI in Pure Rust.” Github.com. <https://github.com/emilk/egui#credits> (accessed Jan. 20, 2023)
- [2] Node-RED. <https://nodered.org/> (accessed Feb. 12, 2023)
- [3] Flask: web development, one drop at a time. <https://flask.palletsprojects.com/en/2.2.x/> (accessed Feb. 12, 2023)
- [4] SQLite. <https://www.sqlite.org/index.html> (accessed Feb. 12, 2023)
- [5] J. Knight. “How to safely expose Node RED to the Internet.” Github.com. <https://github.com/node-red/cookbook.nodered.org/wiki/How-to-safely-expose-Node-RED-to-the-Internet> (accessed Feb. 12, 2023)

File Links:

- [1] D. Rivers, P. Hang, M. Xu, Q. Pan. “Personal Data Acquisition Prototype.” Google Docs. <https://docs.google.com/document/d/1U1gulPXfxN9Xc0RDvS9g5hBjawdem4IQ2Ba02OBDt-0/edit?usp=sharing> (accessed Jan. 20, 2023)
- [2] D. Rivers, P. Hang, M. Xu, Q. Pan. “Personal Data Acquisition Prototype:

Impact Assessment.” Google Docs.

<https://docs.google.com/document/d/1cPkavx00DVzJtBKZPmZQOZ8vHezs9ui6Xf4nUIMfoRg/edit?usp=sharing> (accessed Jan. 20, 2023)

4.3.7 - Revision Table

Date	Revision
2/12/23	Preston Hang: Edited rough draft to include new versions of all sections based on Peer feedback
1/20/23	Preston Hang: Created Block Validation Draft

Section 5: System Verification

5.1 Universal Constraints

The Universal Constraints are specific boundaries that each and every ECE Senior Capstone project must obey. The purpose of these constraints are to ensure quality work in each and every project.

5.1.1 - The system may not include a breadboard

Our project will not include a breadboard as the platform for containing all electronic parts and connections. In the following image, our system will be demonstrated.

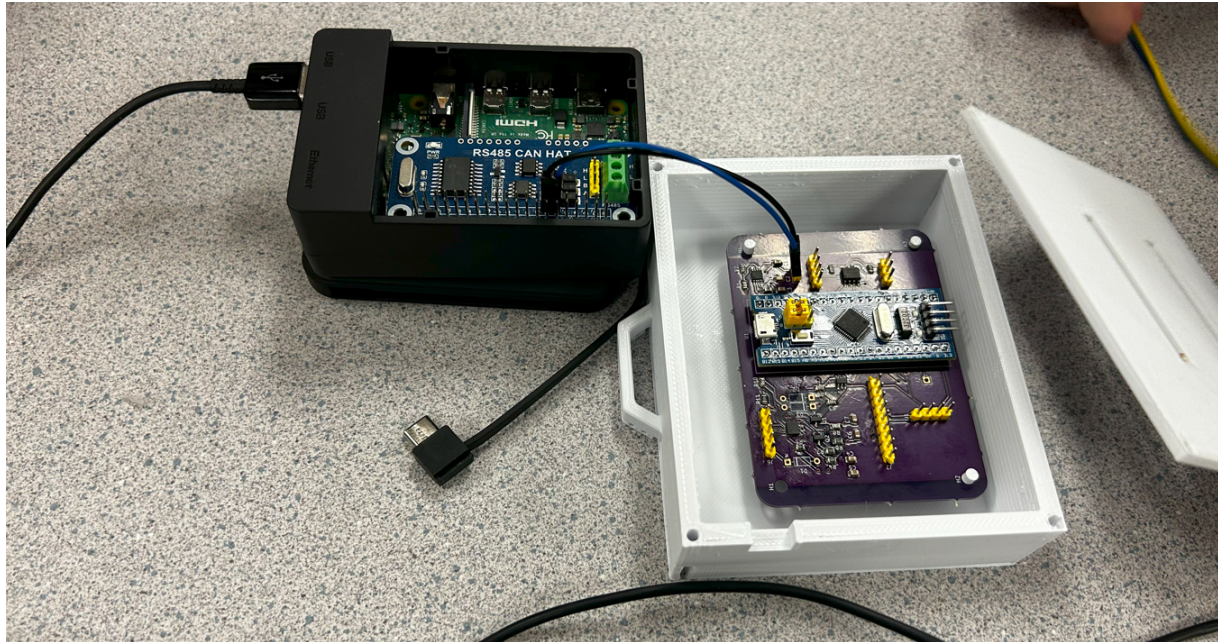


Fig.16. Full System Image

5.1.2 - The final system must contain a student designed PCB

The system will incorporate a PCB designed by our team to house connections. The PCB should include at least 30 non-connector surface mount pads. The PCB should have redesigned components to create less ‘purchased modules’ (see 5.1.5). There are 70 smt pads in the PCB, and the images are shown below.

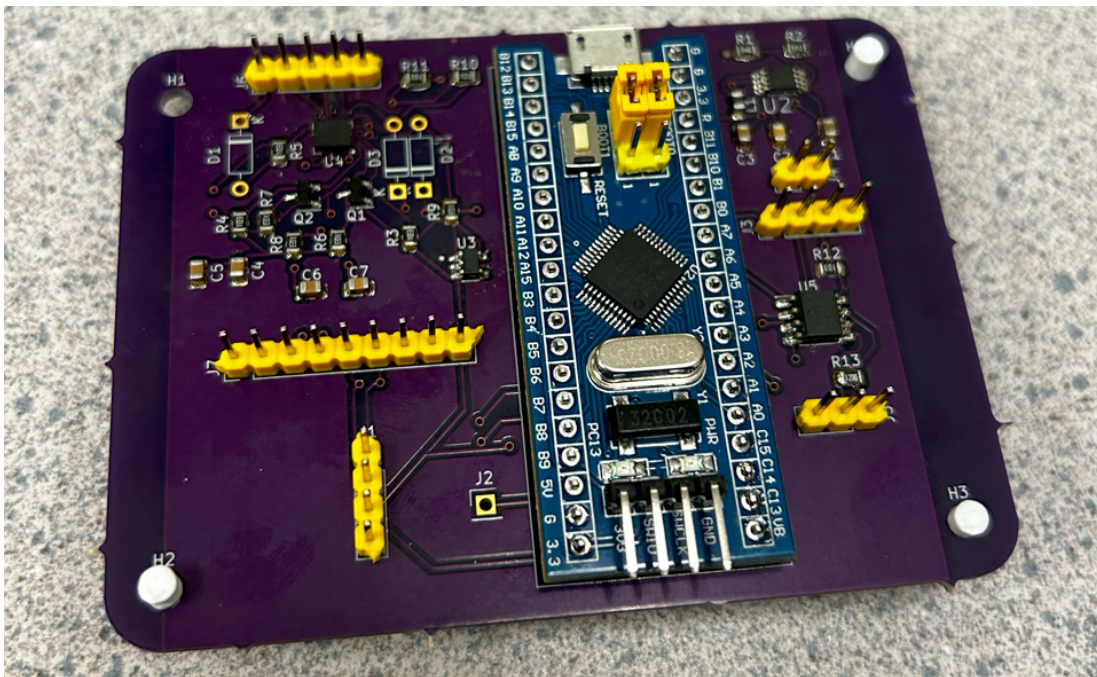


Fig.17. Physical PCB Layout

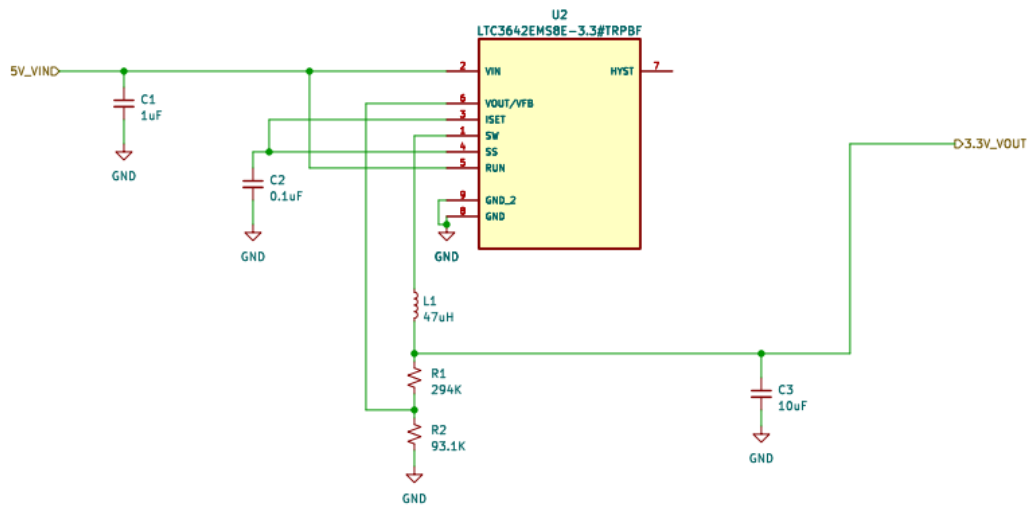


Fig.20. Buck Converter Schematic

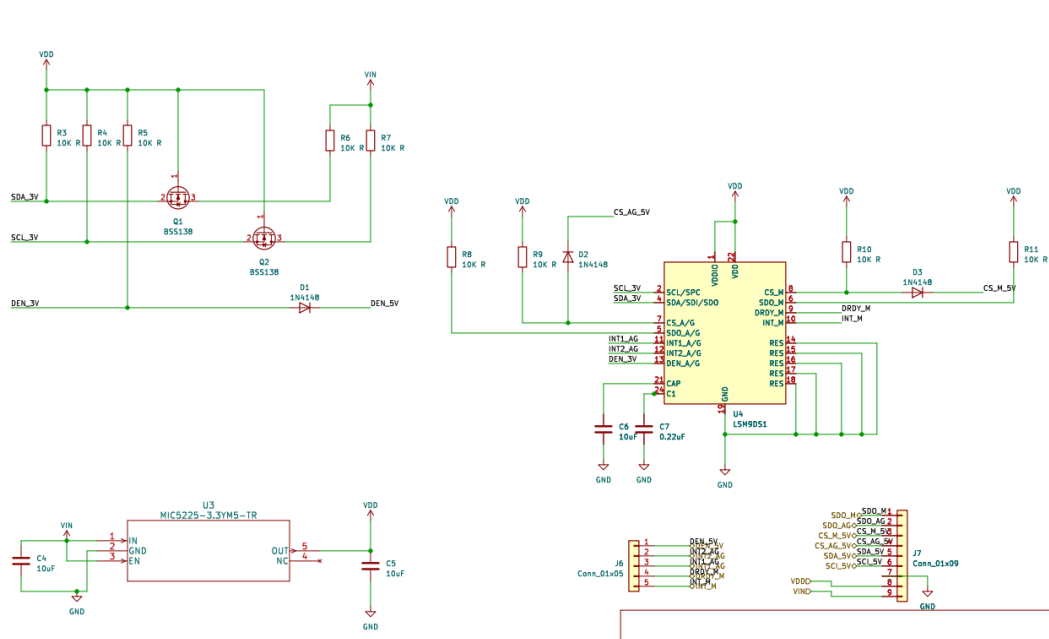


Fig.21. Sensor Schematic

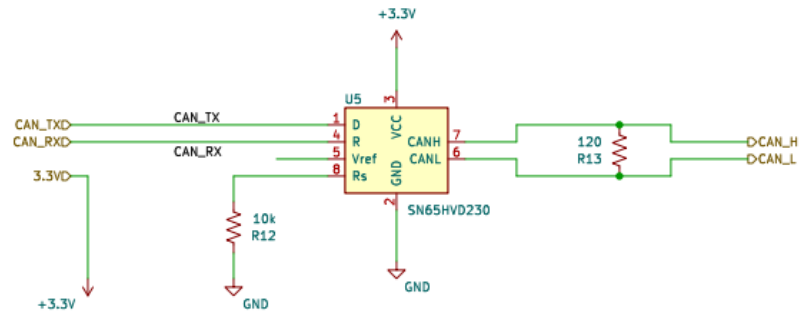


Fig.22. CAN Transceiver Schematic

5.1.3 - All connections to PCBs must use connectors

The system should not have any wires directly soldered to the board. Instead, connectors or headers should be included on the PCB. In the following image, our system will be demonstrated.

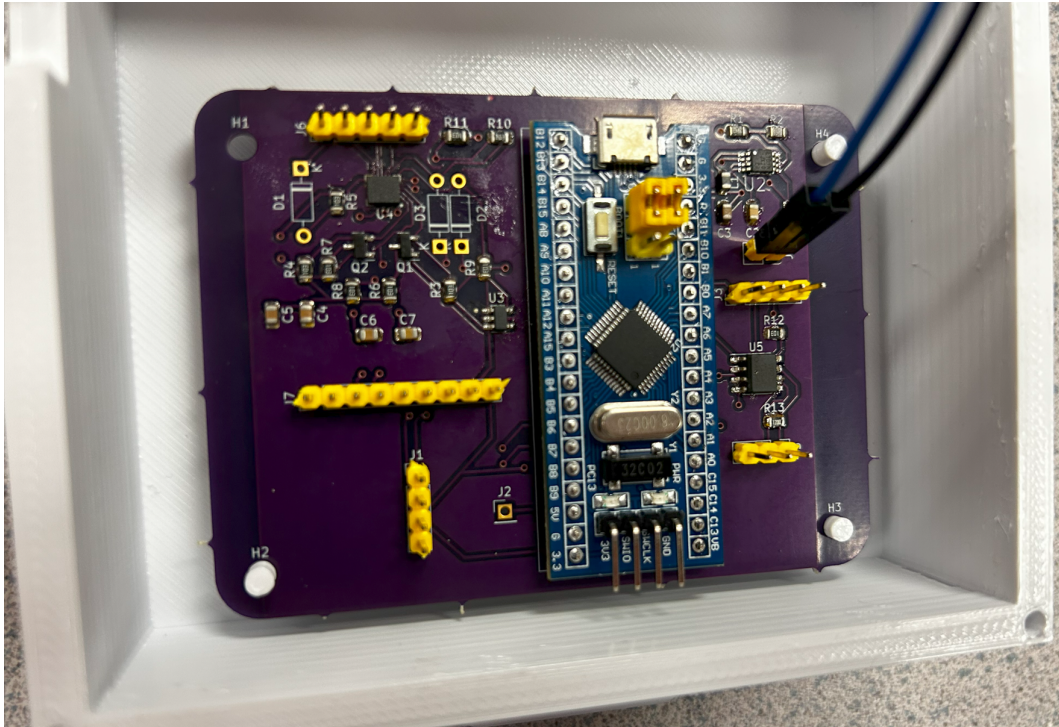


Fig.23. Full System Using Connectors

5.1.4 - All power supplies in the system must be at least 65% efficient

The system's should take into account power consumption and power waste. Thus, any and all power supplies that the system incorporates should be at least 65% efficient.

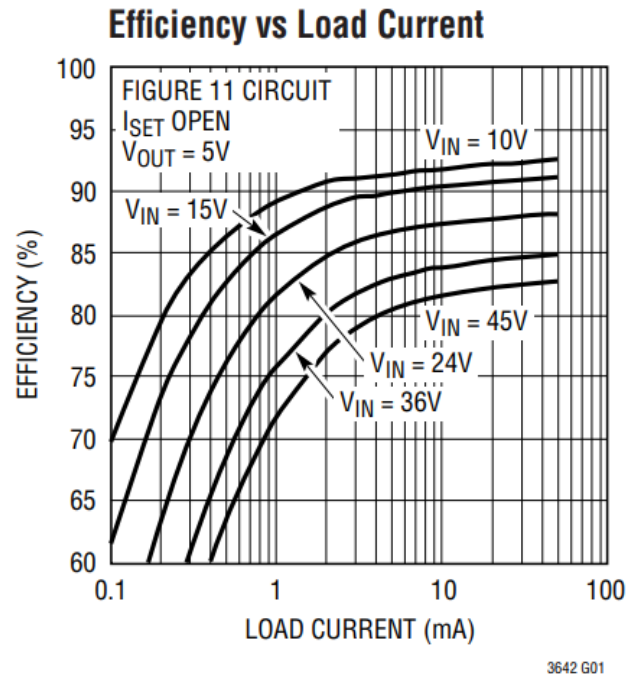


Fig.24. Efficiency vs. Load Current Chart For Buck Converter

The datasheet for the LTC3642EMS8E-3.3#TRPBF chip used in the Buck Converter states that “The efficiency of a switching regulator is equal to the output power divided by the input power times 100%.” According to this, our efficiency would equal:

$$\text{Efficiency} = (3.3V / 5V) * 100 = 66\%$$

Link to Datasheet:

<https://www.analog.com/media/en/technical-documentation/data-sheets/3642fc.pdf>

5.1.5 - The system may be no more than 50% built from purchased modules

At least half of the project's work must not be classified as a 'purchased module.' These are items that are purchased, not modified, and not adjusted for the project. Blocks that are a copy of a module/evaluation kit, but are laid out on the student-created PCB are no longer considered as purchased modules. There are a total of 10 blocks, two of which are 'purchase modules' and the other eight are 'construction modules'. Therefore, 20 percent are 'purchase modules'.

TABLE X
PURCHASED AND BUILT MODULES

Project Block	Statement
Sensor	BUILT: The sensor is included in the PCB design and are no longer considered as purchased modules.
PCB	BUILT: The PCB is fully designed by our team. Schematics used may reference datasheets, but PCB layout is entirely original.
STM32 Code	BUILT: This is the code written and flashed to the STM32. The code is written by Pan on our team.
STM 32 Development Board	BUILT: The STM32 microcontroller is included in the PCB design and is no longer a purchasable module.
CAN transceiver	BUILT: The CAN transceiver is included in the PCB design and are no longer considered as purchased modules.
RS484 CAN Hat	PURCHASED: This is a piece of hardware located on the Raspberry Pi that enables CAN communication.
Raspberry Pi	PURCHASED: The Raspberry Pi is unaltered by us. Only installations of libraries and software are done to the Raspberry Pi.
Raspberry Pi Communication Programming	BUILT: The code for communicating between the Raspberry Pi and the STM32 comes from student work and does not come from any provided code from Raspberry Pi or STM32 manufacturers.
Enclosure	BUILT: Students design their own 3D models and print them out.
User Interface	BUILT: The user interface is a pure software block that is designed by Preston Hang. None of the code included is a part of any demo code.

5.2 Requirements

5.2.1 - Acceleration Precision

5.2.1.1 - Project Partner Requirement

- Needs to be precise

5.2.1.2 - Engineering Requirement

- The system will report values in the X, Y, and Z axis with a precision of at least 0.01G.

5.2.1.3 - Verification Process

- Place the sensor flat on the tabletop. The value is 0 or close to 0.
- Slowly lift the sensor vertically.
- The data shows a slow increase, and the data is accurate to two decimal places.

5.2.1.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/1WhqzKqG8shMDULiMSDRurfA1LRnlOaca?usp=share_link

5.2.2 - CAN Bus

5.2.2.1 - Project Partner Requirement

- Data needs to be accessible over CAN

5.2.2.2 - Engineering Requirement

- The system will support the addition of more than one sensor via a CAN interface.

5.2.2.3 - Verification Process

- Slow-moving sensor.
- CAN HAT can hold more than one set of data.
- It can display multiple data on the screen.

5.2.2.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/16ck7e0C9VEXqjAfhGBkT_yClu5oj9dT4?usp=share_link

5.2.3 - Data Storage

5.2.3.1 - Project Partner Requirement

- Needs to hold a lot of data

5.2.3.2 - Engineering Requirement

- The system will store at least 1200 samples internally.

5.2.3.3 - Verification Process

- Demonstrate that there exists an SQLite database with 3 tables. The total amount of entries should be 1200, or 400 per table.
- To stress test this requirement, show each table with at least 1200 entries each, totaling to 3600 entries total

5.2.3.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/19BTIHN45vLQi7kzU7rTT8x5p3vh3k4Dd?usp=share_link

5.2.4 - Display Live Data

5.2.4.1 - Project Partner Requirement

- Live stream data to remote device (phone, tablet, computer)

5.2.4.2 - Engineering Requirement

- The system will display data that changes within 1 second of changes in motion.

5.2.4.3 - Verification Process

- View User Interface
- Show steady values for non-movement
- While moving the sensor, display a change of data on the User Interface while counting “1 Mississippi” to count off a second
- Alternatively:
 - Set timer for 10 seconds
 - Alternate between 1 second of movement and 1 second of no movement
 - Gauge difference between movement and no movement on User Interface

5.2.4.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/1Cnd11IacPHtPPAztrdHiHHRqdvmMVhWq?usp=share_link

5.2.5 - Read Sensor Data

5.2.5.1 - Project Partner Requirement

- Needs to have a wide range.

5.2.5.2 - Engineering Requirement

- The system will report X, Y and Z axis data in a range of at least $\pm 10G$.

5.2.5.3 - Verification Process

- Datasheet of sensor must include a range of $\pm 10G$
- Report three axis of values: X, Y, and Z data

5.2.5.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/1ggMoPlxyEYJ6i7d1LL1nQiksep1OyK3g?usp=share_link

5.2.6 - Rugged Device

5.2.6.1 - Project Partner Requirement

- Wearable must be able to handle the elements

5.2.6.2 - Engineering Requirement

- The system will retain functionality after being dropped from 5ft to concrete 5 or more times and sprayed with at least 150 ml of water from all angles.

5.2.6.3 - Verification Process

- First, the prototype was taken to a height of 5 feet.
- Let the prototype fall in free fall,
- Place the sensor flat on the tabletop after free fall.
- Slowly lift the sensor vertically.
- The data shows a slow increase.
- Secondly, the prototype was sprayed with water using a 150 ml water spray bottle.
- Place the sensor flat on the tabletop after free fall.
- Slowly lift the sensor vertically.
- The data shows a slow increase.

5.2.6.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/1AcA7C20hN38Wiz_oqcX7FHJqUr2mVIJ6?usp=share_link

5.2.7 - User Friendly Design

5.2.7.1 - Project Partner Requirement

- Enclosure must have some basic user friendly functionality

5.2.7.2 - Engineering Requirement

- The system will be reported as easy to use by at least 9 out of 10 users who are asked to reset the device, start and stop recording, and turn the device off and on.

5.2.7.3 - Verification Process

- Ask 10 people to use the system.
- Most of them should directly start and stop the recording by themselves.
- Ask them about their feelings.

5.2.7.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/1Djtf2_Bf9MFUysf-TvSPnZc0if8oXCH1?usp=share_link

5.2.8 - Wearable

5.2.8.1 - Project Partner Requirement

- Device must be wearable

5.2.8.2 - Engineering Requirement

- The system will be under 600 grams of weight.

5.2.8.3 - Verification Process

- zero the scale
- Put the device on the weighing scale.
- Check that the weight of the device is under 600 grams,

5.2.8.4 - Testing Evidence

- The link for testing (05/07/2023):
https://drive.google.com/drive/folders/1V8alg5VJ_o0g_fHcF_lerMhh_Dymjn5D?usp=share_link

5.3 References and File Links

The Link to datasheet for the LTC3642EMS8E-3.3#TRPBF chip:

<https://www.analog.com/media/en/technical-documentation/data-sheets/3642fc.pdf>

5.4 Revision Table

03/12/2023	Team: Section 5 drafts (5.1 - 5.4)
05/07/2023	Team: add more information in section 5

Section 6: Project Closing

This section goes over all closing remarks by our team. All information presented here is for anyone intending to continue this project, or anyone curious on our reflection of the project. It includes our recommendations, project artifacts, and presentation materials.

6.1 Future Recommendations

6.1.1 - Technical Recommendations

TABLE XI
TECHNICAL RECOMMENDATIONS

Recommendation	Reasoning	Potential Solution
Begin software development of User Interface earlier	<p>The UI can be almost independent from the project, as it doesn't rely on any hardware external to the Raspberry Pi. You can develop the UI first on your personal computer and then push updates to a Raspberry Pi.</p> <p>We found that development of the UI should have started earlier to prevent bugs and issues from blindsiding us as we approach the expo.</p>	<p>Start by researching early potential GUI frameworks, platforms, and programs.</p> <p>Evaluate what programming languages/frameworks you already know and what you want to learn</p> <p>Flask Documentation:</p> <p>“Welcome to flask,” <i>Welcome to Flask - Flask Documentation (2.3.x)</i>. [Online]. Available: https://flask.palletsprojects.com/en/2.3.x/. [Accessed: 28-Apr-2023].</p>

<p>Research flaws and pros of different microcontrollers - try to create the project with two different devices</p>	<p>With the microcontroller being the main brain of the system, having many options on implementation is key in case you run hardships with available libraries, demo code, etc.</p> <p>In our project ,we had chosen the STM32 microcontroller, but as time went on, implementation proved to be complex and challenging, and we could have saved many hours by having another microcontroller.</p>	<p>STM32, Arduino, WROOM implementations</p> <p>STM32 Information:</p> <p>Thomas Gravekamp, “STM32F103C8T6 - Blue Pill,” <i>STM32</i>. [Online]. Available: https://stm32-base.org/boards/STM32F103C8T6-Blue-Pill.html. [Accessed: 28-Apr-2023].</p>
<p>Ensure you are using parts that are available and regularly manufactured</p>	<p>Using parts that may not be available will bring downtime on your project, if your parts won't arrive in time.</p> <p>We had an initial design for our Buck Converter, but found that the chip used was no longer manufactured, and had to quickly find an alternate design.</p>	<p>Keep a living document of the parts you are using and what you may order (pay attention to chips specifically)</p> <p>Make sure parts are orderable before you send your PCB to be fabricated</p>
<p>Research multiple solutions on communication protocols to determine what is best for timing, efficiency, data load, etc.</p> <p>Allows flexibility to project and opens the door to reading different sensors with different protocols (or many sensors with a single protocol).</p>	<p>We had settled on CAN communication, as our project partner recommended, but found numerous difficulties getting that protocol to work out. If we had weighed more protocol options, it would have given us more flexibility with our project and how data is communicated - freeing up more time to focus on things like wire connectors (using new types of connectors), or sending multiple arrays of</p>	<p>CAN communication, USART, ModBus, UDP, I2C</p> <p>A. B, “Popular communication protocols in embedded systems - PART I,” <i>Gadgetronicx</i>, 30-Jul-2020. [Online]. Available: https://www.gadgetronicx.com/popular-communication-protocols-embedded-systems/. [Accessed: 14-May-2023]</p>

	data.	
--	-------	--

6.1.2 - Global Impact Recommendations

TABLE XII
GLOBAL IMPACT RECOMMENDATIONS

Recommendation	Reasoning	Potential Solution
Address privacy and security concerns for users.	In today's world, privacy and security have become critical factors for any technology that deals with personal data. As seen in the past with scandals like Facebook's Cambridge Analytica incident and Yahoo's data breach, failure to protect user data can have severe consequences both for users and the company involved.	<p>To address privacy and security concerns, the future team should develop and implement robust encryption and authentication measures for data storage and transmission. This can include using end-to-end encryption and strong password protection. Additionally, the team should consider incorporating transparent privacy policies and allow users to control their data, including what information is collected, stored, and shared.</p> <p>R. Duus, M. Cooray, and T. Conversation, "Research reveals the dark side of wearable fitness trackers," <i>CNN</i>, 01-Sep-2016. [Online]. Available: https://www.cnn.com/2016/09/01/health/dark-side-of-fitness-trackers/index.html [Accessed: 28-Apr-2023].</p>
Reduce the environmental impact of the device,	Electronic devices contribute to electronic waste and	To reduce the environmental impact, the future team

<p>particularly in terms of electronic waste and resource consumption.</p>	<p>consume significant amounts of resources during production. Improper handling of e-waste can lead to environmental damage and health hazards. Therefore, it is essential to minimize the environmental impact of the Personal Data Acquisition device.</p>	<p>should focus on designing the device for durability, modularity, and easy repair, thereby extending its lifespan and reducing the amount of e-waste generated. They should also use eco-friendly materials and processes during manufacturing, and partner with certified e-waste recycling companies to handle device disposal at the end of its life cycle. In addition, the team explore ways to minimize resource consumption during production by employing energy-efficient technologies and sourcing materials from sustainable sources.</p> <p>“Are Electronics Bad for the Environment?” <i>sustainability CO-OP</i>, March 24 2022. [online]. Available: https://thesustainabilitycooperative.net/2022/03/24/are-electronics-bad-for-the-environment/#:~:text=Many%20toxic%20substances%20are%20used,can%20also%20harm%20human%20health. [Accessed: 28-Apr-2023].</p>
--	---	---

6.1.3 - Teamwork Recommendations

TABLE XIII
TEAMWORK RECOMMENDATIONS

Recommendation	Reasoning	Potential Solution
Plan to have enough group meeting when you start a project	Sometimes there will be some problems between blocks, these problem may not appear when they work themselves, and on the other hand, some teammate will need other people's help when they meet some challenges	Two group meeting and one virtual meeting with project partner per week Brownlee, D. "The secrets to running project status meetings that work!" project management institute, 19 October 2008. [online]. Available: https://www.pmi.org/learning/library/secrets-running-project-status-meetings-7009
Do not be afraid to ask help from your teammates	It can make sure the work can finish on time if you always ask for help when you meet some challenge that cannot be solved. It is better to leave the problem until the deadline.	Keep good communication with teammates, and be enthusiastic to help your group members. Brownlee, D. "The secrets to running project status meetings that work!" project management institute, 19 October 2008. [online]. Available: https://www.pmi.org/learning/library/secrets-running-project-status-meetings-7009

6.2 Project Artifact Summary With Links

Link to Public GitHub Repository (includes code, schematics, and pcb files):

<https://github.com/prestonhang/data-acquisition-prototype>

TABLE XIV
SCHEMATIC AND DESIGN LINKS

Full Schematic	https://github.com/prestonhang/data-acquisition-prototype/blob/main/schematics/PersonalDataPCB.pdf
PCB Layout PDF	https://github.com/prestonhang/data-acquisition-prototype/blob/main/pcb/PersonalDataPCB_FullPCB.pdf
PCB 3D Model	https://github.com/prestonhang/data-acquisition-prototype/blob/main/pcb/PersonalDataPCB_3DModel.jpg
Enclosure Design	https://github.com/prestonhang/data-acquisition-prototype/tree/main/enclosure

TABLE XV
STM32 CODE LINKS

	Code Explanation	Pseudocode	Link
main.c	This is the main functionality of the STM32. It reads the sensor values and then sends it over the CANBus	// Setup // While 1 // Read changes from Sensor // Send through CAN // Use built in send function	https://github.com/prestonhang/data-acquisition-prototype/tree/main/src/stm32

TABLE XVI
RASPBERRY PI CODE LINKS

	Code Explanation	Pseudocode	Link
can_receive.py	This code file is for receiving message from the CAN bus coming from the STM32	# connect and setup CAN # On handshake # Listen() for new value # On new value, read message	https://github.com/prestonhang/data-acquisition-prototype/blob/main/src/canbus/receive.py
app.py	This code file contains main functionality of the User Interface. It includes reading the SQL database, and printing it to the system	def home(): # renders template of website page def livedata(): # on HTTP request, run #connect to SQL database While (1): #Query latest row # return response	https://github.com/prestonhang/data-acquisition-prototype/blob/main/src/ui/app.py
main.html	This is a simple website page that interfaces with the app. On connection to the app, it runs internal JavaScript code to plot the data	# Set connection to website as new event # On new event (when new value is added to database) #Parse new data into a variable # Push X values and Y values into corresponding axis #Update chart	https://github.com/prestonhang/data-acquisition-prototype/blob/main/src/ui/main.html

6.3 Presentation Materials

Link to Project Website:

<https://eecs.engineering.oregonstate.edu/project-showcase/projects/?id=2BMIKekLWcEqNozD>

Link to Public GitHub Repository: <https://github.com/prestonhang/data-acquisition-prototype>




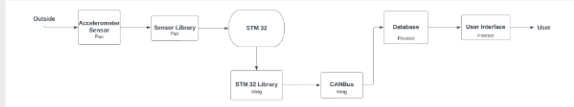
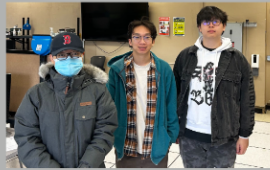
COLLEGE OF ENGINEERING	Electrical Engineering and Computer Science	ECE 16
<p>HARDWARE</p> <ul style="list-style-type: none"> • STM32F103 "Blue Pill" Microcontroller Development board • LSM9DS1 Accelerometer + Gyro + Magnetometer 9-DOF • SN65HVD230 CAN Board Transceiver • RS485 CAN Hat • Raspberry Pi 4 Model B <p>SOFTWARE</p> <ul style="list-style-type: none"> • STM32CubeIDE <ul style="list-style-type: none"> ◦ C/C++ based programming environment • CAN Send and Receive <ul style="list-style-type: none"> ◦ Python • Flask <ul style="list-style-type: none"> ◦ Python, JavaScript, HTML • SQLite Database <p>GITHUB REPO</p>  	<p>PERSONAL DATA COLLECTION PROTOTYPE</p> <p>INTRODUCTION</p> <p>One of the most important that individuals include the ability to collect numerical data. Data Acquisition systems are present today; however, the market has few accessible options between the industry-level and a hobbyist-level.</p> <p>Our goal is to design a data collection system that bridges this gap, allowing user applications that fit in this middle ground.</p>  <p>System Prototype</p>  <p>Top Level Block Diagram</p> <p>TECHNICAL EXPLANATION</p> <p>Our project primarily involves collecting relevant data through an accelerometer sensor, integrating this data using an STM32 Blue Pill microcontroller, and preparing to transmit it to a Raspberry Pi via the CAN BUS. The Raspberry Pi will collect and store data from the previous level and, finally, present the data in tabular form on a mobile device through wireless network transmission.</p>	<p>ENGINEERING REQUIREMENTS</p> <p>The system needs to:</p> <ol style="list-style-type: none"> 1. Report values in the X, Y, Z axis with a precision of at least 0.1G 2. Support the addition of more than one sensor via a CAN interface 3. Store at least 1200 samples internally 4. Display data that changes within 1 second of changes in motion 5. Report X, Y, Z axis data in a range of at least $\pm 10G$ 6. Retain functionality after being dropped 5 feet to concrete and sprayed with at least 150 ml of water from all angles 7. Be determined "easy to use" by at least 9/10 users 8. Under 600 grams of weight <p>MEET THE TEAM</p>  <p>Mingqian Xu xuming@oregonstate.edu CANBus Communication, Enclosure</p> <p>Preston Hang prestantanhang@gmail.com User Interface, SQLite Database, PCB Design</p> <p>Qunyi Pan panq@oregonstate.edu STM32 and Sensor Interfacing</p>

Fig.25. Project Poster

Link to Poster:

<https://docs.google.com/presentation/d/1Sj9eZq7G-FpIYSQ34MViXatUqU79vJ6T/edit?usp=sharing&ouid=106987283591064473662&rtpof=true&sd=true>

6.4 Revision Table

5/14/2023	Preston: Edited Technical Recommendations, added more project artifact links
04/25/2023	Team: Section 6 drafts (6.1 - 6.4)