System Design Document

For

NASA VESTIBULAR CHAIR

Team members:
Brandon Boyle-Fagan
Matthaeus Gebauer
Miles Osborne
Dylan Prothro
Noah Reid
Kent Wilson

Version/Author	Date
1	10/6/2022
1.1	10/27/2022
1.2	11/8/2022
2.1	12/3/2022
2.2	12/4/2022
3.0	12/8/2022
3.1	12/9/2022
3.2	12/11/2022
4.0	2/18/2023
4.1	3/10/2023
4.2	4/26/2023
4.3	4/27/2023

TABLE OF CONTENT

SYSTEM DESIGN DOCUMENT

1 INTRODUCTION

1.1 Purpose and Scope

The goal of the Vestibular Chair project is to restore the basic hardware functionality of a rotating chair that was acquired by Embry-Riddle that was used by NASA to test the human vestibular system and provide it with a new controller. This consists of ensuring the system can reach a specified RPM, hold that specified RPM for a set duration, and allow the servos to gradually slow down to idle. With the controller created and able to provide the intended functionality to the vestibular chair, we've begun integrating more modern software and hardware tools to improve the "quality of life" features of the chair. This includes a web interface, custom test profiles/sequences, and the ability to read and store sensor data from the chair.

1.2 Project Executive Summary

This section provides an overview of the NASA Vestibular Chair project from a macro perspective, showing the framework with which the system design was conceived.

1.2.1 System Overview

The NASA Vestibular Chair system can be broken down into three hardware components, the controller module, the patient remote, and the web interface. The controller module is composed of an embedded development board, motor controller module, 20x4 LCD, and other components to be able to handle digital and analog inputs for the chair motor. The patient remote is a wireless device with two buttons to indicate directionality. While the patient is spinning in the chair, the button pressed will indicate what direction they believe they are spinning. The additional web interface is planned to interface with the control module to allow for more precise measurement and input for the chair. The hardware components of the vestibular chair itself consist of a tachometer, motor, and the actual chair itself as well as some other pins setup for other once-used analog measurements. Figure 1 listed below, details a use case diagram of the NASA Vestibular chair system.

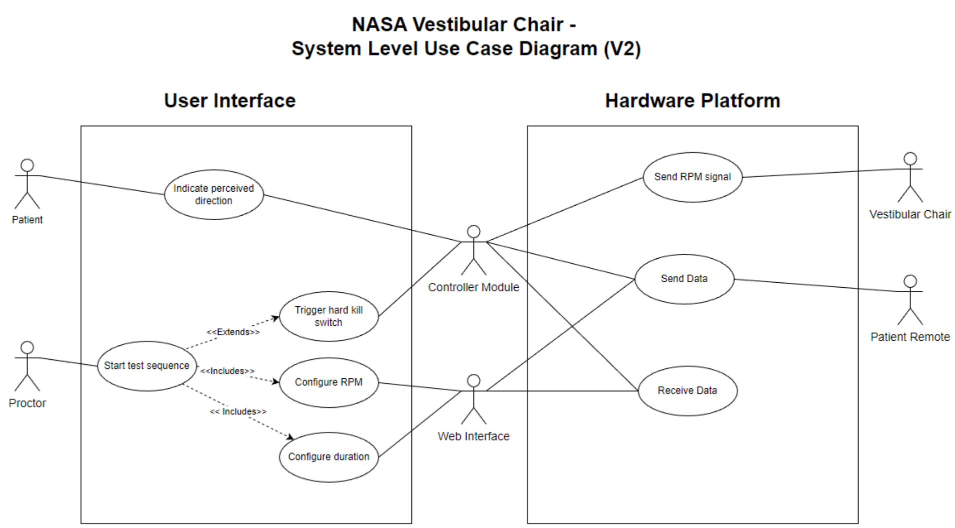

Figure 1: Use Case Diagram for NASA Vestibular Chair

1.2.2 Design Constraints

The development team plans to keep as much of the original internal hardware of the chair as possible. One of the major constraints in the design of the controller of the chair and its interface is to have the new components work with the older technology present in the chair. Another constraint comes from the speed the chair is going to be allowed to move. The chair must have a constraint to be set to not move faster than a rotation of 100 degrees per second, and with that constraint, the controller must be programmed to not allow an input of voltage that would cause a rotational speed higher than that. The controller also must have a mechanical kill switch, meaning it needs to be connected to the power being sent to the chair via the controller to be able to shut off power being sent to the chair if necessary.

1.2.3 Future Contingencies

Future contingencies to ensure smooth use of the controller during its lifespan include a digital part of the controller to account for any analog issues, with the intent to keep an analog and digital component to control the inputs of the chair.

1.2.3.1 Future Contingencies

Another contingency comes from the alternative components that the team researched in case the planned components are not available or do not work for the direction of the project. This includes researching two different types of motor controllers, Pololu - High-Power Simple Motor Controller G2 24v12 and Pololu - RoboClaw 2x15A Motor Controller (V5E). While the G2 is the preferred motor controller for the system, the functionality of the RoboClaw was investigated and was determined to be able to be used in place of the G2 if issues arise such as lack of power output or component damage.

1.2.3.2 Safety Measures

The inclusion of a mechanical kill switch is a contingency to avoid having the chair spin out of control if too high of a voltage is supplied to the device. Additionally, limits are placed within the software to prevent too high operation with an unstable speed.

1.2.3.3 Embedded Software Compatibility

Another contingency that needs to be accounted for is volatility in hardware changes, especially as it pertains to the software and drivers present in the components of the controller. Initially, this was accounted for through the STM boards due to their interchangeability code-wise, along with the code used to drive the motor controller being generic enough to not be locked into the specific motor controller that was chosen, meaning it could be used with different parts if the current one was to break for instance. However, due to better functionality on the front of data collection, the decision was made to utilize an Adafruit Feather nRF52840 express development board instead. This component is a comparable alternative to the Nucleo board we had intended to use but has the added benefits of utilizing a smaller form factor, being slightly more affordable, built-in battery charging circuitry, documented compatibility with Arduino Framework, and less complex Bluetooth libraries.

1.3 Document Organization

This System Design Document is organized into six sections:

1. Introduction
2. System Architecture
3. Human-Machine Interface
4. Detailed Design
5. External Interfaces
6. System Integrity Controls

The purpose of this document is to give the reader a comprehensive understanding of the system design and the characteristics of how the system will work.

1.4 Project References

No references at this time.

1.5 Glossary

No items to list at this time.

2 SYSTEM ARCHITECTURE

The system uses an interface of hardware and software to control the movement of the NASA Vestibular Chair as well as measure its speed and other readings.

2.1 System Hardware Architecture

The composition of the hardware is represented by:

- NASA Chair
  - The NASA chair consists of inner hardware such as its motor and sensors already installed in the device such as a tachometer, which will be used to measure the speed of the chair during operation. The sensors are used to provide feedback to the system in terms of adjusting the speed based on the present load in the chair.

- Controller Module
  - The controller module consists of a dedicated microcontroller, dedicated motor controller, and physical I/O such as a potentiometer and buttons to quickly control the state and execution of the system. The current module setup contains an Adafruit Feather nRF52840 microcontroller to control the system and interface with the Polulu G2 24v12 motor controller to achieve the desired functionality. The motor controller serves as an important factor for the operation of the motor within the chair, as it is a dedicated motor controller as opposed to having the operation handled by another component.

- Wireless bi-direction input
  - Part of the core functionality of the NASA vestibular chair is to allow the user to indicate what direction they believe to be spinning in. Due to the physical constraints of the chair spinning, we must use a wireless input to give the user this feature. The current model for this input is a controller with two buttons to indicate the direction the user believes they are spinning may that be clockwise or counterclockwise.

2.2 System Software Architecture

The software is centered around an embedded systems project mostly developed using C++. The development board utilized is the Adafruit Feather nRF52840 Express which has extensive compatibility with the Arduino Framework. More importantly, the manufacturer, Adafruit, has developed a Bluetooth library for use with the onboard Bluetooth module. Due to the urgency of the project, writing drivers for each component of the project is not feasible and would artificially and unnecessarily increase the difficulty of the project.

The Adafruit Feather nRF52840 Express is the core of the controller module and will interface with several peripherals including the motor controller and a 20x4 LCD. Both peripherals can be controlled via the I2C protocol which makes the software written easier to implement and maintain.

The second aspect of the system is the web interface which gives the proctor more options as it relates to creating test sequences and displaying data during the test. Data is read from the chair via the analog pins and then temporarily stored in a buffer. The data is then sent over UART to the web interface and displayed on the host computer. Likewise, when the proctor wants to actuate the chair, they will use the web interface to configure the test. When ready, the test sequence information will be sent via commands over UART and then processed by the Adafruit development board. The system shall then begin to follow the sequence of events as indicated by the proctor's designated test sequence.

A state transition chart for the system excluding the web interface is listed below in Appendix A and Appendix B. The web interface was excluded at this time to focus on the current priority of more robust mechanical control. However, when the web interface and accompanying features are introduced, the state transition chart should be very similar.

2.3 Internal Communications Architecture

In the NASA Vestibular Chair System, the main communication channels happen over UART from sending and receiving commands from the web interface and processing digital signals from the controller module I/O. Additionally, I2C will be utilized by the Adafruit development board to interface with the motor controller and the seven-segment displays. Lastly, Bluetooth will be utilized to receive data from the wireless input device indicating what direction the user believes they are spinning in. This is feasible because the Adafruit Feather nRF52840 Express has a built-in Bluetooth module. Figure 3 below describes the internal communication between the system components. This figure does not consider the protocol or medium used for communication.

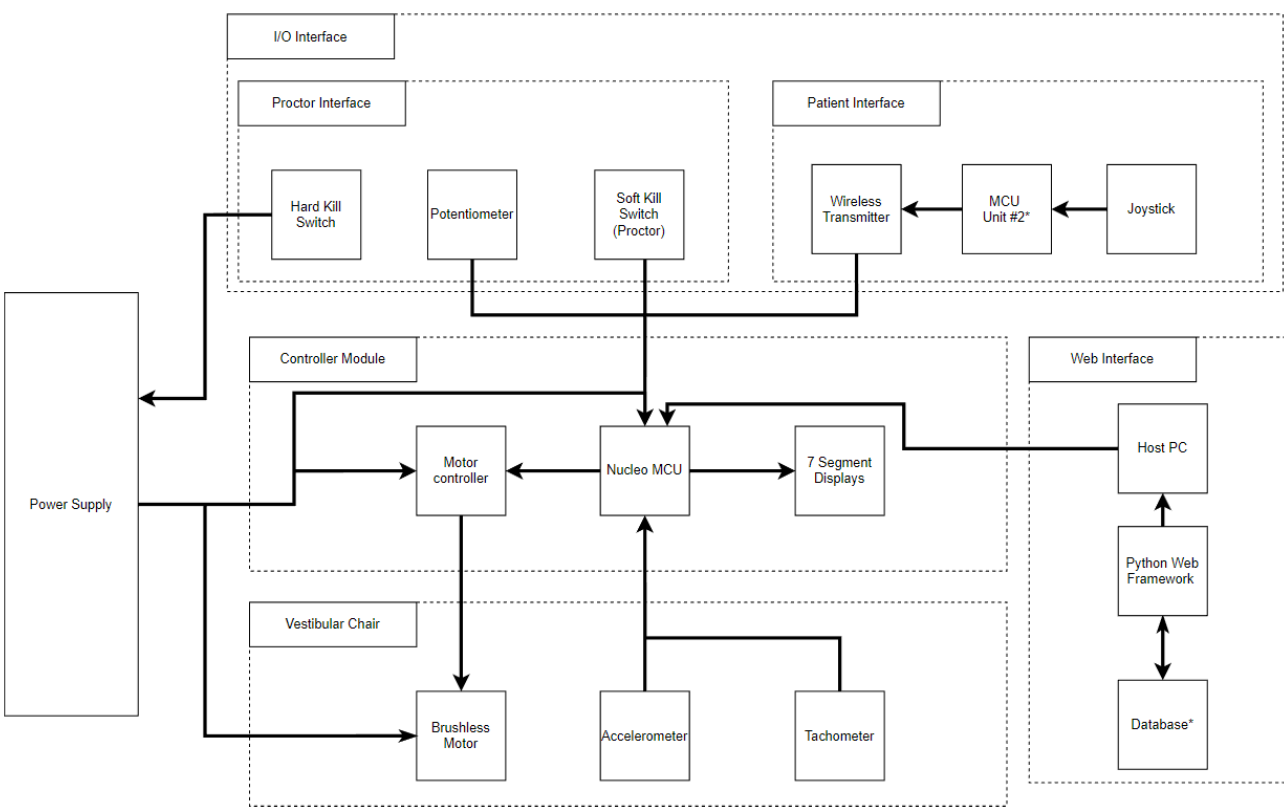

Figure 3: High-Level Hardware Communication Overview

Regarding communication between core systems, Bluetooth LE and UART are utilized. To collect data from the patient remote, a peer-to-peer connection between the controller module and the patient remote is established. While Bluetooth is bi-directional, the method utilized here is uni-directional such that the control module operates as the receiver and the patient remote will serve as the transmitter. When the chair is in operation and a connection is established, the remote will send packets of data the control module will log and process.

The latter two items are physical connections. The web interface is intended to use UART/serial to interface with the control module. Note that the web interface will be hosted on a desktop/laptop system with USB ports. Even though the physical UART pins are not utilized, the USB ports being used are capable of converting a USB signal to a UART-compliant packet. Sending commands to the vestibular chair is done through an analog signal generated by the motor controller component and carried through a Cat 6 ethernet cable. Figure 3.1 listed below details the communication architecture between the critical components of the system.

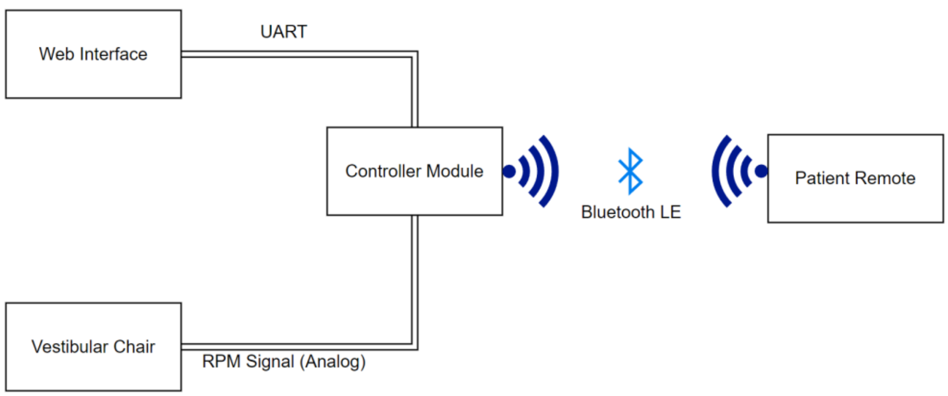

Figure 3.1: System-Level Communication Overview

Figure 4 listed below is a simple data flow diagram, detailing the critical piece of information being transferred from components. To make the diagram simpler, the controller module is assumed to include all required electronics including the Adafruit development board, G2 motor controller, seven-segment displays, switches, and potentiometers.

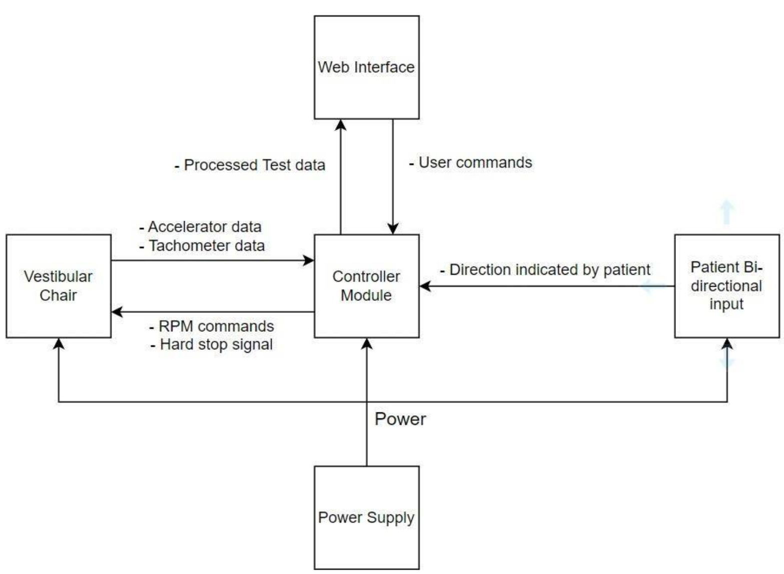

Figure 4: Level 0 Data Flow Diagram for NASA Vestibular Chair System

3 HUMAN-MACHINE INTERFACE

Human input is required to determine the speed of the vestibular chair and by association, the voltage and amperage required to achieve that speed. This interaction will cause the user to be able to adjust the chair's speed through the controller device.

3.1 Inputs

The inputs of the system will be given from the controller to the chair to allow movement of the chair or to set up testing cases for the speed to be applied without needing to be adjusted by the user physically. These inputs are user-generated, as the individual in control of the device sets the RPM that the chair is being given through test cases. The sensors used will also count as input, including things such as the tachometer and accelerometer. Another input will come from the individual sitting in the chair, as they will be able to send data indicating the direction, they believe they are spinning in with the bi-directional input device.

3.2 Outputs

The output response from the chair should be the tachometer reading to feed into the controller to control the speed via its feedback. This will allow the system to avoid reaching a speed that outpaces the scope of the controller. The other output is that given from the chair itself, which is its actual rotation given the input voltage supplied by the controller. The feedback from the chair should also be able to communicate the direction it is rotating to the controller. The bi-directional controller will also have an output of which direction the user indicated they were spinning in.

4 DETAILED DESIGN

This section contains detailed information about the hardware and software design of the system.

4.1 Hardware Detailed Design

The system is centered around the Adafruit Feather nRF52840 Express. This component will perform the following roles at a minimum:

1. Accept user input from the proctor
2. Perform data acquisition at a sampling rate of at least 20hz
3. Process commands sent via UART
4. Send data to peripherals through I2C
5. Process commands via Bluetooth from wireless bi-directional input
6. Send data to the web interface via UART
7. Control the vestibular chair's RPM via commands sent to the motor controller.

Figures 5, 6, and 7 are a simplified overview of the hardware and components present in the system. Note, that these diagrams do not document the internal communication between each module.

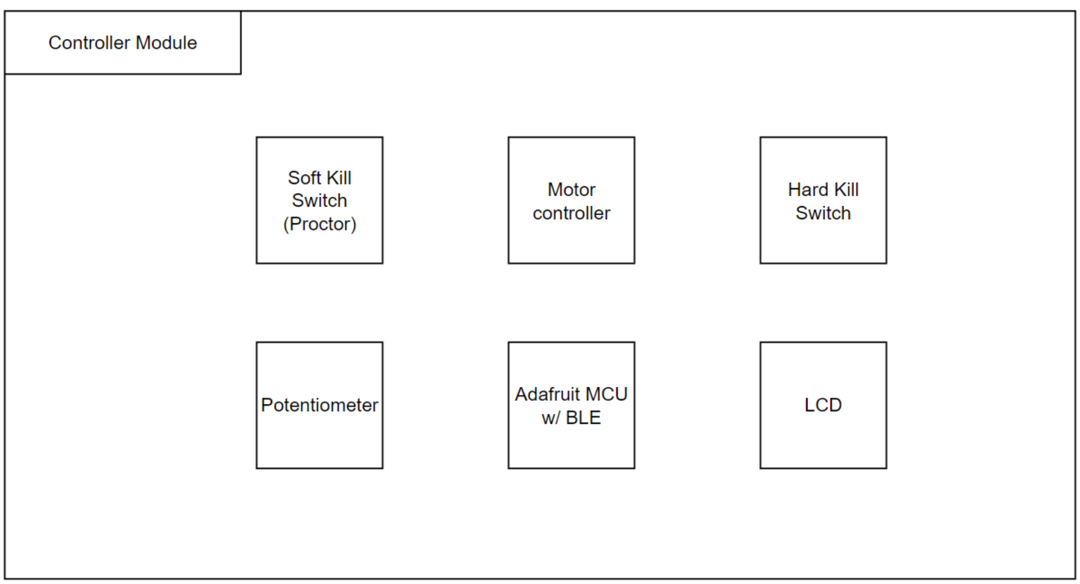

Figure 5: Simplified overview of NASA Vestibular Chair Controller Module

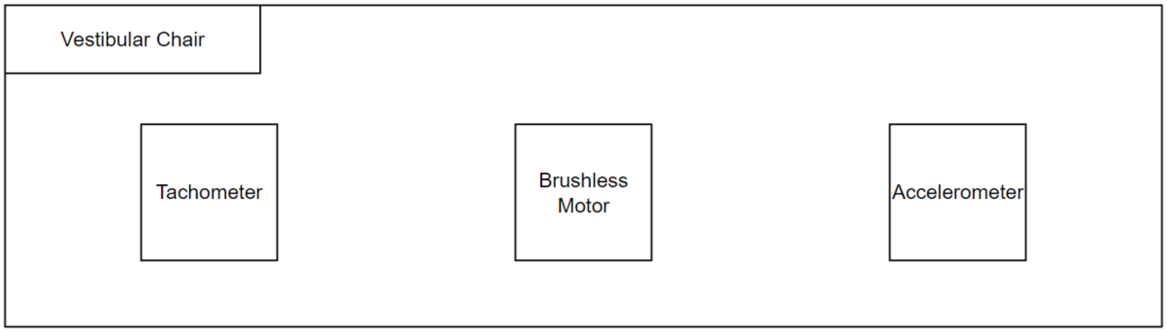

Figure 6: Overview of NASA Vestibular Chair Components

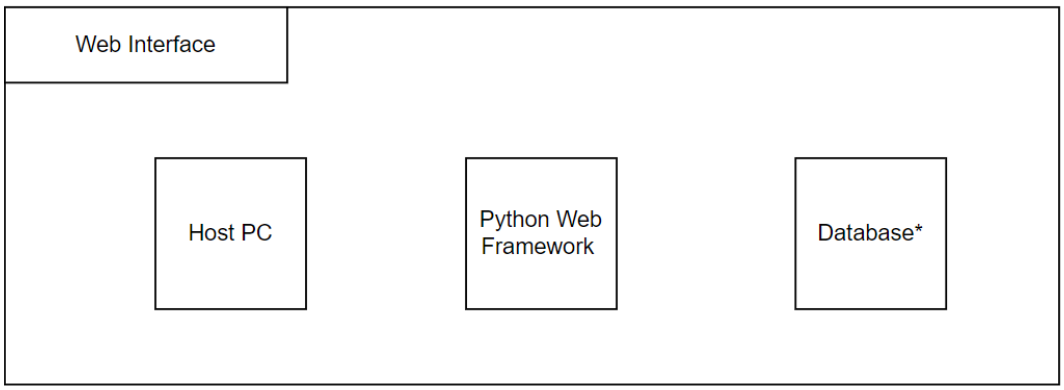

Figure 7: Overview of Web Interface Components

4.2 Software Detailed Design

This system is compromised of multiple components and interfaces which exceed the scope of the document to discuss at length. However, the system can be condensed and discussed based on the five major components and the software dependencies of each. Those components consist of the following:

1. Vestibular chair
2. Controller Module
3. Patient remote
4. External logging application
5. Mbientlab Bluetooth IMU*

A simple architecture diagram is presented below in Figure 8. It is worth noting that item five from the list above was a recent addition to the system and existing figures have not been updated to account for the inclusion.

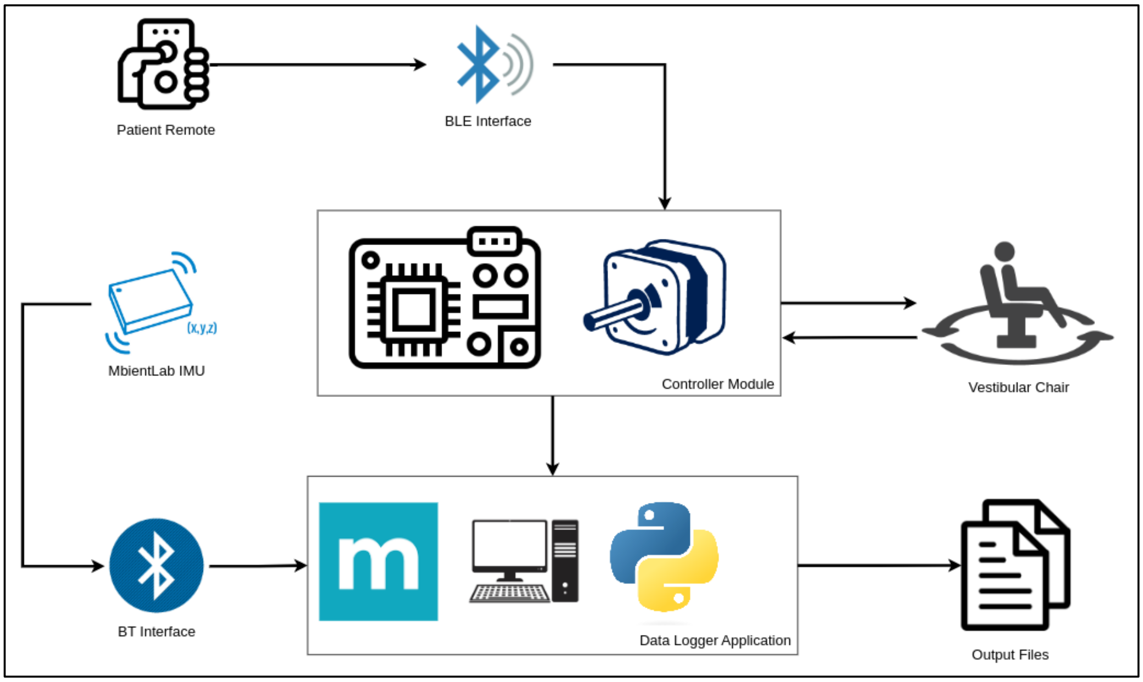

Figure 8: Simplified system architecture diagram

4.2.1 Vestibular Chair

The vestibular chair itself has no software dependency or built-in software system. Its operation is solely dependent on the inputs of the controller module. Safety mechanisms are built in to prevent unintended or non-deterministic behavior. For outputs, the value of the onboard tachometer is obtained by the controller module and logged by the data logging application.

4.2.2 Controller Module

The controller module contains the main processing capabilities of the system. Since we utilized the Adafruit Feather nrf52840 express microcontroller, the language utilized was C++ with the Arduino framework. Writing drivers for the hardware in C/C++ or using Python was available but utilizing the Arduino framework removed the need to develop and test our drivers when there is sufficient support and documentation with the Adafruit Feather board and the Arduino framework. Likewise, our development environment was Visual Studio Code (vscode) with the Platformio extension. The Platformio extension allowed us to utilize a more capable development tool with library management, version control, and multiple build configurations.

The Adafruit Feather microcontroller had to interface with a Pololu motor controller, an LCD, and another Adafruit microcontroller contained in the patient remote through Bluetooth. The interface between the motor controller and LCD was accomplished through the I2C protocol. The Arduino wire library provided the API for completing the I2C transactions for both peripherals mentioned. In particular, the exact communication scheme for the motor controller is mentioned in a later section.

Bluetooth communication was achieved through the use of the provided Adafruit Bluefruit Bluetooth libraries. There are specific configurations for the nrf52 Bluetooth modules utilized on the Adafruit microcontrollers which were utilized. After understanding how to utilize the libraries, we configured the module to only connect and respond to messages sent by the patient despite the potential presence of other Bluetooth signals in the area.

The last aspect of the controller module is the communication between itself and the data logging application. From the perspective of the controller module, communication is accomplished through the use of the Arduino Serial library and Software serial libraries. These libraries utilize the UART protocol which sends messages to the data logging application through USB.

4.2.3 Patient Remote

The patient remote is much simpler than the controller module. Its only functionalities are to respond to button presses by the patient in the chair, connect to the controller module via Bluetooth, and send the button press data to the controller module. Its development environment is identical to the controller module due to the use of identical microcontrollers. To make the process easier, we have configured both modules to only connect to each other despite the potential presence of other Bluetooth signals in the area.

4.2.4 Data Logging Application

The data logging application was developed in Python and runs on an external laptop during the trials. Currently, the application only runs on Windows 10 but Windows 11 support should be functional. Development of the tool is only supported on Windows machines at this time. Linux development was previously possible but the inclusion of packages to connect with the MbientLab MMS IMU has suspended that development. The development environment utilized consists of Python 3.9 and the PyCharm ide for easy integration and package management. To keep the application simple as possible, only a few external dependencies were included with the project. Pyserial was utilized to initialize and conduct serial communications between the controller module. Additionally, the Mbientlab Metawear package was utilized to conduct with the Bluetooth IMU. This package was not difficult to integrate but does require that the host PC running the application has a Bluetooth module on board that was enabled and that the respective IMU was in extreme proximity to be connected.

4.2.5 Bluetooth IMU

The Mbientlab MMS IMU was a recent addition to the system. Adding this component came about due to the discovery that the tachometer in the vestibular chair was likely damaged at some point. The data recorded was extremely noisy and irregular to an extent that made it unusable. After discussion, the decision was made to incorporate an off-the-shelf wireless component that would seamlessly integrate with our existing system. While the IMU comes equipped with a fully-fledged sensor suite including sensor fusion, we are only concerned with the gyroscope values produced by the BMI270 sensor inside. As mentioned above, the IMU connects with the data logging application through the open-source Mbientlab Metawear package. Gyroscope values are then streamed and processed by the external application which is utilized to determine when the chair comes to a complete stop.

5 EXTERNAL INTERFACES

The current external interface is the plan to add a web-based component for the controller. With the controller module design showing positive results, focus on the web interface has been prevalent although it has yet to be completely flushed out. However, it will interface with the controller and serve as the main function for the proctor to interface with the chair and its testing capabilities.

5.1 Interface Architecture

The web interface will interface and interact with the microcontroller of the system which is serving as the main driving force for the motor controller, which powers the chair. This level of interaction will allow the user of the web interface to dictate a certain test sequence via specific parameters such as the length of the test and the speed of rotation for the chair. This will then run through the microcontroller board which will run the code to follow those parameters when it sends data to the motor controller, effectively controlling the rotation of the vestibular chair.

5.2 Interface Architecture

This section details specifications regarding the communication methods for the essential modules of the NASA Vestibular chair system. The figure below gives an overview of how data is transferred between components and the protocol or method utilized. This figure also indicates the directionality of the data sent.

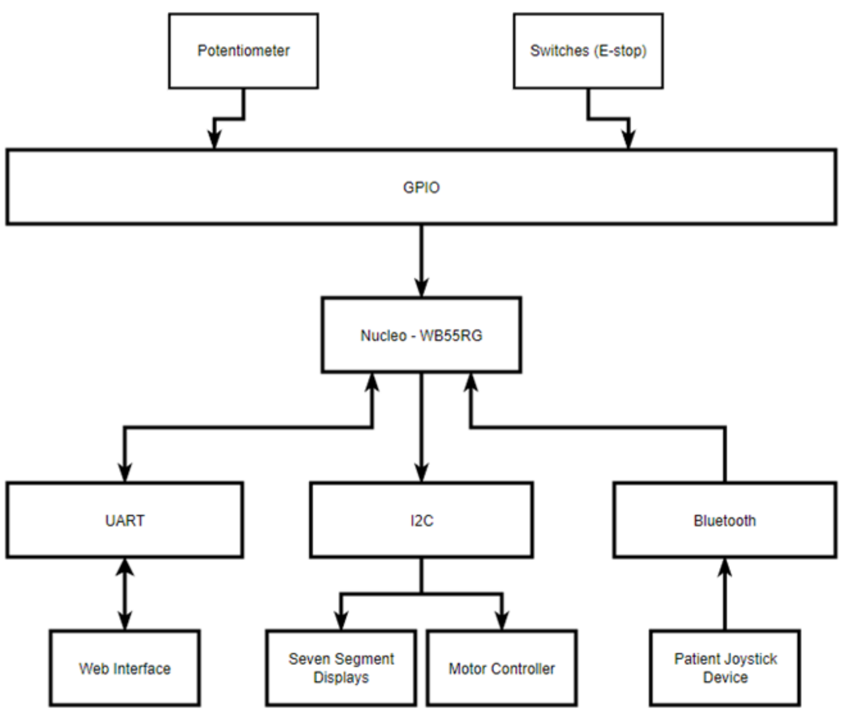

Figure 9: Internal Communication Architecture of NASA Vestibular Chair

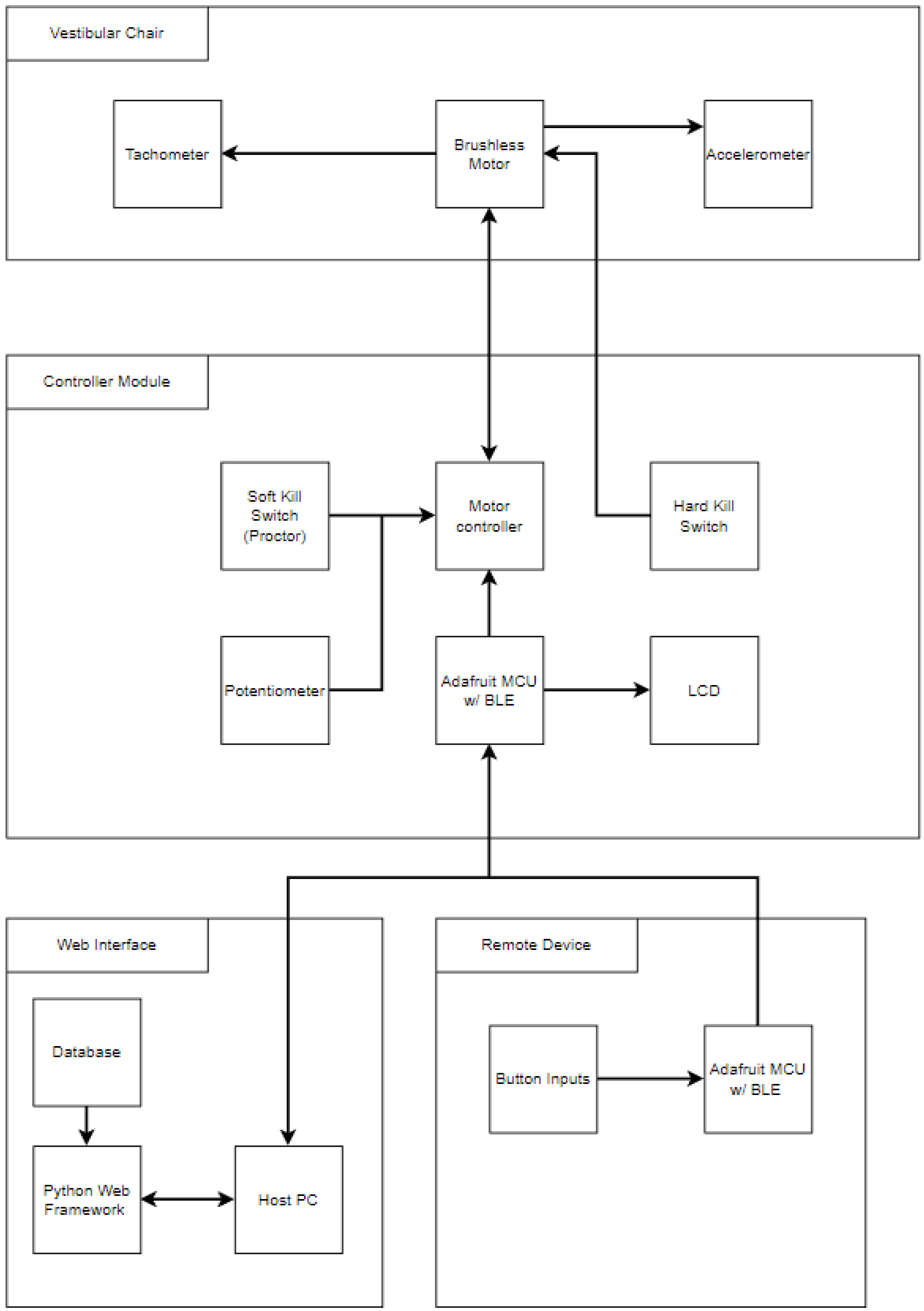

Figure 9.1: Detail view of Internal Communication Architecture

5.2.1 Motor Controller Interface

The Pololu Simple Motor controller interfaces with the control module via the I2C protocol. Details of the I2C specification can be found here. Pololu provides a reference manual on how to properly initialize and use the motor controller. For our system, the motor controller will be set to the Binary mode to process all commands over I2C. All commands sent to the motor controller shall follow the compact protocol which places the command in the first byte sent and the following commands indicate parameters for the commands. The graphic below indicates the packet format of the command in a binary representation.

command byte (MSB set)	[data byte 1]	[data byte 2]	...	[data byte n]
1XXXXXXX	[0XXXXXXX]	[0XXXXXXX]	...	[0XXXXXXX]

Figure 10, I2C, Compact Protocol Packet

A list of valid commands is also included in Pololu's documentation. The commands include but are not limited to: "motor reverse", "motor forward", "set motor limit", and "set current limit". Commands are distinguished by sending bytes with a decimal value between 128 and 255 while data bytes are any value between 0 and 127.

5.2.2 Web Interface

The web interface component of the system will utilize the UART protocol for transmitting data between the host PC and the control module. A formal standard or Interface Control Document (ICD) has not been defined at this time due to the current objective of establishing simple hardware control. However, after reaching this milestone, it shall be included in the next version in a later version.

5.2.3 Patient Input Device

The patient input device will be utilizing the same development board that the control module is based on. Subsequently, the patient remote device is Bluetooth-capable and supports the BLE specification. Furthermore, the Adafruit Feather nRF52480 Express has support for this protocol and sufficient documentation to utilize it. A formal standard or Interface Control Document (ICD) to define the transfer of data has not been defined at this time due to the current objective of establishing simple hardware control. At the time of writing, that objective has just been achieved. For the final release of this document, a formal interface standard will be included.

6 SYSTEM INTEGRITY CONTROLS

Due to the nature of the interaction between the user and the chair itself, there is little question of integrity for the system due to it not being able to be used in a way that would cause a leak of sensitive information. The most likely issue would be a misuse of the chair itself, such as spinning it at velocities it is not built for. However, there are built-in controls and fail-safes to avoid such outcomes.

7 Appendix A

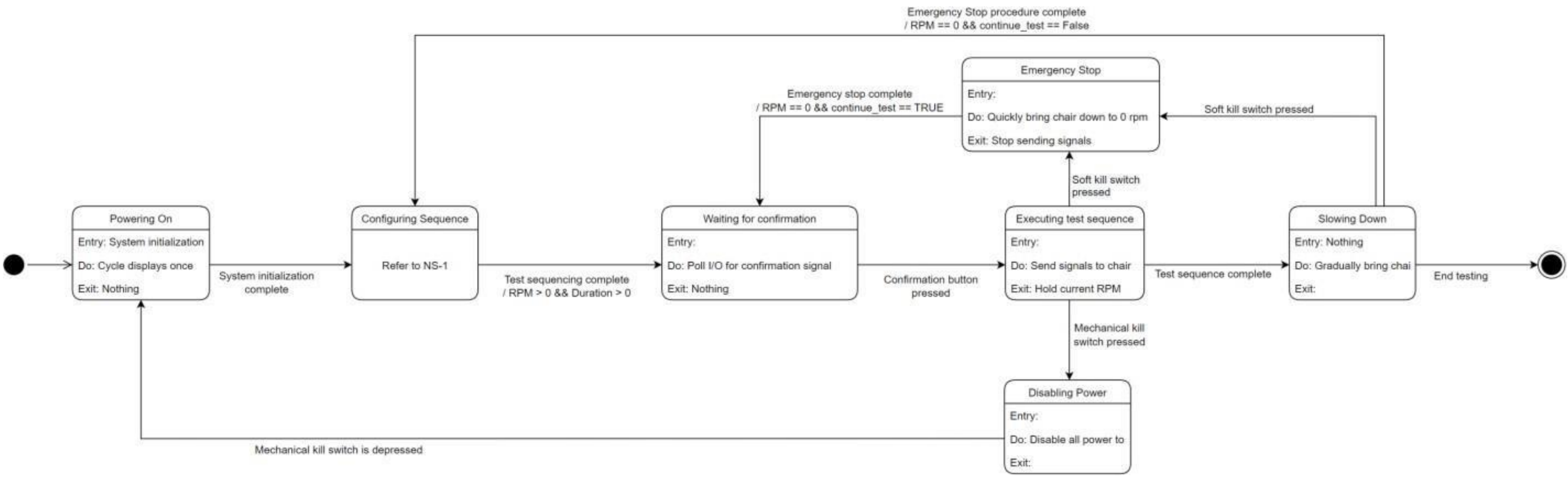

Figure 2: State Transition Chart for NASA Vestibular Chair

8 Appendix B

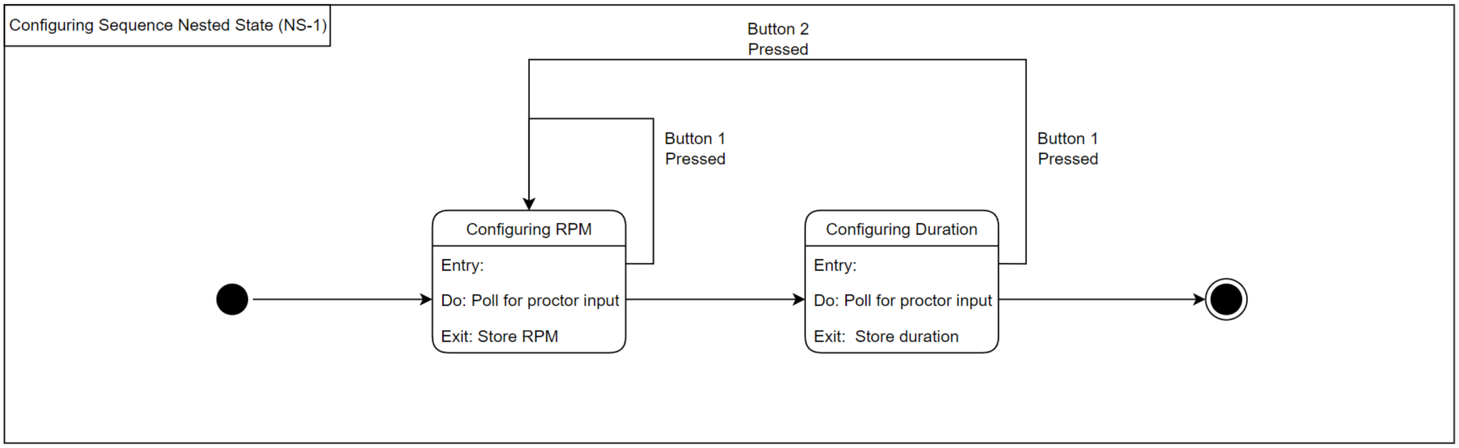

Figure 2.1: Nested State 1 for State Transition Chart (NS-1)