Air Mouse

Wireless translation of hand gestures into two and three-dimensional mouse movements

Prepared by:

John Chen

Wania Zahra

Nicholas Paluch

Hammed Ayoade

Ibrahim Idris

Simran Parmar

Final report submitted in partial satisfaction of the requirements for the degree of

Bachelor of Science

in

Electrical and Computer Engineering

in the

Faculty of Engineering

of the

University of Manitoba

Academic Advisor:

Dr. Ken Ferens

Winter 2018

Abstract

With the advent of smartphones, laptops, computers and other forms of technology virtually becoming an indispensable part of everyday life, user interaction and communication with such devices lies at the helm of progress. Air Mouse endeavours to enhance the classic user experience of the 2D mouse control by giving them the freedom to operate wirelessly, in air, through hand gestures.

The Air Mouse development can be broken down into two major categories: hardware design and software programming. At the heart of the project lies the nRF52 microcontroller which houses the Bluetooth LE Raytac module essential for pairing the device. This microcontroller is mounted on a cotton glove that is worn by the user. The glove's peripheral sensors read hand gestures which are converted by the microcontroller into 2D mouse functions. The components of the Air Mouse were individually tested on a breadboard and eventually integrated onto the glove fabric to produce the final prototype.

This project can be deemed a success as it was able to meet all the proposed specifications. The final prototype can perform all basic mouse functions such as scrolling, left and right clicks, hold and drag items where applicable, like a conventional mouse through a finite state machine. A small vibration motor between the index and middle fingers provides the user with haptic feedback upon a successful clicking action. The battery is able to support the Air Mouse operations for 2 hours and 20 minutes which is higher than the proposed requirement of 2 hours. In line with the proposal, Air Mouse also comes equipped with a laser diode to aid in presentations with a pressure sensitive control. The glove fabric along with the careful selection and placing of components also ensures that it is light and ergonomically designed making the Air Mouse a pioneering technology.

Acknowledgements

We would like to thank our academic advisor, Dr. Ken Ferens, P.Eng., for approving our initial idea of an Air Mouse as well as for his support and guidance throughout the project.

Thanks to Dr. Derek Oliver, P.Eng., for his course expertise and valuable feedback throughout the year. The guidance we received will certainly help us succeed in our professional careers.

We highly appreciate the time and support devoted to the team and its progress by Mr. Daniel Card, P.Eng. His advice on the technical design elements and critical reasoning has aided us all in growing into well-rounded and practical-thinking, future engineers.

Thanks to Ms. Aidan Topping for her help with technical documentation. This report and all of the previous Design Review reports would not have been able to come together as well as they did without her feedback.

Thank you to Mr. Glen Kolansky for his assistance in organizing our budget throughout the approval process and his continued support with components and devices instrumental in the completion of this project.

Thanks to Mr. Sinisa Janjic for the help in obtaining all of the parts needed as required to build the Air Mouse.

We would also like to extend our gratitude to Mr. Darcy Cook, and Mr. Tim Hoeppner from JCA Electronics for sharing their knowledge and references regarding IMUs and centimeter precision displacement sensing.

Contributions/Division of Work

Milestone	Task	MEMBER(S) RESPONSIBLE					
		John Chen	**Ibrahim Idris**	**Wania Zahra**	**Nick Paluch**	**Hammed Ayoade**	**Simran Parmar**
Design & Simulation	Research and Literature Review, Budget Finalization , Hardware Selection, Design Feasibility study	x	x	x	x	x	x
Prototyping & Testing	Assemble physical Wiring of Components	x	x	x	x	x	x
	Application: Sensor data, Bluetooth, transmit mouse command over Bluetooth	x		x			x
	Testing Toolchain for Development Board	x		x	x		
	Range and Battery Life Testing	x	x		x		x
	Review and Troubleshooting	x	x	x	x	x	x
Integration & Validation	Calibration and Device Compatibility Test with Sensor and Position Control			x	x	x	
	Final glove: Mounting Setup on Glove, Calibration	x	x	x	x	x	x
	Checking the the project meets performance Metrics	x		x			
Final Report Writing	Drafting, editing, revising and documenting the report	x	x	x	x	x	x

Table of Contents

List of Figures

List of Tables

Nomenclature

Abbreviation	Description
2D	Two (2) Dimensional
3D	Three (3) Dimensional
ADC	Analog to Digital Converter
API	Application Programming Interface
BLE	Bluetooth Low Energy
Bluetooth LE	Bluetooth Low Energy
EMA	Exponential Moving Average
FreeRTOS	Free Real Time Operating System
FSM	Finite State Machine
FSPL	Free Space Path Loss
GAP	Generic Access Profile
GATT	Generic Attribute
GPIO	General Purpose Input Output (Pin)
HID	Human Interface Device(s)
I2C	Inter-Integrated Circuit
IEEE	Institute of Electrical and Electronics Engineers
IMU	Inertial Measurement Unit
ISM	Industrial, Scientific and Medical (Radio Bands)
LDO	Low Dropout (Voltage Regulator)
LiPo	Lithium Polymer
RF	Radio Frequency
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver-Transmitter

Chapter 1. Introduction

1.1 Background

Electronic devices such as laptops, computers and smartphones take user input in various forms. The conventional 2D mouse is one of the most commonly used aids for this. With Air Mouse, the team plans to devise an innovative way to control mouse movements wirelessly to enhance the user experience of interacting with technology.

1.2 Existing Mouse Technologies

By convention an optical/laser computer mouse has always been a hand held device that needs a flat surface in order to move and operate the cursor on a display. This method of control restricts the user to operating in locations very close to the device and the need for a smooth platform which can be very inconvenient in certain situations such as presentations and screen navigation on smartphones. The idea of a wearable, inertial mouse system was first explored by Patent US4787051[5]. Initial designs and prototypes used RF for the wireless communication protocol.

The Air Mouse design aims to simplify and improve the quality of the design by making use of Bluetooth LE that eliminates the need of a base station, thereby, reducing hardware and allowing the user to use a single mouse for multiple applications and devices. It incorporates hand gestures that are intuitive and/or easy to remember and aid in the operation of technological devices to a degree of greater freedom and flexibility.

Chapter 2. Purpose and Design

This section will discuss the objectives of this project and will give a general overview of the system setup and topology. It will also provide an insight into the simulations and testing stages that led to the final prototype.

2.1 Project Purpose

The purpose of this project was to make a user-friendly glove that is able to control a mouse pointer, wirelessly, through hand gestures. The scope of the project encompassed the design and implementation of a wearable glove technology with Bluetooth LE [7] as the communication protocol. Thus, making the Air Mouse compatible for pairing with any electronic device that supports this standard.

2.2 System Overview

The project was divided into two main categories: software development and hardware design. The hardware system comprises mainly of the microcontroller, flex sensors, battery, NeoPixel, pressure sensitive sheets, laser diode and a vibration motor. Each of these components were carefully chosen to actualize the features of a conventional mouse, such as scrolling, clicking and 2D spatial movements, in air. The software programs were developed in conjunction with the hardware design to meet the proposed requirements of the project. Table 2.3 enlists these requirements.

2.3 System Performance Metrics

The following table contains the proposed requirements of the project for it to be deemed a success:

ID	FEATURE	RANGE	COMPLETED
1	All components must fit on all standard glove sizes (XS-XL)	Yes/No	Yes
2	Design is able to send/receive information using the Bluetooth LE standard	Yes/No	Yes (nRF52 Bluetooth Module)
3	Design can move mouse cursor on a computer using hand movements	Yes/No	Yes (controlled by BNO055) *Pointing Mode Precision* Horizontal: 29 pixels/degree Vertical: 16 pixels/degree *Moving Mode Precision* Both: 114 pixels/(degree*second)
4	Design can perform mouse clicks on a computer using hand movements	Yes/No	Yes (Left and Right Click triggered by Flex)
5	Design can scroll/pan on a computer using hand movements	Yes/No	Yes (controlled by BNO055)
6	Design is able to be used in a 3D axis application	Yes/No	Yes (Windows 10 Application)
7	Design is powered by a portable power source	Yes/No	Yes (3.7V 110mAh LiPo battery)
8	User is able to turn on/off device to reduce power consumption	Yes/No	Yes (Switch)
9	Battery Life Specifications	> 2 hours	Yes (2h 20m)
10	Battery State of Charge Indicator	Percentage, or Segments/Low Warning	Yes (Green, Yellow, Red, with varying brightness)
11	Haptic feedback on mouse clicks	Yes/No	Yes (3V motor)
12	Hand gestures to enable laser pointer aid for presentations	Yes/No	Yes (triggered by pressure sheets)

Table 2.3-1: Performance Metrics

2.4 Methodology

2.4.1 Requirements of Development Board, Bluetooth LE Module, and Sensors

The Development Board, Bluetooth LE module[18], and sensors each have requirements for the successful completion of this project. The components along with their criteria for selection is listed below:

Development Board

- Microcontroller must be powerful enough to parse sensor data, and issue commands to the Bluetooth module without noticeable delay to the user input.
- Development board chosen should allow for connection of an external power supply, either through a connector, or an easily accessible pin/solder pad.
- Microcontroller must contain sufficient peripherals for this application. For this project, it must have at least one (1) set of SPI/I^2C/UART pins, and at least five (5) analog input pins, for each finger.
- The development board should not be overly large or heavy that it would cause discomfort to the user when placed on the glove.

Bluetooth LE Module[7]

- The Bluetooth LE Module must be able to communicate easily with the microcontroller using a standard protocol, such as UART or I^2C.
- The Bluetooth LE Module should already have an easy to use Bluetooth LE API that supports the GATT HID Service in order to simplify the development process.
- Again, the Bluetooth LE Module should not be overly large or heavy that it would cause discomfort to the user when placed on the glove.

Based off of these requirements, the Adafruit Bluefruit nRF52 Feather Development Board[13] was selected for this design. The Adafruit Bluefruit nRF52 is a single, small, and lightweight board that contains both the microcontroller and Bluetooth LE module, and has support for the Adafruit Bluetooth LE APIs. For technical details of the board, please refer to section 3.1.

Sensors[1]

- The motion sensor must be able to read spatial displacement in the order of centimeters, and angular displacement in order of degrees.

- The motion sensor should give these spatial and angular displacement readings in all three Cartesian axes (x, y, z).
- The sensors must be able to translate a hand gesture to a binary (true/false) output, if it is performed.
- The sensors must communicate via a standard protocol so that they can be read by the selected development board.

To satisfy these requirements, the Bosch Sensortec BNO055 IMU sensor[1] was selected to read spatial and angular displacement values. Specifically, the Adafruit BNO055 module was selected because it uses the I^2C protocol, and comes with an API that allows Arduino based boards to communicate with the sensor easily. Flex sensors were also selected to translate hand gestures into binary output. The microcontroller can be reprogrammed to increase or decrease the sensitivity based on the user. If required, the sensors can also be used as an analog input. Pressure sensitive sheets were also included as a means of control for the laser diode.

2.4.2 Movement Based on Euler Angle Orientation

One of the major requirements for the Air Mouse to be truly functional in air was to devise a way of translating spatial and rotational hand movements into 2D mouse movements. This was accomplished by using the BNO055 due to its ability to provide user hand displacement information in the form of Euler angle displacements in the x, y and z planes. The BNO055 is discussed in detail in section 3.2 of the report[1].

2.4.3 Finite State Machine

The concept of a finite state machine was introduced keeping in mind the various forms of receiving user input from hand gestures. Finite state machines are able to control execution flow by putting the device into one or more states as necessary[11]. The functions required of the Air Mouse could also be broadly classified into states such as scrolling, left click, right click etc. making it a perfect fit for the project.

2.4.4 Checking I^2C Packets using Saleae Logic Analyzer

The Saleae Logic Analyzer was used to examine the I^2C packets to understand their functioning:

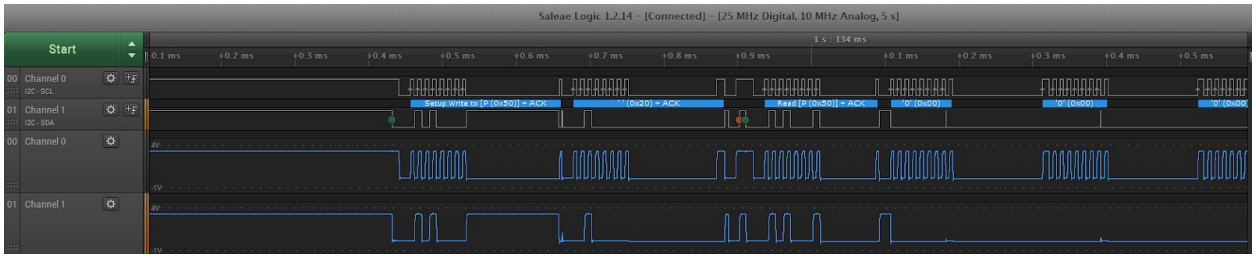


Figure 2.4-1: I²C communication between the nRF52 (Master) and BNO055 (Slave)

In the figure above, the nRF52 requests data from the BNO055 using its slave address 0x50[13],[1]. Note that the slave address, 0x28, provided by Adafruit is valid for 7 bit addressing. Bit shifting this 7 bit address to the left by 1 digit yields the correct 8 bit address.

2.5 Topology - Right Hand Glove

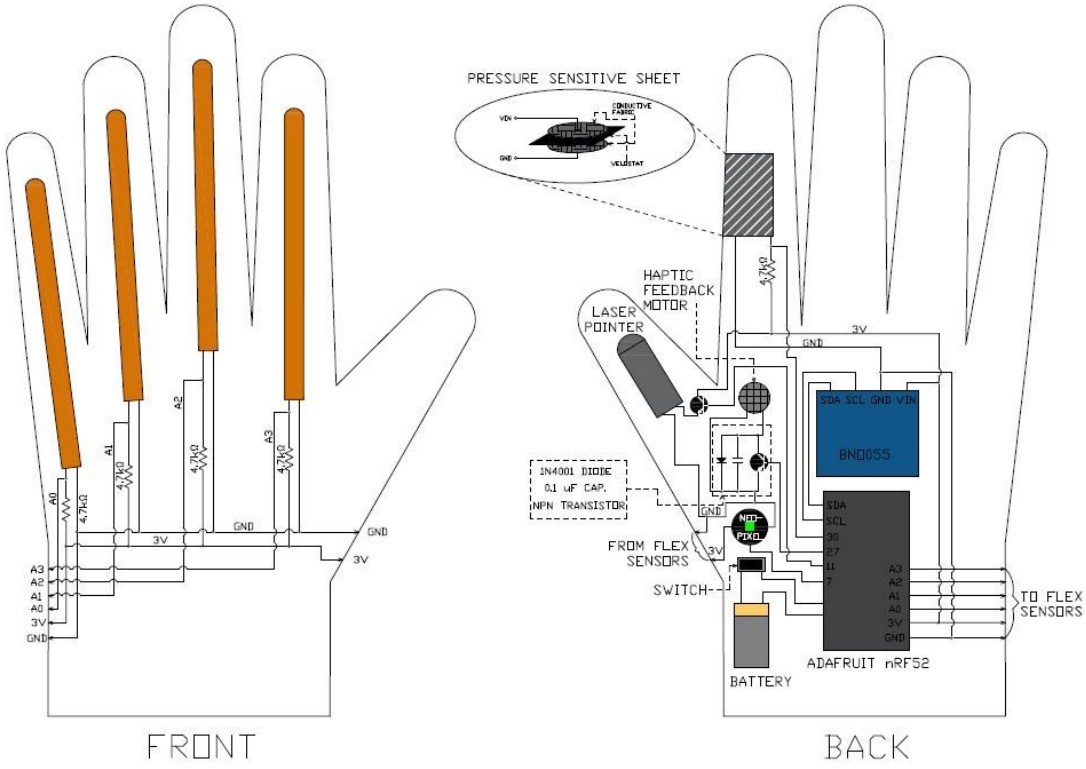


Figure 2.5-1: AutoCAD Drawing of Air Mouse components

The glove material was chosen using the following criterion:

- Lightweight (< 50g) to allow for unstrained hand movements.
- Antistatic Fabric to ensure against static electricity buildup.

- Air-permeable to reduce sweating or related discomfort.

A ready-to-wear right-hand glove made of cotton was chosen in standard unisex, medium size, for the prototype.

The Bluefruit nRF52 board was positioned and sewn on the lower middle portion of the top side of the glove such that the micro-USB connector is oriented towards the user for recharging the battery or connection of any external sources. This reduced the rigidity of the glove as well as made all connections to and from the microcontroller easier.

The BNO055 sensor receives the hand orientation and tilt information based on angle displacement from its initial position. Therefore, to allow for maximum sensitivity to hand movements and gestures, it was mounted vertically and sewn in the upper middle part (right above the microcontroller) of the glove.

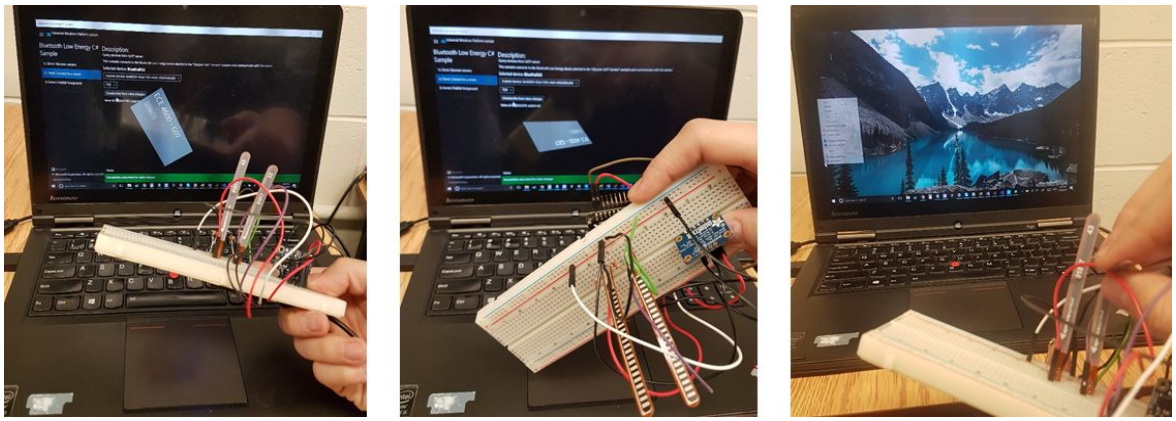
The flex sensors were placed on the inner sides of the four fingers of the glove using glue for more flexibility and to reduce the amount of pressure exerted on the fingers.The battery was mounted next to the charging port of the microcontroller, i.e, the lower left side of the glove. A slider switch was placed close to the battery's location for ease of turning the glove on or off.The NeoPixel was placed right above the switch to indicate the remaining battery life.

A pressure sensitive sheet patch was sewn on the middle left edge of the index finger to act as a control for the laser diode. The laser diode was, in turn, positioned on top of the thumb such that it points straight when the user presses its pressure-sensitive sheet control. A mini disk vibrator was placed on the front of the glove just below the knuckle of the index finger to receive feedback in the form of small vibration on successive left or right click.

2.6 Simulation and Prototyping

2.6.1 Breadboard Prototype

The design was initially built on a breadboard to test the basic functionality of the glove such as correct orientation detection, left and right clicks. Figure 2.6-1 shows the test results being carried out on Microsoft Windows 10 BLE application.



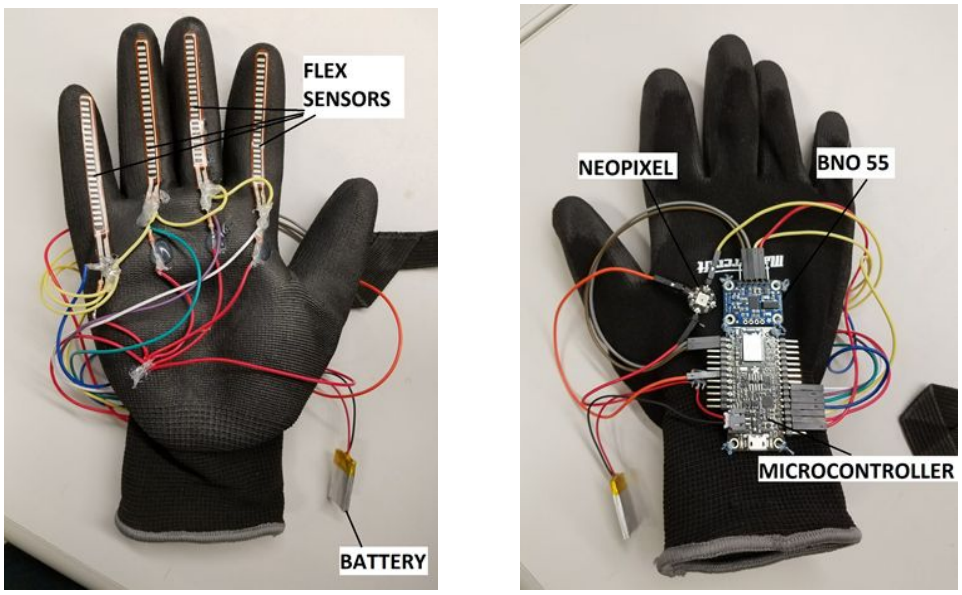
(a) (b) (c)

Figure 2.6-1: Breadboard prototype used in testing (a) Tilting Right (b) Tilting 180 degrees down (c) Right Clicking using Flex Sensor

The breadboard prototype allowed for the testing of the components individually. This was particularly useful in writing the initial code, determining the threshold values and capabilities of each of the chosen parts. Critical design decisions such as placing of components and their intended use were made at this stage.

2.6.2 Initial Glove Prototype

The first prototype was built on a 90% nylon glove. Components were positioned on the glove as shown in Figure 2.6-2. The first design layout was wired using jumper cable connections in order to provide flexibility of rewiring while testing. The design satisfied the working requirements but since nylon induces static charge, the same layout was carried on to the final prototype with an air-permeable, cotton fabric.



(a) (b)

Figure 2.6-2: Initial Air Mouse prototype on nylon glove (a) Back side (b) Front side

2.6.3 Final Prototype

(a) (b) (c)

Figure 2.6-3: Final Air Mouse prototype with components placed. (a) Back side (b) Front Side (c) With slip-on

The figure above shows the final assembled Air Mouse device for the project. A cotton slip on was designed that buttons up around the glove leaving the NeoPixel, the switch and the pressure sheet exposed for user's use. The BNO055 has an effective precision of 0.01 radians, or 0.57 degrees for the converted Euler angle representation in each of the x, y and z axes.

In the "Moving" control mode, the sensitivity of the cursor movement is set to the same in, both, the horizontal and vertical axes. The user is allowed a precision of 100 pixels/radian, or 1.72 pixels/degree for the step size of the cursor movement at each update interval. The maximum step size allowed is set to exactly 100 pixels to prevent unwanted large cursor movements. At a theoretical update interval of ~15ms, the precision of the cursor speed is given by 114 pixels/(degree*second).

In the "Pointing" control mode, with a screen resolution of 2560x1440, and +/-45 degree range of movement, the user is allowed a precision of 29 pixels/degree in horizontal cursor movement, and 16 pixels/degree in vertical cursor movement.

Chapter 3. Hardware

3.1 Adafruit Bluefruit nRF52 Feather

The Adafruit Bluefruit nRF52 Feather is a Bluetooth Low Energy development board based off the Nordic Semiconductor nRF52832 System on a Chip[12],[13]. Earlier Bluetooth development boards had separate microcontrollers that send commands to standalone Bluetooth modules. On this board, there is a single module, the Raytac MDBT42Q, which handles all Bluetooth LE communications as well as serves as the microcontroller[14]. The nRF52 Feather is programmed using the Arduino IDE and user sketches run in parallel with the low-level Bluetooth LE stack developed by Adafruit. The board uses FreeRTOS as the task scheduler which allows both code branches to run asynchronously at the same time[6],[13].

The Feather nRF52 was chosen as it comes with sufficient GPIO pins, analog pins and peripherals such as I^2C for our application in the project[13]. It is also very powerful as it uses an ARM Cortex M4F, clocked at 64MHz. For the technical details of the product specification, as well as the board layout on the Adafruit website, and in the supporting documents for the modules used[4], please refer to the Appendix.

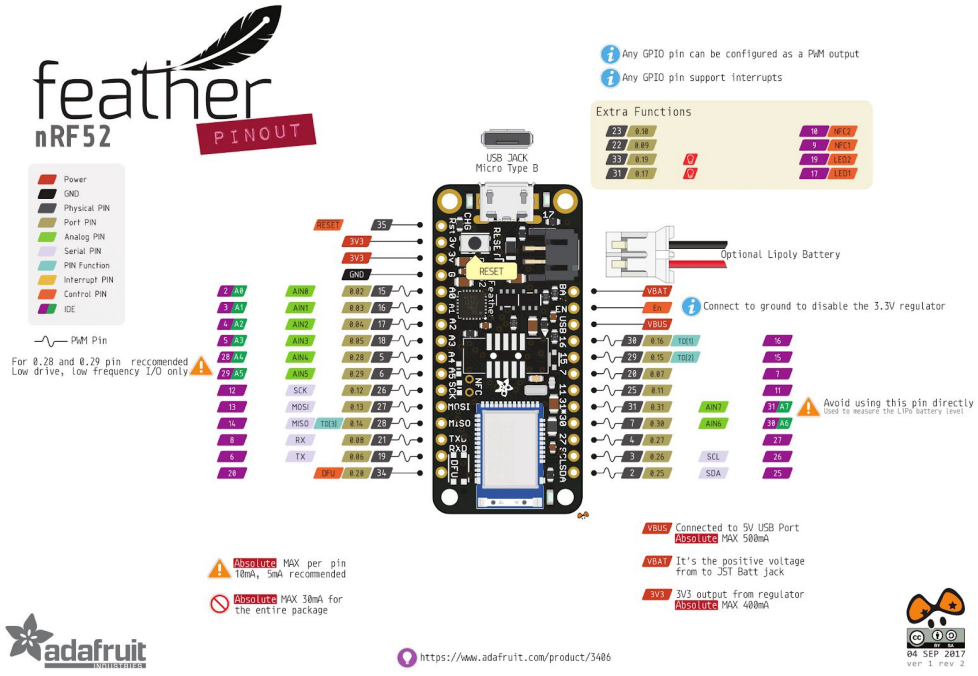


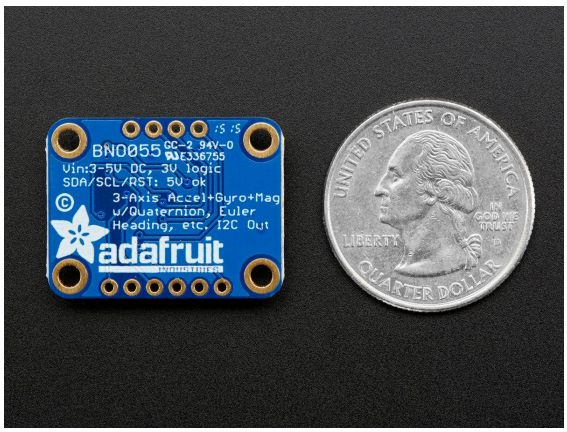
Figure 3.1-1: Adafruit Bluefruit nRF52 device pinout.

nRF52832 Technical Details:

- ARM Cortex M4F (with HW floating point acceleration) running at 64MHz
- 512KB flash and 64KB SRAM
- Built in USB Serial converter for fast and efficient programming and debugging
- Bluetooth Low Energy compatible 2.4GHz radio (Details available in the nRF52832 product specification)
- FCC / IC / TELEC certified module
- Up to +4dBm output power
- 1.7v to 3.3v operation with internal linear and DC/DC voltage regulators
- 19 GPIO, 8 x 12-bit ADC pins, up to 12 PWM outputs (3 PWM modules with 4 outputs each)
- Pin #17 red LED for general purpose blinking
- Power/enable pin
- Measures 2.0" x 0.9" x 0.28" (51mm x 23mm x 8mm) without headers soldered in
- Light as a feather - 5.7 grams
- 4 mounting holes
- Reset button
- Optional SWD connector for debugging

3.2 Adafruit BNO055 - Absolute Orientation Sensor

3.2.1 Background



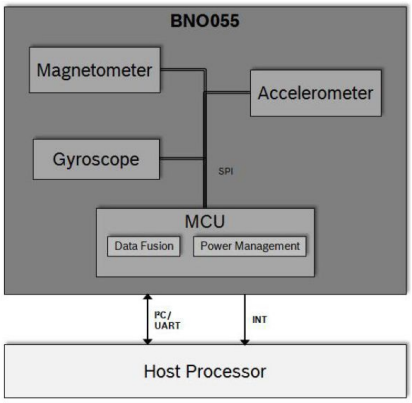


Figure 3.2-1: BNO055 size in comparison to a quarter

Figure 3.2-2: BNO055 system architecture

The BNO055 was selected as the motion and orientation sensor[1]. It constitutes of three triaxial sensors, an accelerometer, a gyroscope and a magnetometer that simultaneously measure tangential acceleration, rotational acceleration and local magnetic field respectively. It also has an inbuilt M0+ microprocessor that takes the data from the sensors and uses a fusion algorithm (by Bosch Sensortec) to give the absolute orientation of BNO055 in space.

Sensors	Parameters	Value
Accelerometer	Power Mode	NORMAL
	Range	+/- 4g
	Bandwidth	62.5Hz
	Resolution	14 bits
Gyroscope	Power Mode	NORMAL
	Range	2000 °/s
	Bandwidth	32Hz
	Resolution	16 bits
Magnetometer	Power Mode	FORCED
	ODR	20Hz
	XY Repetition	15
	Z Repetition	16
	Resolution x/y/z	13/13/15 bits

Table 3.2-1: Default sensor configuration of BNO055

A host microcontroller can request any or all of the data from the sensors (accelerometer, gyroscope, and/or magnetometer) in non-fusion mode or request absolute and relative orientation (angles or quaternions) in fusion mode.

BNO055 has the following dimensions:

- Length: 5.20mm
- Width: 3.10mm
- Thickness: 0.13mm

The powerful sensing abilities and compact size made the BNO055 a perfect fit for the Air Mouse[1]. The BNO055 is responsible for determining the hand orientations in space and relays this information to the microcontroller.

3.2.2 Configuration

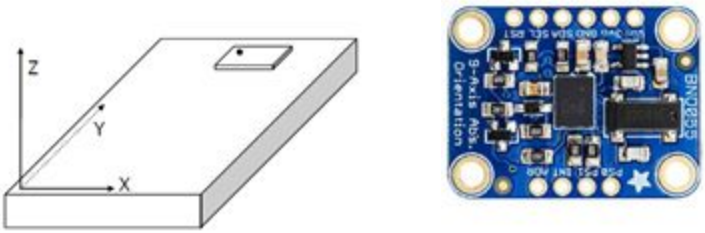


Figure 3.2-3: BNO055 positioned on the XY plane

The BNO055 is mounted on the glove and configured to the reference axis as shown in Figure 3.2-3.

The following configurations were set while initializing the BNO055 settings appropriate for use in mouse function[1]:

- Power mode is set to normal: all sensors are always switched ON.
- Operation mode is set to NDOF Fusion mode: Absolute orientation data is calculated quickly from all three sensors giving a high output data rate.
- The Fast Magnetometer calibration is turned ON giving quick calibration of the magnetometer and high output data accuracy.

VDD (or Vin pin) is the main power supply for the internal sensors and takes upto 4.2V maximum. The BNO055 supports bi-directional I²C interface with a microcontroller. Figure 2.3-4 shows the I²C bus connecting serial clock pin SCL and serial data I/O signal lines to the nrf52.

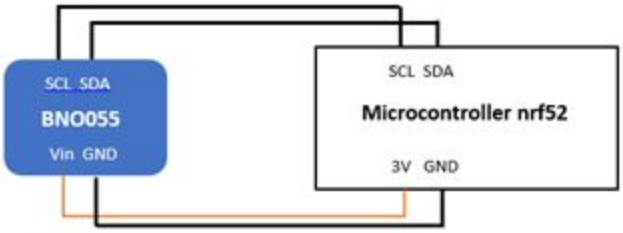


Figure 3.2-4: SPI bus between nRF52 and BNO055

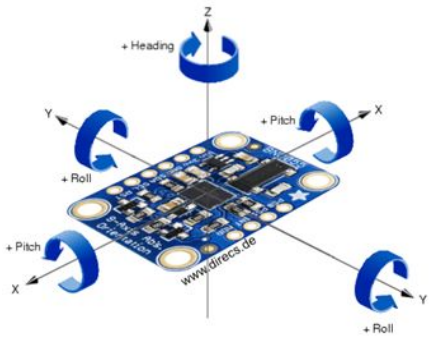


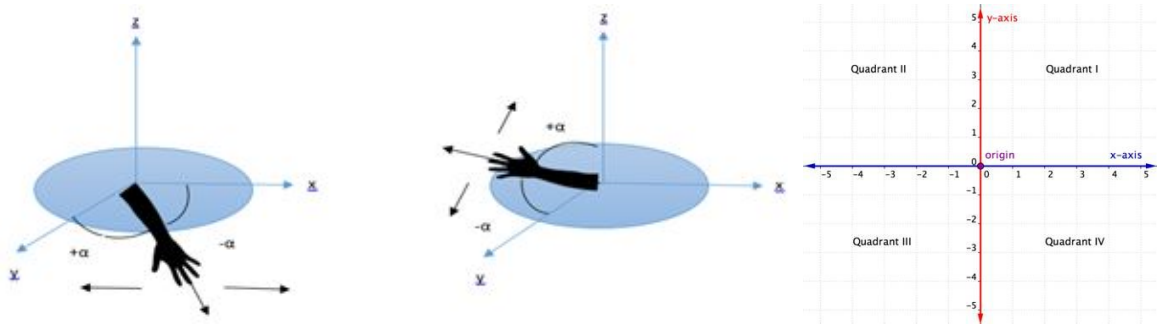
Figure 3.2-5: Euler angles on BNO055

The BNO055 filters and synthesizes the sensor data on its built in microprocessor and gives the relative orientation as a vector of quaternions[1]. However we wanted the return in the form of Euler angles, since it is the easiest to visualize and manipulate, therefore we used a function that converts quaternions to three euler angles with units of radians. The euler angle format is based on convention of rotation angles for roll, pitch and heading as shown in Figure 3.2-5 and explained in Table 3.2-2.

Rotation angle	Range (Android format)	Range (Windows format)
Pitch	+180° to -180° (turning clockwise decreases values)	-180° to +180° (turing clock-wise increases values)
Roll	-90° to +90° (increasing with increasing inclination)	
Heading / Yaw	0° to 360° (turning clockwise increases values)	

Table 3.2-2: Rotation angle conventions

For purpose of using mouse on a 2D plane, only euler angles formed in x and z direction are used for calculation of position. Initially a starting position of the hand is stored in the memory of the nRF52 as reference angle. Updated Euler angles are continually read from the BNO055 and the deviation of the updated angle from the starting angle is returned as the relative position of the hand. Everytime the user wants to set a new reference position, they have an option of recalibration that is programmed in FSM (refer to section 4.3.1.3). As seen in figure 3.2-6 (a) and (b) the angle that the hand forms on the plane changes when moving from one quadrant to the other. To take this change into consideration a necessary condition was programmed on the microcontroller that checks whether the hand has crossed PI radians or not[1],[13],[15].



(a) (b) (c)

Figure 3.2-6: Calibrating the origin to measure angular displacement (a) Reference in Quadrant I of x-y plane (b) Reference in Quadrant II of the x-y plane (c) Four Quadrants of a two dimensional axis system

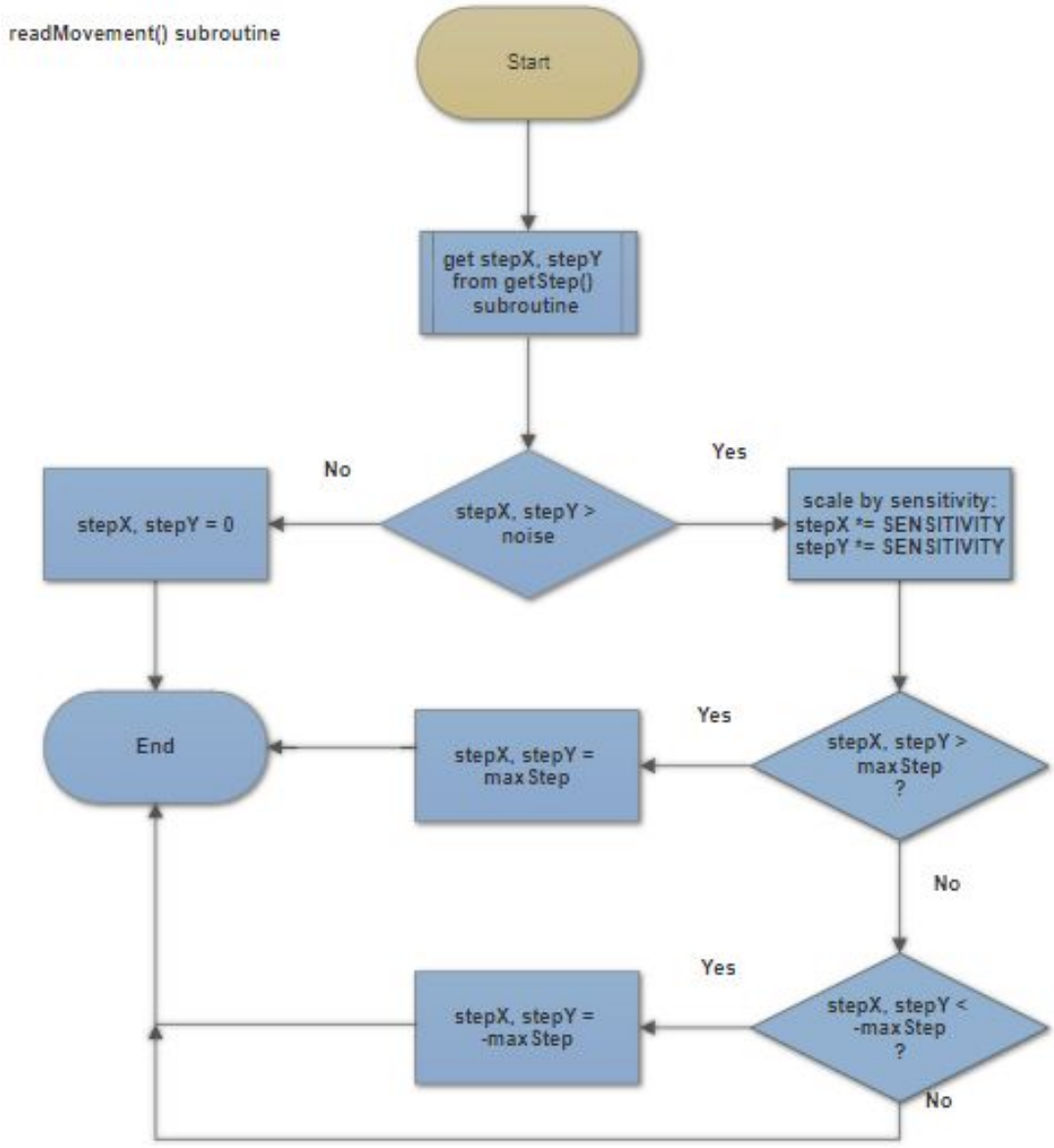


Figure 3.2-7: Flowchart for readMovement() subroutine

Flowchart in figure 3.2-7 shows the algorithm the microcontroller uses to determine how many pixels the mouse cursor moves on changing the hand angle. The maximum number of pixels the cursor can move was decided to be 10 and the sensitivity was selected to be 100 after running several tests on a 1440x2560 resolution.

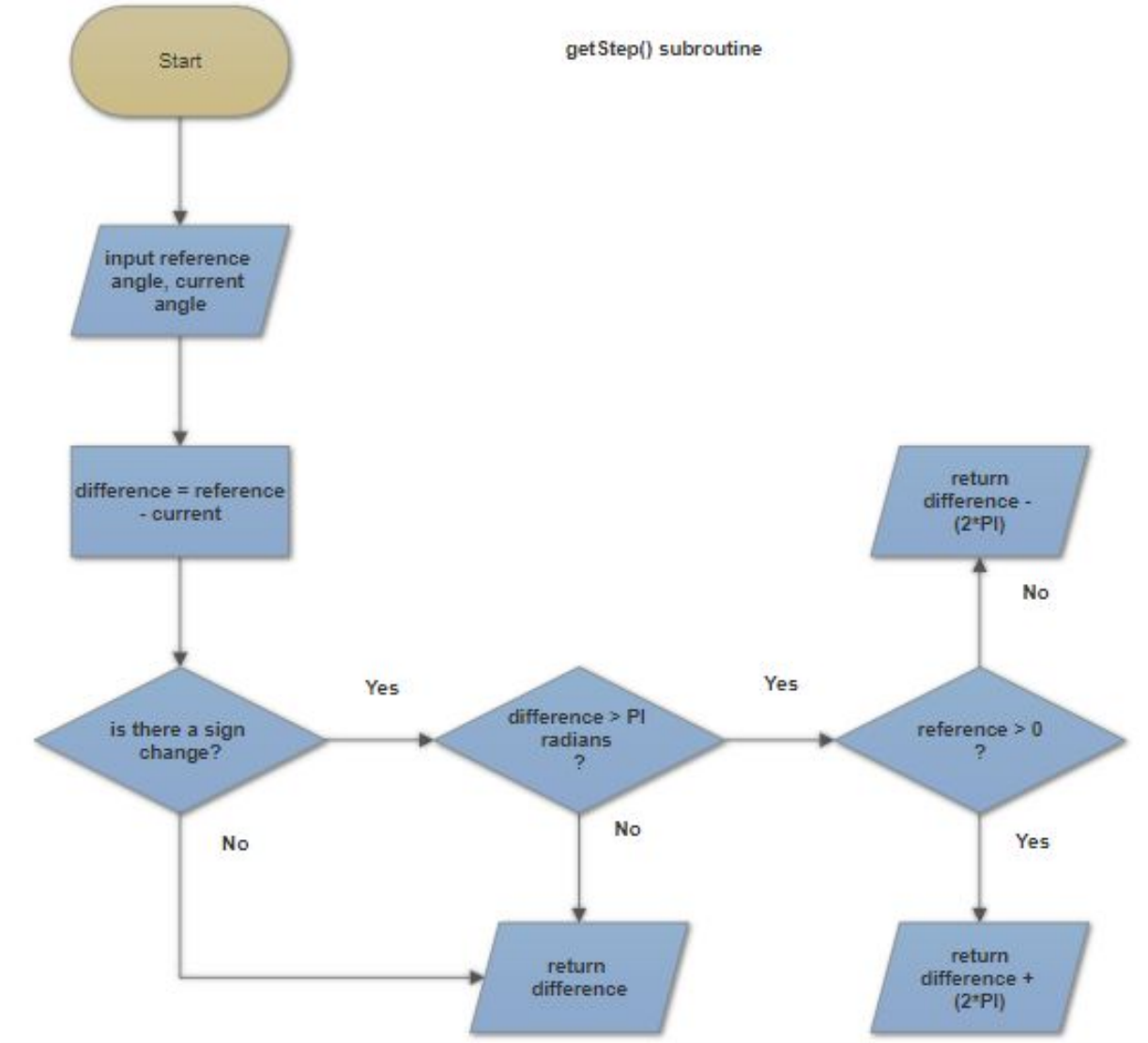


Figure 3.2-8: Flowchart for getStep() subroutine

The flowchart in figure 3.2-4 describes how the microcontroller is programmed to calculate the angle the hand moves based on the reference angle. The calculation used to determine the angle offsets can be found in the appendix section.

3.3 Flex Sensors

Figure 3.3-1: Flex sensor size compared to a quarter[4]

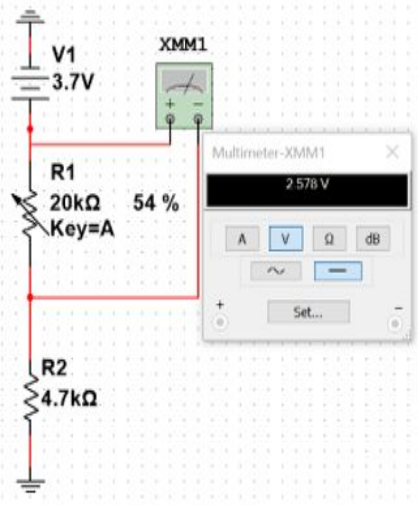


Figure 3.3-2: Multisim model of Flex Sensors

The flex sensors used for the project are RB-spa-989 with a length of 2.2 inches (5.6cm). Each of the flex sensors used is rated flat resistance of 10k in their idle, unflexed state. At the prototyping stage, the resistance was tested to be about 58 to 110KΩ when fully bent[25]. The variation in resistance depends on the degree of flexion from 0 to less than 90 degrees. The resistance and voltage values obtained for the flex sensors in the idle state on different fingers is shown in Table 3.3-1. The operation of the flex sensors can be understood as that of a voltage divider circuit wherein the output voltage is directly proportional to the degree of flexion.

3.3.1 Testing and Simulation

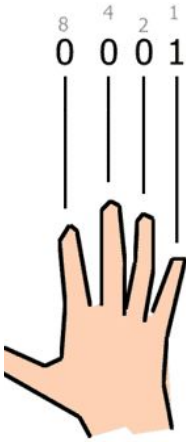
The model of flex sensor shown in figure 3.3-2 was used to determine the voltage draw. Each of the flex sensors were then placed on a breadboard to test for threshold voltages and were used in the preliminary breadboard prototype design. These values were re-recorded for the initial nylon glove prototype and the final cotton glove prototype in order to account for any changes or deviations in these values. As a result of their consistent results for threshold values despite regular bending actions made them durable for their purpose. The final results are shown in table 3.3-1.

Pin	Label	Flexed Resistance R (Ω)	Threshold voltage (V)
0	Pinky	10.83K	2.58
1	Ring	11.40K	2.62
2	Middle	11.11K	2.60
3	Index	15.76K	2.85

Table 3.3-1: Voltage measurements for the flex sensors on each finger.

The flex sensors are placed on each finger of the glove barring the thumb, this was due to size of the flex sensor relative to the finger and ease of flexing to obtain an effective threshold voltage value. The thumb is, however, used to support other hardware like laser diode and to control its pressure sensitive switch. A 4.7KΩ pull-up resistor was used to make the input pin read a high state when the sensor is not flexed to protect the battery from draining when the glove is not in use.

3.3.2 Finite State Machine



1 – Flexed
0 - Unflexed

- MODE_POINTING = B0000,
- MODE_LEFTCLICK = B1000,
- MODE_RIGHTCLICK = B0100,
- MODE_CALIBRATION = B0010,
- MODE_MOVING = B0111,
- MODE_SCROLLING = B1111

Figure 3.3-3: Mapping the Finite State bits onto the user's fingers[26].

Note: The default mode is B0000, with none of the flex sensors bent. Please refer to Section 4.3.1 for the implementation of these modes.

The flex sensors allowed the implementation of basic 2D mouse functions such as clicking, scrolling and cursor movements in the form of a finite state machine. The following describes the different executable state modes:

- MODE_POINTING (B0000): In the Pointing Mode, the distance that the cursor moves is set by the angular displacement of the user's hand. The output of the angle calculated by BNO055 from the user's gesture is stored in the point X and Y. This is the default state mode.
- MODE_LEFTCLICK (B1000): This mode allows the user to left click as on a mouse.
- MODE_RIGHTCLICK (B0100): This mode allows the user to right click as on a mouse.
- MODE_CALIBRATION (B0010): This mode sets the reference of angle X and Y for the moving mode.
- MODE_MOVING (B0111): In the Moving mode, the speed that the cursor moves is set by the angular displacement of the user's hand relative to the reference set in the calibration mode.
- MODE_SCROLLING (B1111): This mode allows the user to scroll up and down using the angular displacement of the user's hand relative to the reference set in the calibration mode.

3.3.2.1 Reading Flex Sensors into State Machine

The following flowchart shows how the microcontroller reads the flex sensor information. Please refer to section 4.3.1.3 for information on the microcontroller processing of this received data and software implementation of the finite state machine.

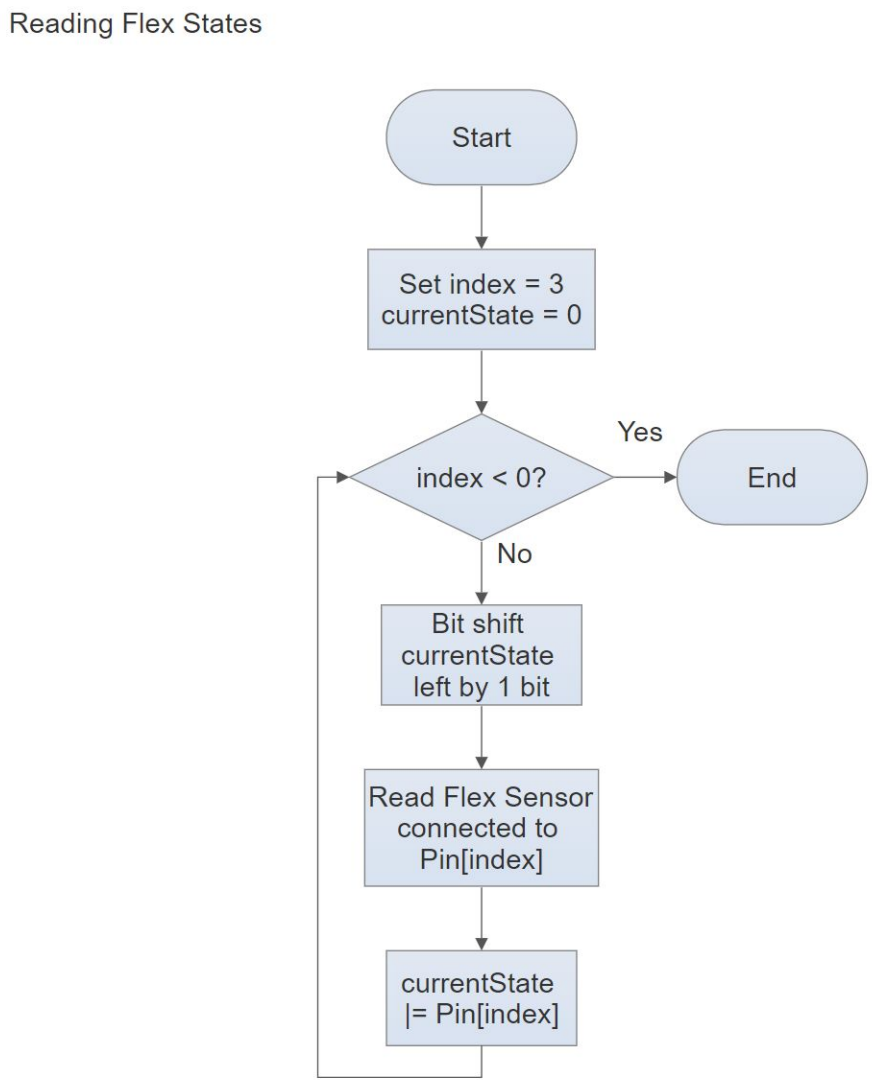
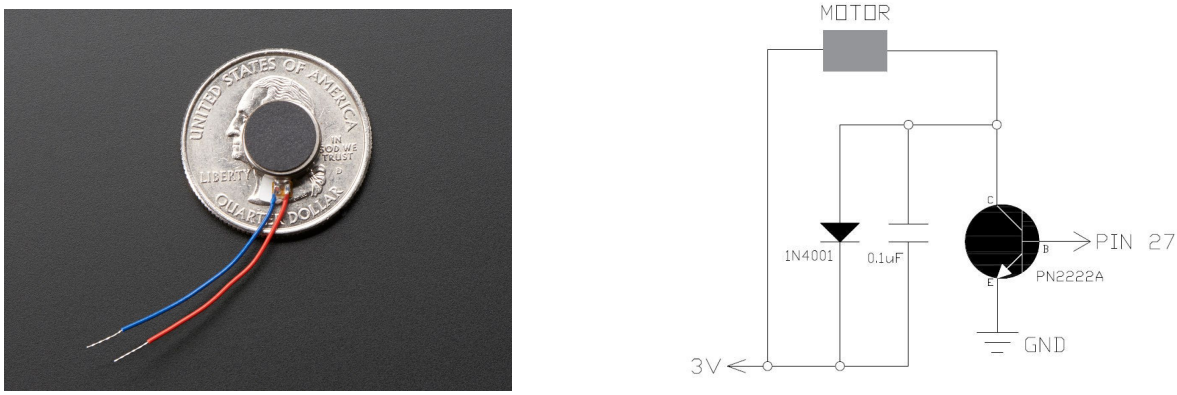


Figure 3.3-4: Flowchart for setting the current state from the flex sensors.

The flowchart in figure 3.3-4 shows the algorithm that the microcontroller uses to poll the states of all the flex sensors by reading their pin voltages and accordingly identifies the FSM state.

3.4 Vibration Motor



(a) (b)

Figure 3.4-1: Vibration motor and circuit [4]. (a) Vibration motor size compared to a quarter (b) Electrical schematic to connect to the nRF52

The Air Mouse provides the user with haptic feedback after every successful left or right click action through the means of a vibration motor. The vibration motor is made up of tiny discs which are completely sealed inside a casing intended to keep the user safe and easily implement onto the glove. The vibration motor is connected to the microprocessor as is illustrated in Figure 3.4-1. The duty cycle controls the intensity of the vibration, and which was set to 20% for this design. Thus, the motor intensity changed depending on user preference.Pulse-width modulation (PWM) is a modulation technique used to encode a message into a pulsing signal. Although this modulation technique can be used to encode information for transmission, its main use is to allow the control of the power supplied to electrical devices such as motors

The following are the technical details :

- Dimension: 10mm diameter, 2.7 mm thick
- Voltage: 2V - 5 V
- 5V current draw: 100mA, 4V current draw: 80mA, 3V current draw: 60mA , 2V current draw: 40mA
- 1100 RPM at 5V
- Weight:0.9 gram

3.5 Battery and Power Management System

3.5.1 Power Supply

Figure 3.5-1: LiPo battery size compared to a quarter[20].

A 3.70 V/110 mAh rated Lithium Ion rechargeable battery was chosen to fulfill the requirement of a portable power source[20]. The battery served as an appropriate choice due to its lightweight and compact size, which were essential to meet the portability and ergonomic constraints of the project. The following are some key specifications of the product that were instrumental in the selection of the battery:

- Nominal Voltage = 3.70V
- Rated Capacity = 110mAh
- Discharge Cut-off Voltage = 2.80V
- External Battery Dimensions = 27mm x 15.5mm x 4.2mm
- Weight = Less than 8g

3.5.2 Battery Monitoring System

As seen in Figure 3.5-2, the battery connects to the nRF52 board via a JST PH connector[27]. The lithium battery used has a voltage range of 2.8 - 4.2 V. The “vbat” pin on the board is connected to an internal ADC channel. This is connected to a voltage regulator which means that the voltage read by the vbat pin is not the raw battery voltage but the voltage coming out of the regulator. The following figure shows the internal LiPo monitoring circuit:

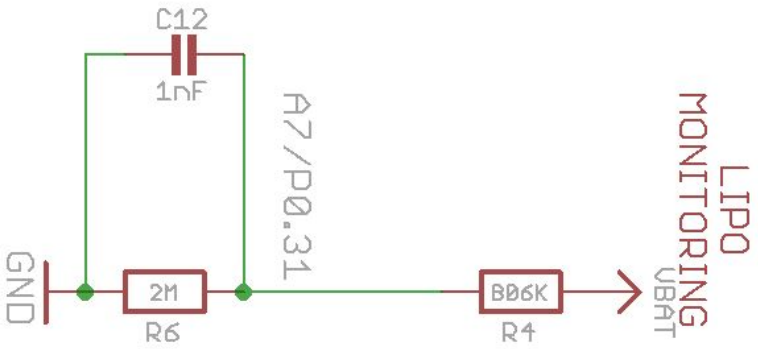


Figure 3.5-2: LiPo battery monitoring circuit on the nRF52 board.

Therefore,

Maximum voltage = 4.2V * (2M / (0.806 M + 2M)) = 3V
Minimum voltage = 2.7V * (2M / (0.806 M + 2M)) = 1.93V

This regulator configuration results in a leakage current of 4.2V/ 2.8MOhms = 1.5uA and a gain of ⅛ with an internal reference of 0.6V.

Using a 12-bit ADC setup,

ADC value at 4.2V = 3V * (⅛)/ 0.6V * 4095 = 4095
ADC value at 2.7V = 1.93V * (⅛)/ 0.6 * 4095 = 2634
Resulting in a usable ADC resolution of 1461.

Hence, the microcontroller reads the at a set delay of 1 second and returns a raw 12-bit ADC value. This value is converted into a mV value using a mV per LSB (least significant digit) factor of 3000mV/4096 = 0.7324. However, this mV value is not compensated to account for the divider loss. Thus, the results are multiplied by a compensation factor of 1.403 which result in the actual LiPo voltage level.

Based on typical LiPo battery discharge curves, the battery discharge curve was arbitrarily divided into set levels that correspond to a particular battery percentage level. The raw voltage value obtained is converted into a percentage level based on the following criteria:

Voltage Level (mV)	Remaining Battery Life (%)
>= 3000	100
> 2900	99 - 43
> 2740	42 - 19
> 2440	18 - 6
> 2100	6 - 1
= 0	0

Table 3.5-1: Remaining battery life vs. LiPo voltage level

3.5.3 Battery State Indicators

There are two ways of determining the remaining battery life of the Air Mouse:

1. NeoPixels

A color code scheme was arbitrarily developed to indicate the battery state of charge to the user. This led to the introduction of NeoPixels as a medium to display the appropriate color[2]. NeoPixels are smart LEDs with a constant current driver and 24-bit color ability. The battery level is exhibited by the NeoPixels using the following color scheme[21]:

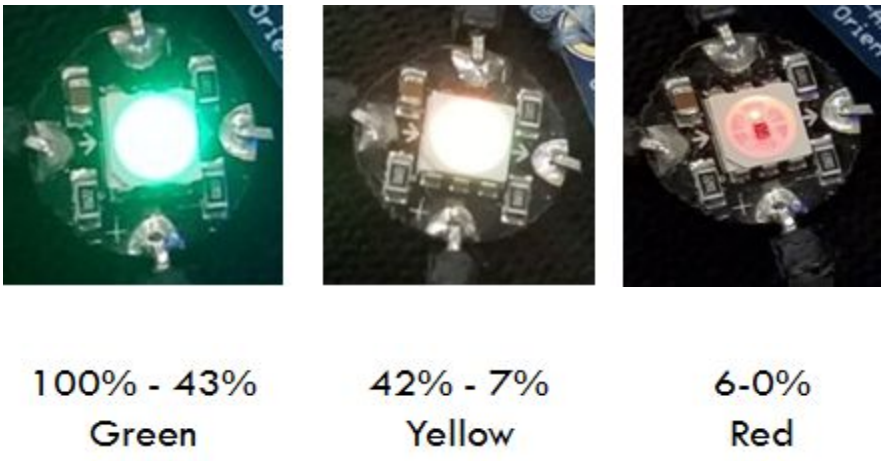


Figure 3.5-3: NeoPixels color code displaying battery state of charge.

Another parameter that adds to the accuracy of determining the battery level from simply looking at the NeoPixels is the varying LED brightness at each color level. The brightness level changes directly

according to the battery percentage level. For example, the LED is at its brightest green color light at a maximum charge of 100% and then diminishes slowly till it finally changes to the brightest yellow. Similarly, once it reaches the brightest red light, it fades till the battery is completely drained, or in other words, reaches 0%.

2. Microsoft Application

The second way to read battery percentage level is on the Universal Windows Platform and Bluetooth LE 3D Application[15],[16]. The microcontroller uses the BLEBas and GATT services to wirelessly communicate with the Application in real time to display the remaining battery percentage level[2].

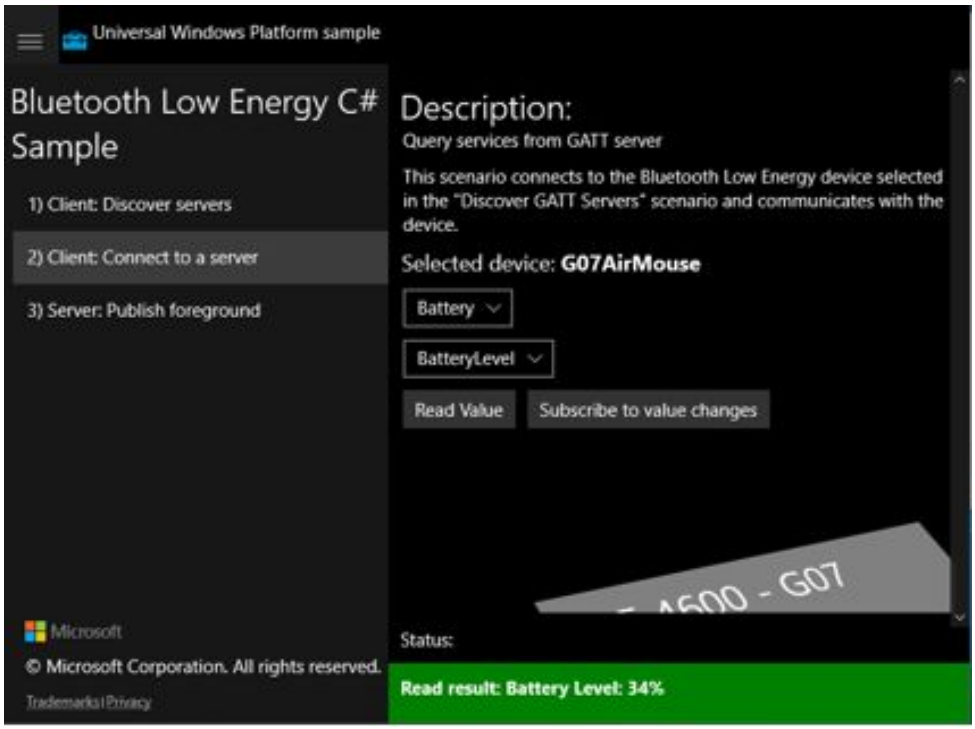


Figure 3.5-4: Battery percentage level displayed on the Windows Application.

More information on BLEbas along with other libraries and GATT services can be found in Section 4.1.3.

3.5.4 Battery Life and Operational Time

The Air Mouse is currently programmed to run at its maximum transmit power output level of +4dBm. The theoretical battery lifetime was calculated using the equation 3.1:

$$\textit{Operational Time} = (\textit{Battery Capacity}/\ \textit{Decive Current Draw}) \times 0.8^{*} \quad (3.1)$$

$= (110mAh / 44.3mA) \times 0.8$ *

**Note: The factor of 0.8 accounts for various real-world losses and provides a more conservative battery life estimate.*

Power Supply Management

The nRF52 board comes with an internal power management unit that automatically allocates the required resources, at any given time, to optimize the system and achieve lowest power consumption without active user input[13],[15].

Air Mouse Power Specifications

Current draw and Lifetime Expectations Table.

DEVICE	**CURRENT DRAW**	
NeoPixels	18.5mA	
BNO055	12.3mA	
Flex Sensors	1.05mA	
Laser pointer*	25mA	
Vibration Motor*	16mA	
Pressure Sensitive Sheet	0.7mA	
Antenna Transmission Current	7.5mA	
nRF52 CPU Current	3.9mA	
Total Operational Time	**THEORETICAL**	**TESTED**
	2 hours 39 mins	2 hours 20 mins

Table 3.5-2: Air Mouse components current draw and operational time.

As with most battery operated devices, battery consumption and operational times may vary according to the use of Air Mouse and its features. This is due to non-constant current draw of the various components and the fact that as current draw increases, the battery capacity decreases. Hence, average current consumption was used to determine the expected runtime. The test results are based on the use of Air

Mouse features such as the laser pointer in moderation, rather than running continuously, to achieve the results.

**Note: The theoretical operational time calculations assume that the laser pointer and vibration motor are on for 1% of the functional time.*

Range Optimization and Tradeoffs:

The range of connectivity of the Air Mouse to a desired Bluetooth enabled device was calculated as follows, using f = 2.45GHz for Bluetooth LE:

Sensitivity = -96dBm
Maximum transmit power = +4dBm
Minimum transmit power = -20dBm
FSPL (dB) = 20log(d) + 20log(f) + (-27.55) (d in meters and f in Gigahertz)
Maximum path loss = transmit power – receiver sensitivity + gains – losses – fade margin
Distance (m) = 10000*(maximum path loss – 32.44 – 20log(f))/20

TRANSMIT POWER (dBm)	RANGE (m)
+4	30
0	13
-10	5
-20	1

Table 3.5-3: Operational range vs. transmit power

Thus, there exists a tradeoff between range and battery life. The lower the transmit power, lower the range but increased battery life due to lesser power consumption.

3.5.5 Air Mouse Charging Methods

The battery can be recharged using voltage supply off a USB connector, nominally 4.5 - 5.2 V, or through a wall socket connected to a USB adapter. The VBUS from the USB is connected to the VDD pin of the charging controller below.

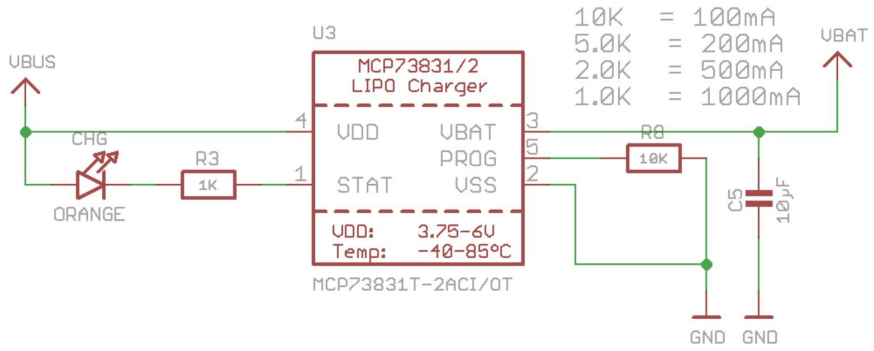


Figure 3.5-5: LiPo Controller used for charging[28]

The nRF52 board uses the MCP73831/2 Li-Polymer charge management controller with a programmable charge current of 15 mA to a maximum of 500 mA. The Adafruit implementation connects a 10 kΩ resistor to Pin 5, which results in a charging current of 100 mA. For the rated capacity of 110 mAh, the time taken to charge the battery from a fully discharged state:

(110mAh/100mA)*60 mins = 65 mins or 1 hour 5 minutes

The time taken to fully charge the battery from a "dead" state (battery level < 6 mV) using a USB connector was experimentally determined to be 1 hour 10 minutes.

3.6 Pressure Sensitive Sheets

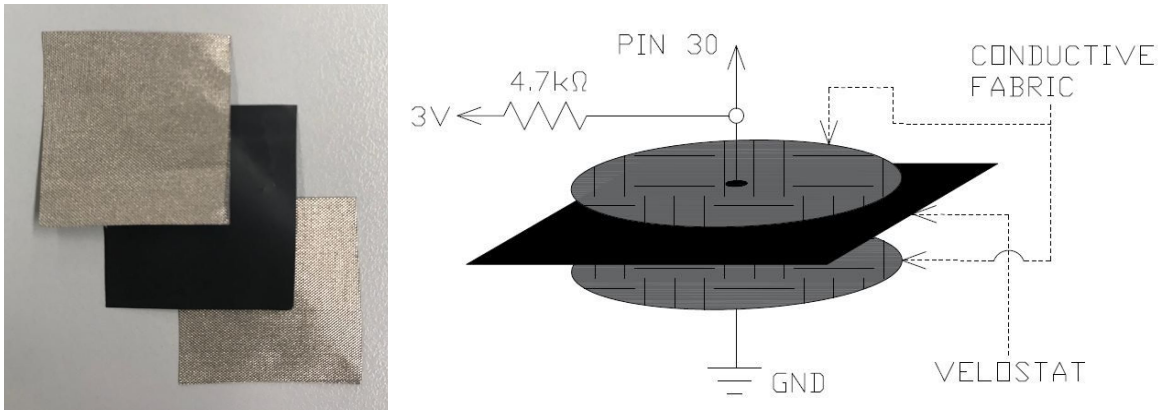


Figure 3.6-1: Pressure and conductive sheets sandwich.

3.6.1 Background

The pressure sensitive sheets, as referred to in the project, are a structure consisting of a section of velostat conductive sheet sandwiched between two sheets of conductive fabric, one of which is connected to the ground and the other to the nRF52. Electrical connections are made at the top and bottom silver sheets, which are connected through the pressure sheet. When this sheets structure is pressed, the pressure increases and the resistivity of the sheets reduces. This creates a voltage difference large enough for the threshold voltage to be triggered within the nRF52, allowing the glove to receive this command. The threshold voltage is taken as 1.5V, which is approximately the midpoint of the ADC range, because the pressure sheet is effectively a short circuit to ground when pressed.

3.6.2 Pressure Sensitive Sheet Technical Details[29]

The following are the technical details for the velostat conductive sheets and the conductive fabric used:

Velostat (One Sheet)

- Dimensions: 11" x 11" (280mm x 280mm)
- 4 mil / 0.1mm thick
- Weight: 18.66g
- Temperature Limits : -45°C to 65°C (-50°F to 150°F)
- Heat Sealable : Yes
- Volume Resistivity : <500 ohm-cm

- Surface Resistivity : < 31,000 ohms/sq.cm

Conductive Fabric

- Weight: 80+/-10 g/m^2
- Thickness: 0.08+/-0.01 mm
- Surface Resistivity: <0.05 ohm/sq.cm

3.6.3 Evolution Of Pressure Sensitive Sheet Design

From the offset, the intended use of the pressure sensitive sheets was in combination with the flex sensors as a means of relaying the necessary click commands from hand gestures to the the nRF52. The concept of using this combination was brought forth to make the glove more user friendly for people with limited hand mobility. The notion was that having more than one way of sending each command might make one of the hand positions or gestures more comfortable for those with afflictions such as carpal tunnel syndrome, diabetic hand syndrome or Dupuytren's contracture. By placing the sheets on each finger tip and pressing them against a surface such as one's thumb, the range of motions would be centred less on the finger flexing which might be difficult for many with the said conditions.

During testing of the components for implementation onto the glove, it was found that the pressure sensitive sheets were rather unreliable due to the structure shifting between uses as a result of the pinching action. This unreliability made it difficult to set a threshold value sufficient for our intended use. If the system was set to a high threshold, it would require the user to press or pinch harder to trigger a command, thereby, increasing the difficulty for people with limited hand mobility. Too low of a threshold would make the sheets vulnerable to sending undesired commands due to unintended pressing. This uncertainty resulted in the temporary removal of the pressure sensitive sheets from the design of our glove as ensuring the functional use of the more reliable flex sensors on the glove.

The use of the sheets was revisited as the project entered the prototyping stage. A user friendly way of effectively and fluidly powering on and off the laser diode was explored using the pressure sensitive and conductive sheet sandwich as shown in Figure 3.6-1. They were found to be the best choice due to their low-profile structure and ease-of-use. By placing a sheet on side of the middle phalanx of the index finger, the sheet structure can be effortlessly activate by the thumb.

3.7 Laser Diode

Figure 3.7-1: Laser diode used on the Air Mouse[29]

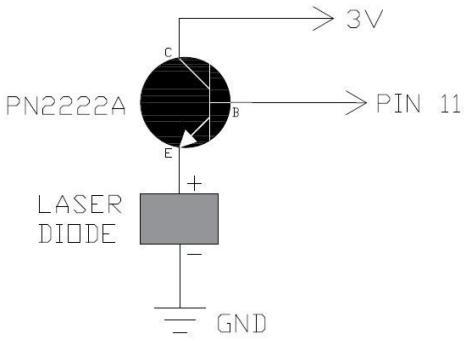


Figure 3.7-2: Laser diode circuit

The laser diode is an additional feature of the Air Mouse as a visual aid for presentations. A low power, light weight, red light laser diode was selected to go on the proximal phalanx of the user's thumb. The diode has the following attributes:

- Class IIIa 5mW
- 650nm red wavelength
- Weight 6.3 grams
- Diameter 10mm
- Length 31mm

The action of tucking the thumb into a curl and holding index finger (Figure 3.7-2) allows for ergonomic activation and directioning of the laser diode.

(a) (b)

Figure 3.7-3: Controlling the Laser Diode using the Pressure Sheet. (a) Unpressed state diode OFF (b) Pressed state diode ON

The diode draws a maximum current of 25mA whereas the GPIO pins of the nRF52 can source a maximum of 14mA. To ensure enough current supply to the diode a transistor circuit was added as shown in figure 3.7-2.

Chapter 4. Software

4.1 Bluetooth Low Energy

Bluetooth Low Energy (BLE), formerly known as Bluetooth Smart, is a low power subset of classic Bluetooth and was introduced as part of the Bluetooth 4.0 core specification. The support for Bluetooth 4.0 and Bluetooth Low Energy (which is a subset of BT 4.0) is available on most major platforms as of the versions listed below:

- iOS5+ (iOS7+ preferred)
- Android 4.3+ (numerous bug fixes in 4.4+)
- Apple OS X 10.6+
- Windows 8+ (XP, Vista and 7 only support Bluetooth 2.1)

Thus, making the Air Mouse compatible with all of these Bluetooth LE enabled devices.

4.1.1 Bluetooth LE Physical Layer

Bluetooth Low Energy uses the 2.4 GHz ISM (Industrial, Scientific, and Medical) band. This band is divided into 40 channels on 2 MHz spacing from 2.4000 GHz to 2.4835 GHz, starting at 2402 MHz, with center frequencies at 2402 + k*2MHz, where k ranges from 0 to 39.

The 40 channels are divided into 3 Advertising Channels (Ch. 37, 38, 39), and 37 Data Channels (Ch. 0-36)[18]. The Advertising Channels are used for device discovery, connection establishment, and broadcast transmissions. The Data Channels allow bidirectional communication between connected devices and adaptive frequency hopping used for subsequent connection events.

Bluetooth LE transmits at 1 Mbps, with 1 bit per symbol, which is optimized for sending small chunks of data quickly. The modulation scheme used is Gaussian Frequency-Shift Keying (GFSK). In GFSK, the data pulses are filtered with a Gaussian Filter before being applied to alter the carrier frequency to make the frequency transitions smoother. In practice however, the BLE protocol overhead limits overall data throughput to significantly less than 1 Mbps.

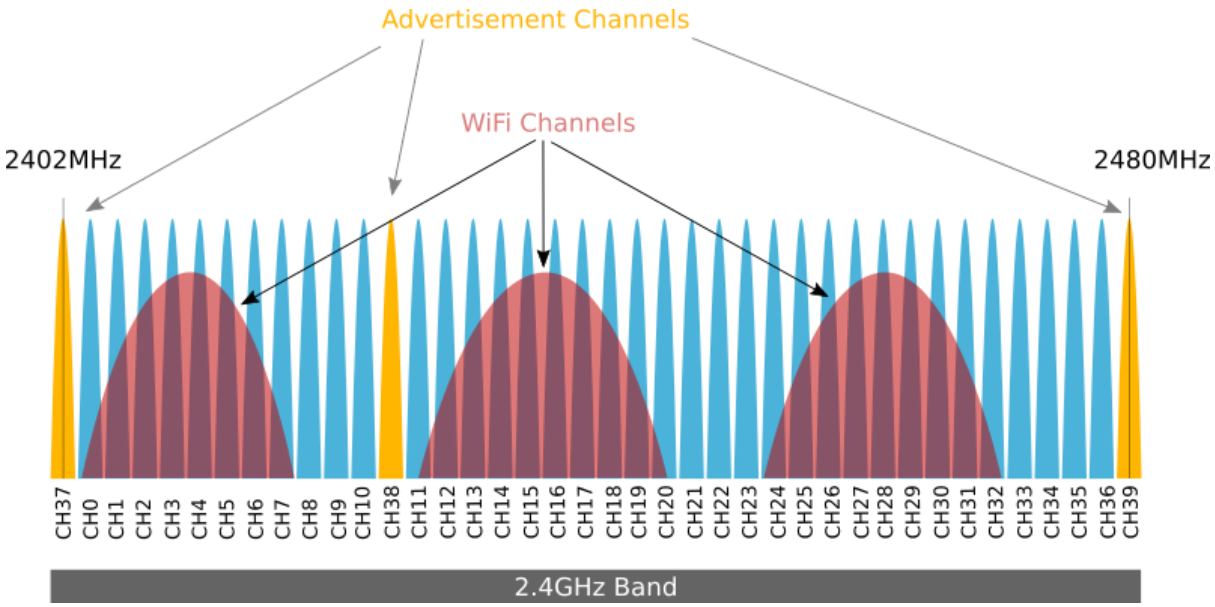


Figure 4.1-1: Bluetooth LE and IEEE 802.11 2.4GHz channels[31].

If any single advertising channel is blocked, the other channels are likely to be free since they are separated by a couple MHz of bandwidth making Bluetooth LE technology a powerful and reliable communication protocol suitable for the Air Mouse.

4.1.2 Bluetooth LE Generic Access Profile (GAP) and Pairing

The Generic Access Profile (GAP) controls connections and advertising in Bluetooth LE. GAP makes devices visible to the outside world, and determines how two devices can (or can't) interact with each other. GAP defines various roles for devices, with the two key categories called central devices and peripheral devices[15].

- Central devices are usually the mobile phones, tablets or personal computers that you connect to with far more processing power and memory.
- Peripheral devices are small, low power, resource constrained devices that can connect to a much more powerful central device. The Air Mouse falls in this category.

Advertising is the first step in establishing Bluetooth LE connections and allows devices to broadcast information defining their intentions. The Raytac module on the nRF52 constantly broadcasts advertising

packets with an advertising interval[18]. Advertising intervals can be changed but there is a requirement for minimum and maximum advertising interval.

In order to establish a connection with the Air Mouse, or such peripheral devices, a central device that wants to start a connection has to scan for an advertising packet and send a connect request packet after it receives the advertising packet from the peripheral device. The connect request must be sent within the RX windows opened by the advertiser right after each advertising packet transmission.

When the Air Mouse receives a connect request and if the connection request is accepted, it stops advertising in the next channel and/or next advertising interval and then follows the parameter in the connect request packet to start a connection. The following figure shows an example of when the microcontroller receives a connect request on channel 38. It stops advertising when receives the packet, and a connection is established with the parameter included in the connect request.

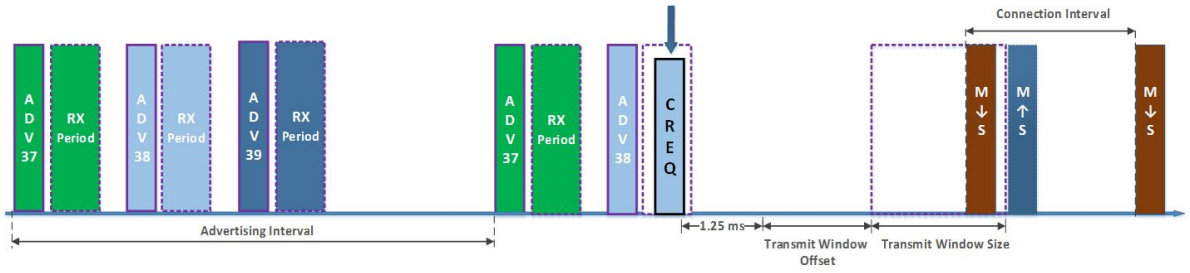


Figure 4.1-2: Establishing Bluetooth LE connections with Advertising[32].

4.1.3 Bluetooth LE Generic Attribute Profile (GATT)

The Generic Attribute Profile (GATT) establishes how data will be organized and exchanged over a Bluetooth Low Energy connection. GATT uses the Attribute Protocol (ATT) which is used to store Services, Characteristics and related data in a simple lookup table using 16-bit IDs for each entry in the table.

GATT comes into play once a dedicated connection is established between two devices, meaning that you have already gone through the advertising process governed by GAP. A BLE peripheral such as an Air Mouse can only be connected to one central device (a mobile phone, etc.) at a time. This means that as soon as the Air Mouse connects to a central device, it will stop advertising itself and other devices will no longer be able to see it or connect to it until the existing connection is broken.

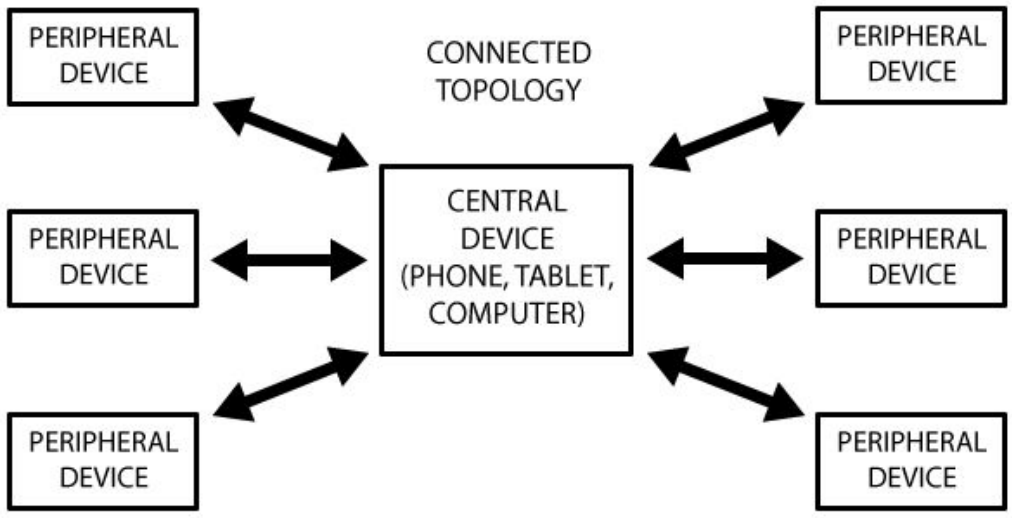


Figure 4.1-3: Air Mouse (Peripheral) connects to devices (Central).

GATT transactions operate using a client/server relationship. The peripheral (Air Mouse) is known as the GATT Server, which holds the ATT lookup data and service and characteristic definitions, and the central device is known as the GATT Client (the phone/tablet), which sends requests to this server. All transactions are started by the master device, the GATT Client, which receives response from the slave device, the GATT Server.

GATT transactions in BLE are based on high-level, nested objects called Profiles, Services and Characteristics.

Profiles are predefined collections of Services that has been compiled by either the Bluetooth SIG or by the peripheral designers.

Services are used to break data up into logic entities, and contain specific chunks of data called characteristics. A service can have one or more characteristics, and each service distinguishes itself from other services by means of a unique numeric ID called a UUID, which can be either 16-bit (for officially adopted BLE Services) or 128-bit (for custom services).

The lowest level concept in GATT transactions is the Characteristic, which encapsulates a single data point (though it may contain an array of related data, such as X/Y/Z values from a 3-axis accelerometer, etc.). Each Characteristic also distinguishes itself via a predefined 16-bit or 128-bit UUID.

4.2 Adafruit Bluefruit nRF52 Application Programming Interface (API)

The Adafruit nRF52 core defines a number of custom classes that aim to make it easy to work with BLE in projects. The classes used for this project are:

- BLEHid Adafruit Class: The BLEHid Adafruit class allows you to simulate a mouse or keyboard using the HID (Human Interface Device) profile that is part of the Bluetooth Low Energy standard.
- BLEUart Class: BLEUart is a wrapper class for NUS (Nordic UART Service), which is a proprietary service defined by Nordic Semiconductors. This service can be used to easily send ASCII or binary data in both directions, between the peripheral and the central device[13],[33].

4.3 nRF52 User Firmware/Sketch

4.3.1 HID Mouse over Bluetooth LE

The flow charts below show the code sequence of the setup and main functions that run on the nRF52 to bring the Air Mouse to life[15],[18]. Please refer to the Appendix for the code implementation of these flowcharts.

4.3.1.1 Setup Board

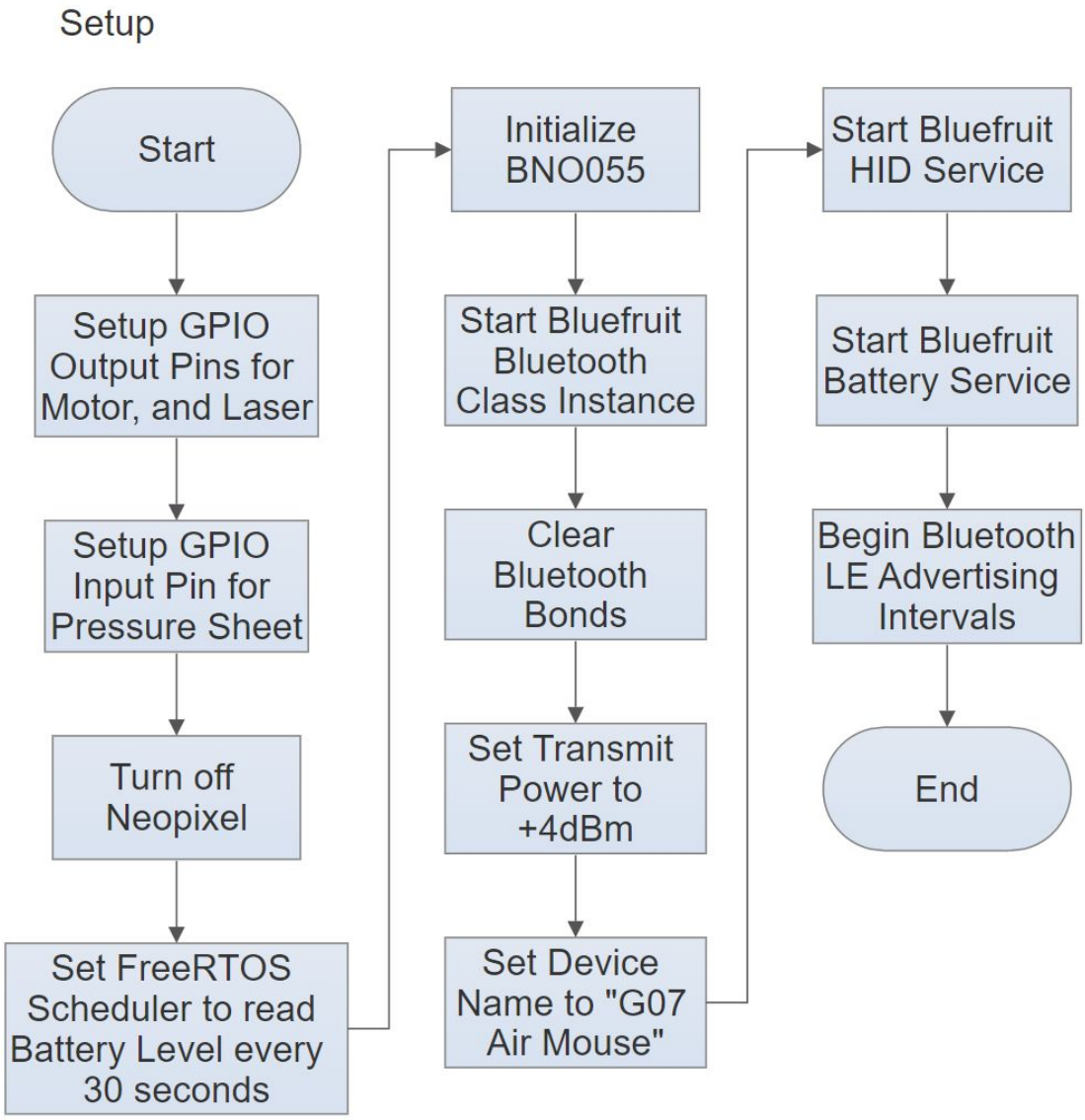


Figure 4.3-1: Flowchart to setup peripherals on nRF52

4.3.1.2 Main Loop

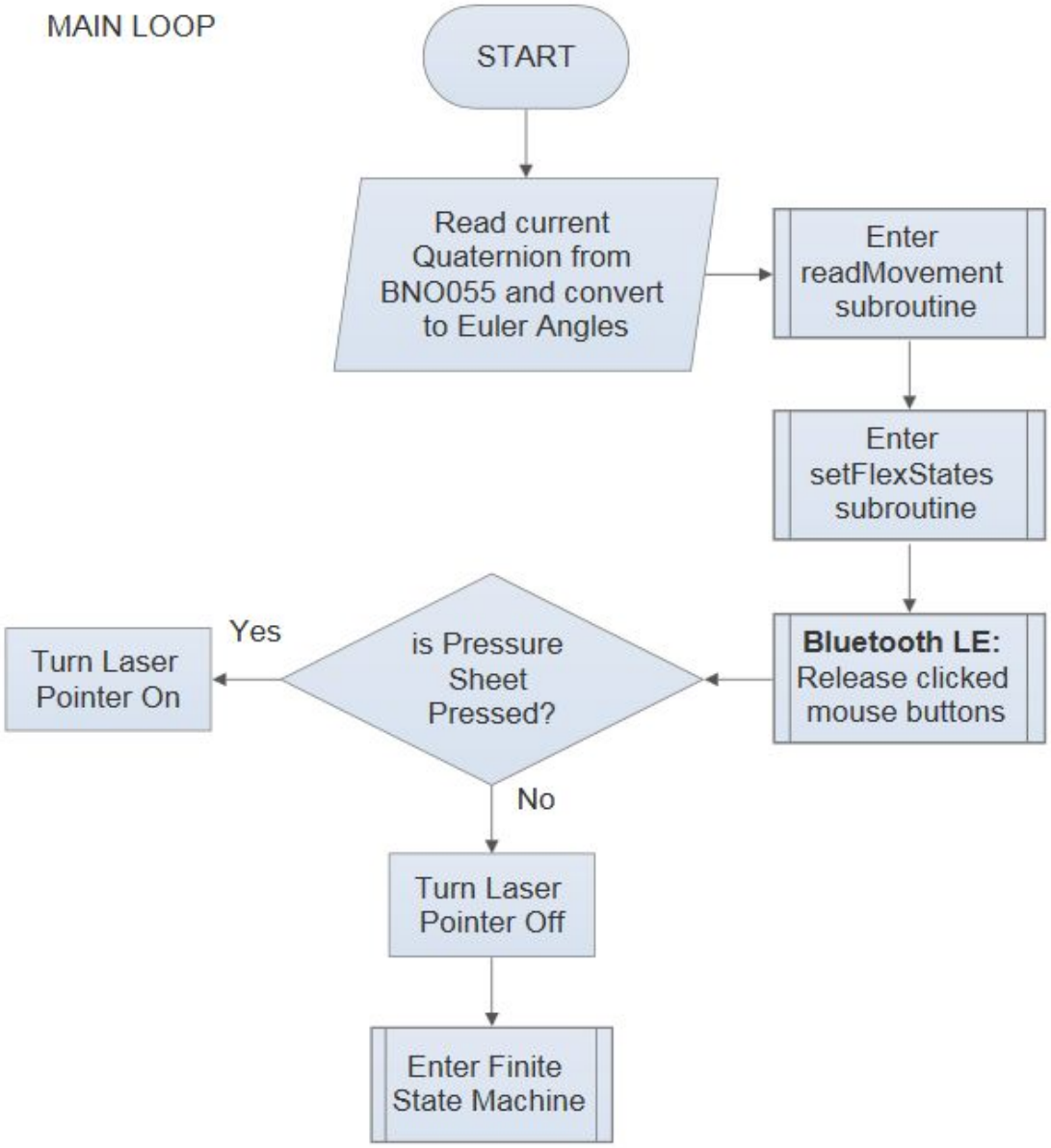


Figure 4.3-2: Flowchart for main loop

4.3.1.3 Finite State Machine

The following flowchart is a representation of the finite state machine implementation on the software level.

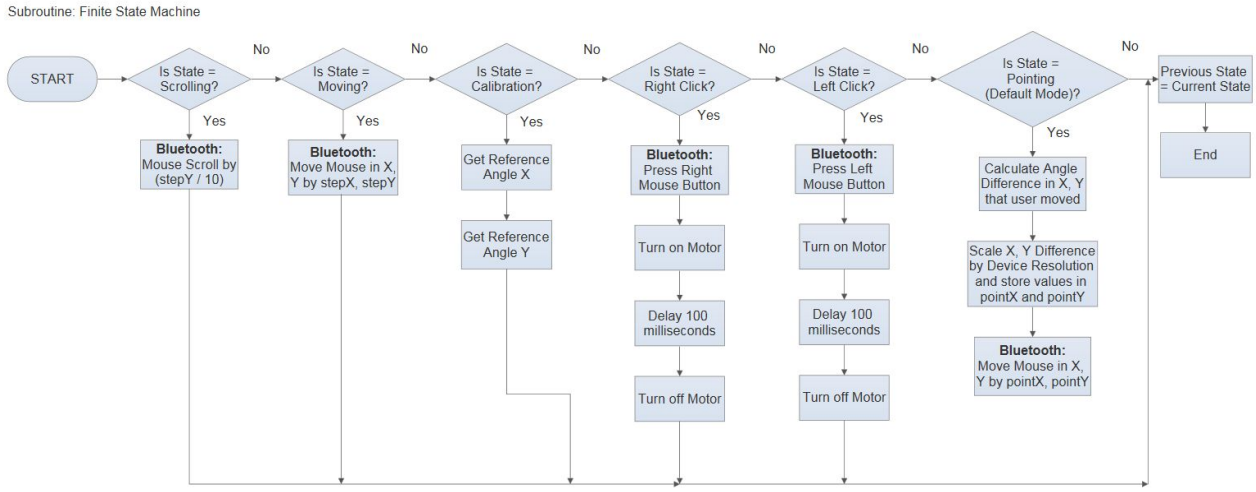


Figure 4.3-3: Flowchart for different states of FSM

4.4 Bluetooth LE Windows Application

Microsoft published a GitHub repository with examples that demonstrate the API usage patterns for the Universal Windows Platform (UWP) in the Windows Software Development Kit (SDK) for Windows 10. These code samples were created with the Universal Windows Platform templates available in Visual Studio, and are designed to run on desktop, mobile, and future devices that support the Universal Windows Platform.

The examples demonstrates how to use the Windows Bluetooth LE APIs to act either as a BLE client or server. It shows how to act as a client to communicate with a Bluetooth Low Energy (LE) device using the Bluetooth GATT protocol and query for supported services and characteristics.

4.4.1 Panel to Demonstrate Air Mouse Movements in Space

Some functions were added to the Microsoft Bluetooth LE example application to parse the sent commands explained in Section 4.3.2 of this document. A panel was added to the application to allow the

user to visualize the orientation and movement of the glove on the user's hand. If the received packet corresponds to an Euler angle (eulerX, eulerY, eulerZ), the program rotates the panel to match the orientation of the user's hand. If the received packet corresponds to position (posX, posY, posZ), the panel is offset by an amount in the corresponding axis, which will give the visual effect of moving in 3D space.

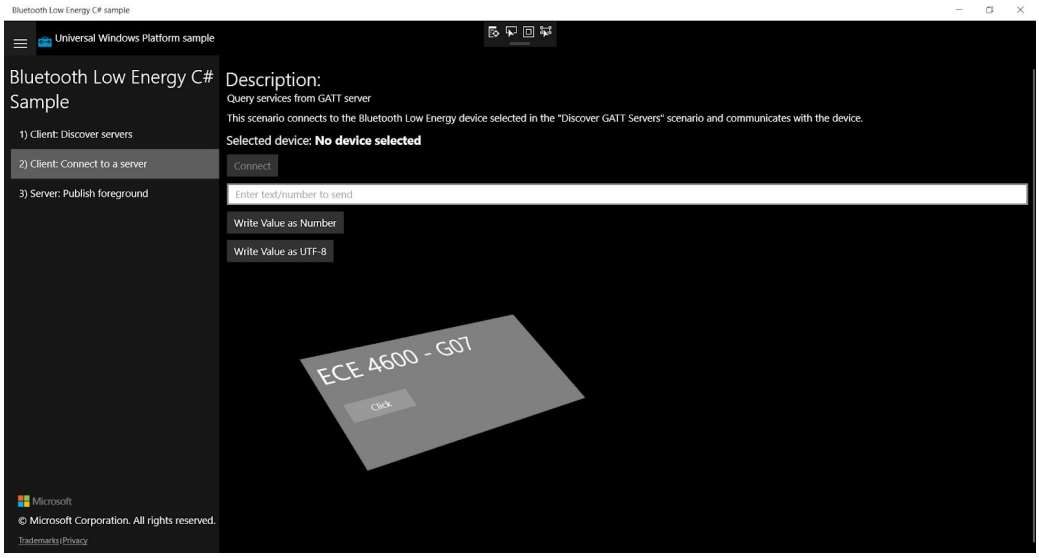


Figure 4.4-1: Visual demonstration in the Windows Application.

As seen in Figure 4.4-1, the panel window is a rectangular plane that appears in the Microsoft Windows application which replicates the 3D user movements in space and in real time[15],[16].

4.4.2 Transmitting Positional Information to the Application

A separate program for the nRF52 was written to communicate with the Windows Application. The nRF52 reads orientation in the form of quaternions and acceleration values from the BNO055 over I^2C[1]. The acceleration values are high pass filtered to remove the gravity vector, then double integrated to obtain the position data. The position data is averaged over 5 samples to obtain more consistent user input. The quaternion data is converted to Euler angles using the Adafruit quaternion library. The Euler angles and displacement values are transmitted over Bluetooth LE using the BLE UART service[15],[16].

If the command corresponds to position (posX, posY, posZ), the value sent is a non-standard integer measure of distance. If the command corresponds to Euler Angles (eulerX, eulerY, eulerZ), the value sent is an integer representing the orientation angle with respect to one axis in degrees[1],[17],[19].

Chapter 5. Summary of Results and Conclusion

5.1 Future Considerations

The Air Mouse prototype successfully met the proposed requirements, which entails that it can perform all the functions of a conventional mouse with some additional features such as the laser pointer and haptic feedback. However, there are certain improvements that can make it an even more powerful device with the capability to pioneer innovative ways of interacting with technological devices. One such development would be to expand in the 3D space design. This would mean adding more functionality to the Air Mouse such that it is able to replace gaming tools to provide a more engaging and involved experience to the user.

Currently, the design is limited to devices that have an inbuilt Bluetooth LE support. The basic design of the Air Mouse can be emulated to work with other, more universal platforms.

The final prototype can also be further improved by using conductive thread instead of the copper coated wires that it is currently designed with. A larger battery will also provide the user more hours for operation. A smaller laser or one that draws less current will also help prolong battery life.

An ambitious goal would be to include keyboard typing abilities solely based on hand gestures to the Air Mouse which will make it an all-in-one device that truly gives the user freedom of movement and complete wireless control.

5.2 Conclusion

To conclude, there are many existing and developing technologies for human-computer interaction. The Air Mouse project is an attempt to make this interaction less restrictive in terms of setting and space. The technology gives the user freedom of mobility along with the potential for complete device control literally at their fingertips. This makes the device capable of being integrated into several emerging fields, such as virtual/augmented reality, and remote control in industrial or medical applications. The project can also be seen as a foundation and reference to re-invent as well as explore interesting possibilities in the Bluetooth LE communication domain.

Bibliography

1. https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor?view=all#arduino-code
2. https://www.nxp.com/docs/en/application-note/AN10216.pdf
3. https://www.direcs.de/2017/07/der-adafruit-bno055-9-dof-sensor-imu-breakout-macht-es-einem-nicht-leicht-oder-doch/
4. https://cdn-shop.adafruit.com/1200x900/2472-02.jpg
5. https://patents.google.com/patent/US4787051
6. https://learn.adafruit.com/introduction-to-bluetooth-low-energy/introduction
7. https://en.wikipedia.org/wiki/Bluetooth_Low_Energy
8. http://www.argenox.com/a-ble-advertising-primer/
9. http://microchipdeveloper.com/wireless:ble-introduction
10. D.Fontaine, D.David and Y.Caritu, "Sourceless Human Body Motion Capture," Smart Objects Conference (SOC 2003), Grenoble, 2003.
11. J.Bernstein, H.R.Everett, L.Feng, and D.Wehe, "Mobile Robot Positioning Sensors & Techniques," Journal of Robotic Systems, Special Issue on Mobile Robots, Vol. 14, No. 4, pp. 231-249.
12. Adafruit Feather nRF52 - https://www.adafruit.com/product/3406
13. Nordic Semiconductor nRF52832 - http://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.0.pdf
14. Raytac MDBT42Q - http://www.raytac.com/download/MDBT42/MDBT42Q-Version%20E.PDF
15. https://github.com/JohnTT/BluetoothLEWin10NRF52/tree/master/Samples/BluetoothLE
16. https://github.com/JohnTT/ECECapstone2018
17. https://devzone.nordicsemi.com/b/blog/posts/bluetooth-smart-and-the-nordics-softdevices-part-1
18. https://blog.bluetooth.com/bluetooth-low-energy-it-starts-with-advertising
19. https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gap
20. https://www.sparkfun.com/products/13853
21. https://www.generationrobots.com/ea/flora-rgb-smart-pixels.pdf
22. https://www.arduino.cc/en/Tutorial/LiquidCrystalDisplay
23. https://codebender.cc/sketch:227118
24. https://github.com/adafruit/Adafruit_nRF52_Arduino/blob/master/libraries/Bluefruit52Lib/examples/Hardware/adc_vbat/adc_vbat.ino
25. https://www.adafruit.com/product/182
26. http://blogoscoped.com/archive/2004_12_05_index.html
27. https://learn.adafruit.com/bluefruit-nrf52-feather-learning-guide/device-pinout
28. https://learn.adafruit.com/assets/39913
29. https://www.adafruit.com/product/1361
30. https://www.digikey.ca/product-detail/en/adafruit-industries-llc/1054/1528-1391-ND/5629439
31. http://www.argenox.com/wp-content/uploads/ble-advertising-channels-spectrum.png
32. https://devzone.nordicsemi.com/cfs-file/key/communityserver-blogs-components-weblogfiles/00-00-00-00-04-DZ-841/4431.ConnectionEstablish.PNG
33. https://learn.adafruit.com/bluefruit-nrf52-feather-learning-guide/bluefruit-nrf52-api

Appendix

List of Figures - Appendix

Pulse Width Modulation

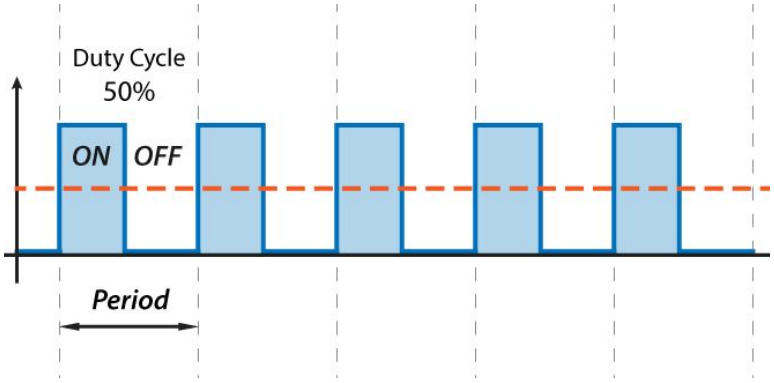


Figure A-1: Pulse width modulation used to control the power supplied to a motor.

EMA High Pass Filter

```
// EMA High Pass Filtering
// https://www.norwegiancreations.com/2016/03/arduino-tutorial-simple-high-pass-band-pass-and-band-stop-filtering/
EMA_S.x[0] = (EMA_a * accelSample.x[0]) + ((1 - EMA_a) * EMA_S.x[0]);
EMA_S.y[0] = (EMA_a * accelSample.y[0]) + ((1 - EMA_a) * EMA_S.y[0]);
EMA_S.z[0] = (EMA_a * accelSample.z[0]) + ((1 - EMA_a) * EMA_S.z[0]);

accelSample.x[0] -= EMA_S.x[0];
accelSample.y[0] -= EMA_S.y[0];
accelSample.z[0] -= EMA_S.z[0];

// Naive integration with dt^2
posX += Kpos * dt * dt * accelSample.x[0];
posY += Kpos * dt * dt * accelSample.y[0];
posZ += Kpos * dt * dt * accelSample.z[0];

// Add posX,posY,posZ to pos class
pos.addSample(posX, posY, posZ);
```

Figure A-2: EMA high-pass filter for acceleration data

Modifications to Microsoft Program

```
449         private void ParseCommand(string cmd)
450         {
451             List<string> cmdList = new List<string>(cmd.Split('='));
452 
453             if (cmdList[0].Equals("eulerX"))
454                 bleRotationX = Convert.ToInt32(cmdList[1]);
455             else if (cmdList[0].Equals("eulerY"))
456                 bleRotationY = Convert.ToInt32(cmdList[1]);
457             else if (cmdList[0].Equals("eulerZ"))
458                 bleRotationZ = Convert.ToInt32(cmdList[1]);
459 
460             else if (cmdList[0].Equals("posX"))
461                 blePosX = Convert.ToInt32(cmdList[1]);
462             else if (cmdList[0].Equals("posY"))
463                 blePosY = Convert.ToInt32(cmdList[1]);
464             else if (cmdList[0].Equals("posZ"))
465                 blePosZ = Convert.ToInt32(cmdList[1]);
466             debugText = cmdList[0];
467         }
468 
469         private int bleRotationX = 0;
470         private int bleRotationY = 0;
471         private int bleRotationZ = 0;
```

Figure A-3: ParseCommand() Implementation. This function performs different tasks depending on the command received from the BLE UART Service.

Bluetooth LE Stack

The following diagram depicts the architecture (major layers) of the BLE protocol stack. The Bluetooth Low Energy (BLE) physical layer (PHY) contains the analog communications circuitry responsible for translation of digital symbols over the air. It is the lowest layer of the protocol stack, and provides its services to the Link Layer. The Generic Access Profile (GAP) modes and procedures form the cornerstone for Bluetooth Low Energy (BLE) 'control-plane' operations such as establishing connections and broadcasting data over L2CAP.

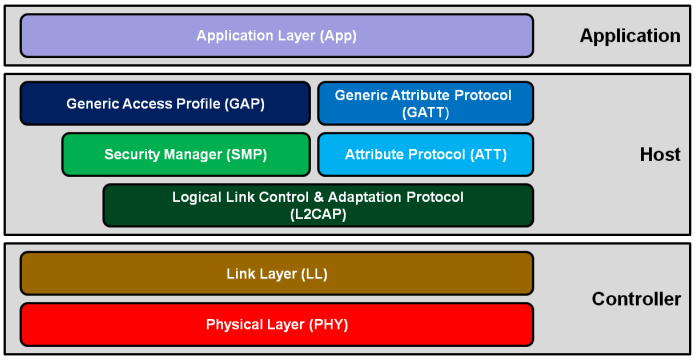


Figure A-4: Layers of the BLE Protocol Stack.

Source: https://microchip.wdfiles.com/local--files/wireless:ble-introduction/ble-protocol-stack.png

Bluetooth LE Generic Access Profile (GAP) and Advertising

https://devzone.nordicsemi.com/b/blog/posts/bluetooth-smart-and-the-nordics-softdevices-part-1

https://blog.bluetooth.com/bluetooth-low-energy-it-starts-with-advertising

https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gap

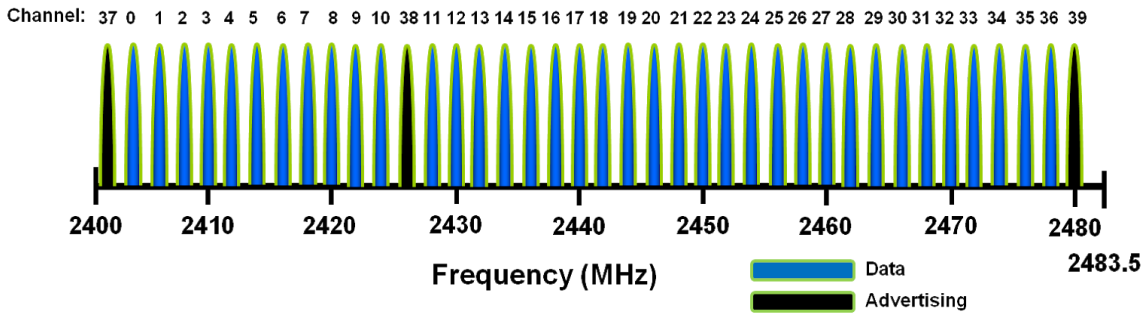


Figure A-5: Bluetooth LE Channel Allocation. Advertising occurs on Channel 37, 38 and 39. Source: https://microchip.wdfiles.com/local--files/wireless:ble-phy-layer/ble-phy-channel-assignment.png

Generic Access Profile, Advertising and Establishing a Connection.

There are two ways to send advertising out with the Generic Access Profile (GAP), either by using the Advertising Data payload or the Scan Response payload.

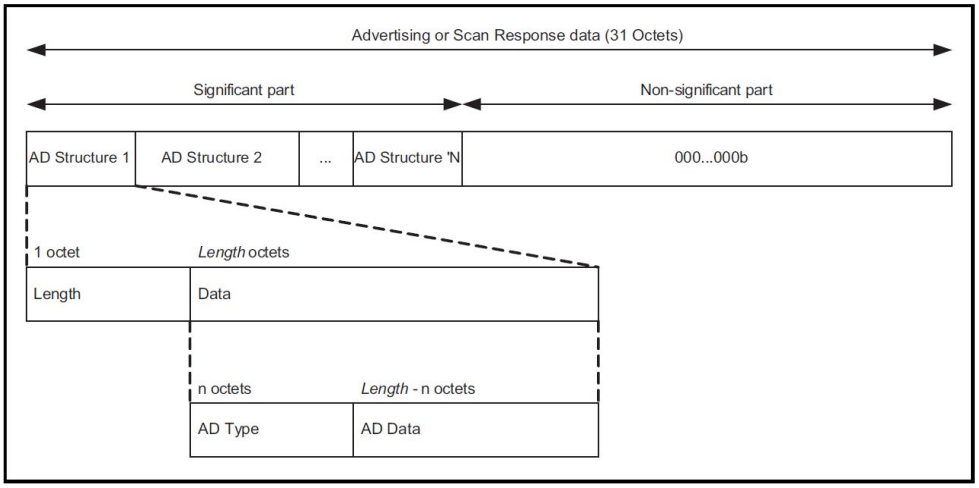


Figure A-6: BLE Advertising Packet. Bluetooth 4.0 Core Specification: Chapter 11 Part C Vol 3: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=229737

Both payloads are identical and can contain up to 31 bytes of data, but only the advertising data payload is mandatory, since this is the payload that will be constantly transmitted out from the device to let central devices in range know that it exists. For normal, undirected advertising, the advertising intervals ranges from 20ms to 10.24s (Section 7.8.5 Part E Vol. 2).

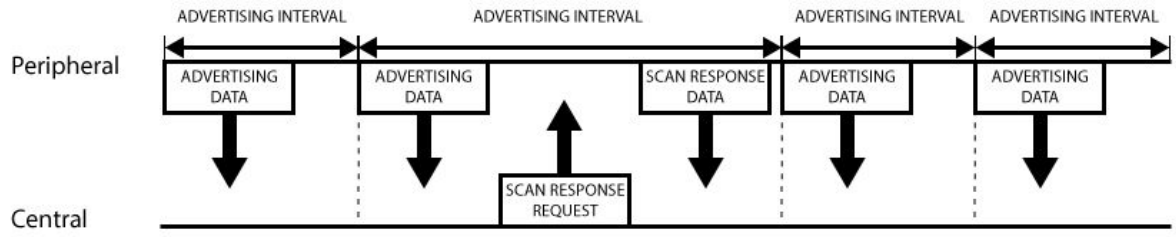


Figure A-7: BLE Advertising Interval. Source: https://learn.adafruit.com/assets/13825

The scan response payload is an optional secondary payload that central devices can request, and allows device designers to fit a bit more information in the advertising payload such a strings for a device name, etc.

BLE Data Transmission - GATT Transactions

When establishing a connection, the peripheral will suggest a 'Connection Interval' to the central device, and the central device will try to reconnect every connection interval to see if any new data is available, etc. However, the central device may not be able to honour the request because it is busy communicating with another peripheral or the required system resources are not available.

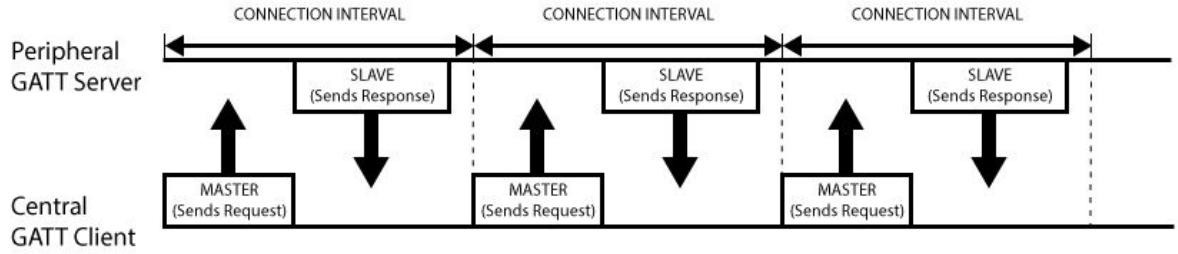


Figure A-8: Data Transmission through GATT in BLE. Source: https://learn.adafruit.com/assets/13827

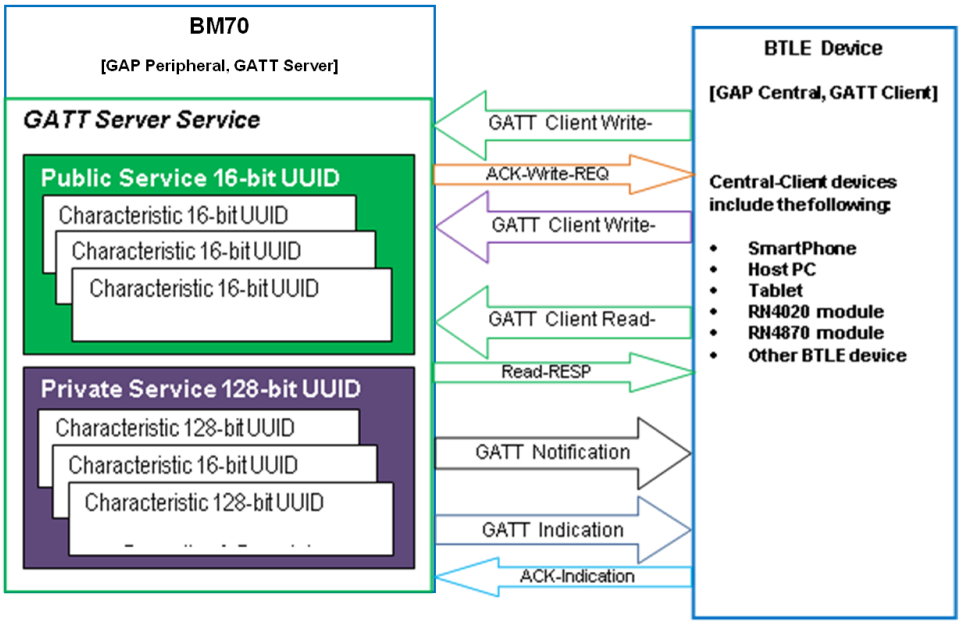


Figure A-9: Client (Peripheral) and Server (Central) Model in BLE. Source: https://microchip.wdfiles.com/local--files/wireless:ble-gatt-overview/gatt-example.png

Bluetooth LE coexists in the same 2.4 GHz ISM band as 802.11/Wi-Fi. As the advertising channels form the basis for how BLE operates, they have been assigned center frequencies that minimize overlapping with the most common 802.11 channels shown above. Channels 37, 38 and 39 are spread across the 2.4GHz spectrum. 37 and 39 are the first and last channels in the band, while 38 is in the middle.

Command Set for the BLE UART Service

- posX=%d
- posY=%d
- posZ=%d
- eulerX=%d
- eulerY=%d
- eulerZ=%d

Where %d is an integer corresponding to the sensor value.

BLEHidAdafruit Class

Source:

https://github.com/adafruit/Adafruit_nRF52_Arduino/blob/master/libraries/Bluefruit52Lib/src/services/BLEHidAdafruit.h

```
// Constructor
BLEHidAdafruit(void);

// Call this once to start the HID service
virtual err_t begin(void);

// Keyboard
err_t keyboardReport(hid_keyboard_report_t* report);
err_t keyboardReport(uint8_t modifier, uint8_t keycode[6]);
err_t keyboardReport(uint8_t modifier, uint8_t keycode0, uint8_t keycode1=0,
uint8_t keycode2=0, uint8_t keycode3=0, uint8_t keycode4=0, uint8_t keycode5=0);

err_t keyPress(char ch);
err_t keyRelease(void);
err_t keySequence(const char* str, int interal=5);

// Consumer Media Keys
err_t consumerReport(uint16_t usage_code);
err_t consumerKeyPress(uint16_t usage_code);
err_t consumerKeyRelease(void);

// Mouse
err_t mouseReport(hid_mouse_report_t* report);
err_t mouseReport(uint8_t buttons, int8_t x, int8_t y, int8_t wheel=0, int8_t
pan=0);

err_t mouseButtonPress(uint8_t buttons);
err_t mouseButtonRelease(void);

err_t mouseMove(int8_t x, int8_t y);
err_t mouseScroll(int8_t scroll);
err_t mousePan(int8_t pan);
```

BLEUart Class

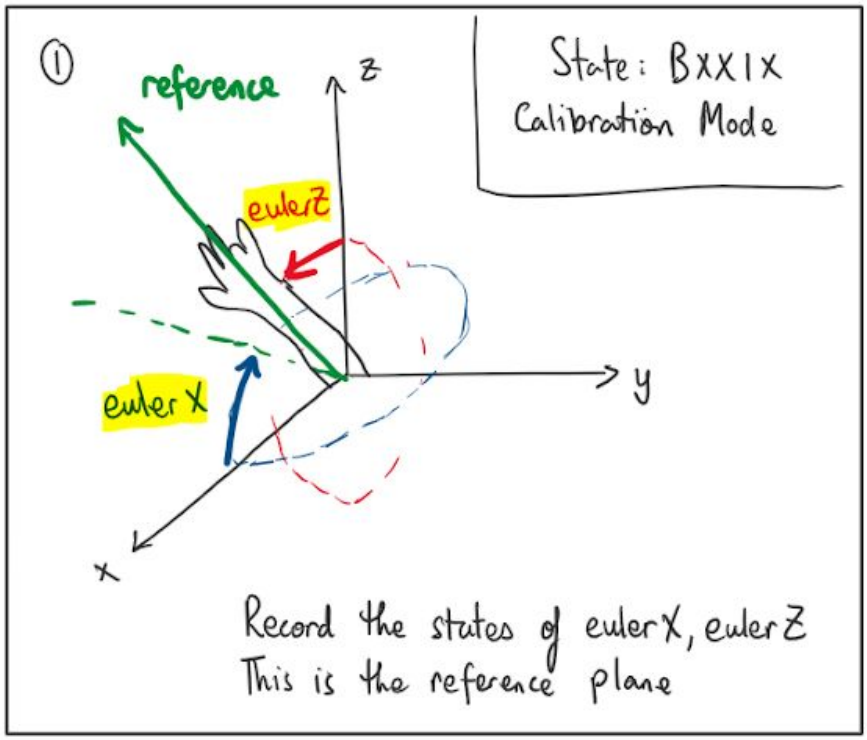
Source:

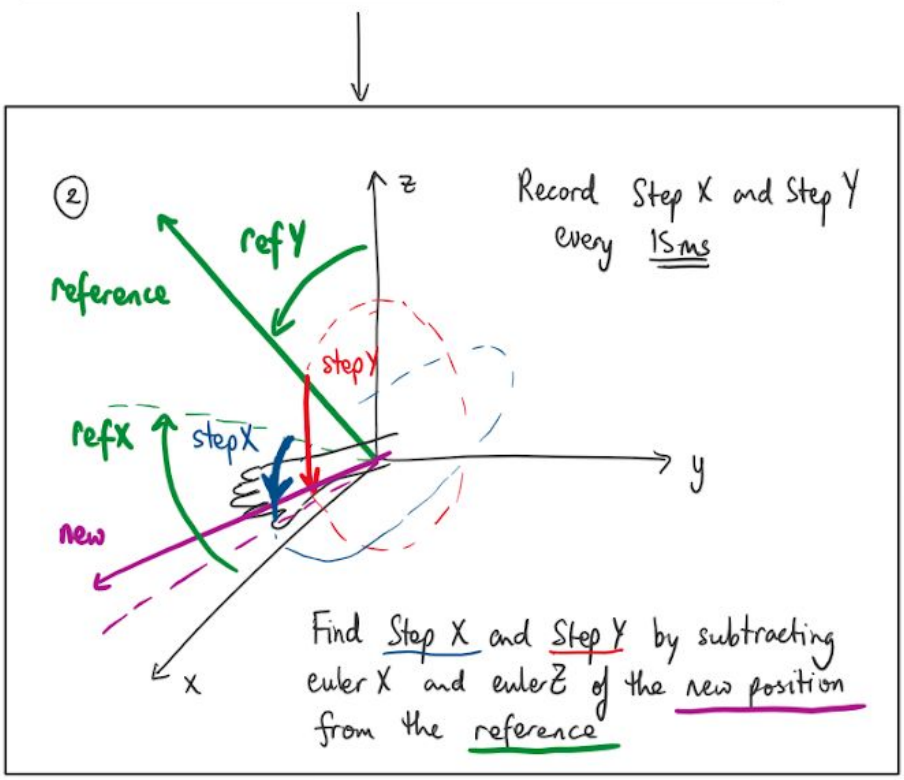
https://github.com/adafruit/Adafruit_nRF52_Arduino/blob/master/libraries/Bluefruit52Lib/src/services/BLEUart.h

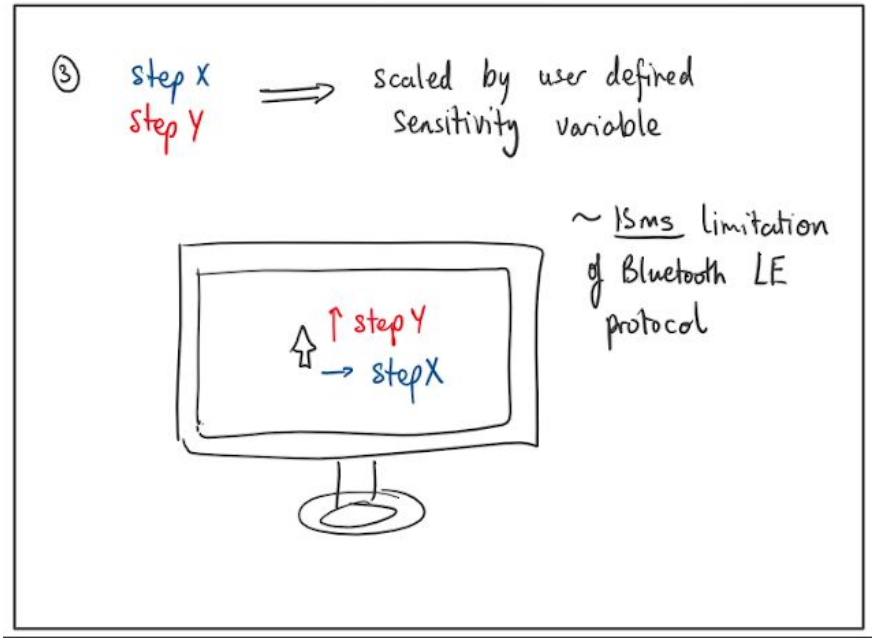
```
// RX Callback signature (fires when data was written by the central)
typedef void (*rx_callback_t) (void);

// Constructor
BLEUart(uint16_t fifo_depth = BLE_UART_DEFAULT_FIFO_DEPTH);

virtual err_t begin(void);

bool notifyEnabled(void);

void setRxCallback( rx_callback_t fp);

// Stream API
virtual int       read       ( void );
virtual int       read       ( uint8_t * buf, size_t size );
virtual size_t    write      ( uint8_t b );
virtual size_t    write      ( const uint8_t *content, size_t len );
virtual int       available  ( void );
virtual int       peek       ( void );
virtual void      flush      ( void );

// Pull in write(str) and write(buf, size) from Print
using Print::write;
```

Control Schemes - Drawings

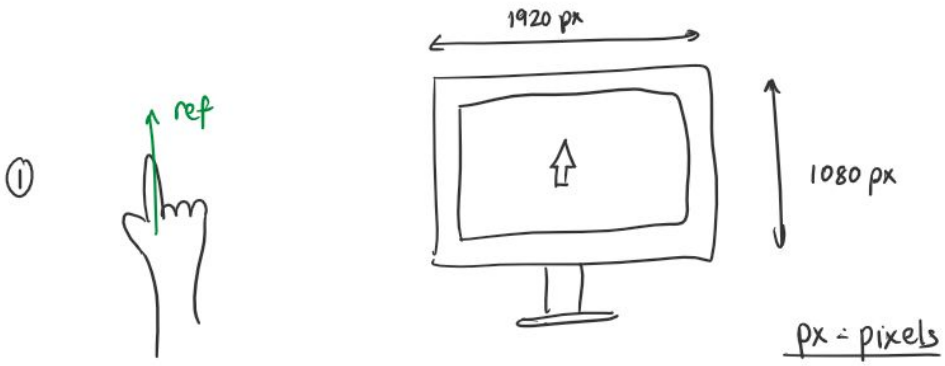
Moving Mode




②
z
Record Step X and Step Y every 15ms
refY
reference
stepY
refX
stepX
y
new
x
Find Step X and Step Y by subtracting euler X and euler Z of the new position from the reference


Every 15ms
③
step X
step Y
scaled by user defined Sensitivity variable
~ 15ms limitation of Bluetooth LE protocol
↑ step Y
→ stepX

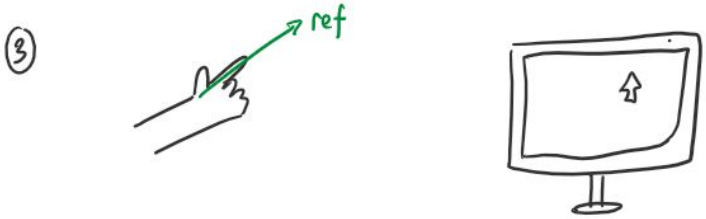
Pointing Mode



Restrict Movement to $\pm 90^\circ$ in each axis

$90^\circ = \pi/2$ radians : $1920/2$ px → X
: $1080/2$ px → Y

ptMove Scale X = $(1920/2) / (\pi/2)$ px/radian
ptMove Scale Y = $(1080/2) / (\pi/2)$ px/radian



Budget and Resources

The team budget remained close to the proposed values with minor changes as outlined below:

Item	Vendor	URL (if available)	Price (CAD)	Proposed Qty	Finall Qty
FEATHER NRF52 BLUEFRUIT LE	Digikey Canada	https://www.digikey.ca/products/en?mpart=3406&v=1528	$33.72	2	3
USB 2.0 A MALE TO USB 2.0 MICRO	Digikey Canada	https://www.digikey.ca/product-detail/en/qualtek/3025030-03/Q966-ND/6188812	$5.23	3	3
THREAD 316L THIN COND 3PLY 60'	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/641/1528-1268-ND/5356753	$10.00	1	1
CONN HEADER .100" SNGL STR 40POS	Digikey Canada	https://www.digikey.ca/products/en?mpart=PRPC040SFAN-RC&v=35	$1.14	2	2
CONN HEADER .100" SNGL R/A 40POS	Digikey Canada	https://www.digikey.ca/products/en?mpart=PRPC040SBAN-M71RC&v=35	$1.24	2	2
FLORA ACCEL/GYRO/MAGN 9-DOF	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/2020/1528-1335-ND/5356820	$26.96	1	1
LiPo Battery Cell - 3.7V 110mAh	Robotshop Canada	http://www.robotshop.com/ca/en/lipo-battery-cell-37v-110mah.html	$6.35	1	1
Pressure-Sensitive Conductive Sheet	Robotshop Canada	http://www.robotshop.com/ca/en/pressure-sensitive-conductive-sheet-velostat-linqstat.html	$5.06	1	1
FLORA PLATFORM RGB NEOPXL V2 4PK	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/1260/1528-1310-ND/5356795	$10.74	1	1
Woven Silver Conductive Fabric 400cm²	Robotshop Canada	http://www.robotshop.com/ca/en/woven-silver-conductive-fabric-400cm.html	$6.35	1	1
MOD FLORA WEARABLE BLUEFRUIT LE	Digikey Canada	https://www.digikey.ca/products/en?mpart=2487&v=1528	$23.65	1	1
FLORA ELECTRONIC PLATFORM V2	Digikey Canada	https://www.digikey.ca/products/en?mpart=659&v=1528	$20.21	1	1
LEAD SET 10 MINI-ALLIGATOR 22AWG	Digikey Canada	https://www.digikey.ca/products/en?mpart=BU-00285&v=314	$9.41	2	2
SWITCH SLIDE SPDT 300MA 6V	Digikey Canada	https://www.digikey.ca/product-detail/en/apem-inc/MHSS1105/679-1849-ND/1949465	$0.73	5	5
VIBRATION MOTOR 3VDC	Digikey Canada	https://www.digikey.ca/products/en?mpart=3160400001&v=1597	$1.76	1	1

BREADBOARD PERMA-PROTO PCB SGL	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/1606/1528-1100-ND/5154675	$9.39	1	1
JUMPER WIRE M/M 40X6" 150MM	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/758/1528-1154-ND/5353614	$5.34	1	1
FEMALE/FEMALE JUMPER WIRES 40X6	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/266/1528-1379-ND/5629427	$5.34	1	1
JUMPER WIRE F/M 40X6" 150MM	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/826/1528-1162-ND/5353622	$5.34	1	1
LASER DIODE - 5MW 650NM RED	Digikey Canada	https://www.digikey.ca/product-detail/en/adafruit-industries-llc/1054/1528-1391-ND/5629439	$8.04	1	1
ANALOG 2-AXIS THUMB JOYSTICK WIT	Digikey Canada	https://www.digikey.ca/products/en?mpart=512&v=1528	$8.56	1	1
2.2" 10K Flexible Sensor	Robotshop Canada	http://www.robotshop.com/ca/en/22-10k-flexible-sensor.html	$9.99	10	1
BNO055 9 DOF Absolute Orientation IMU Fusion Breakout Board	Robotshop Canada	http://www.robotshop.com/ca/en/bno055-9-dof-absolute-orientation-imu-fusion-breakout-board.html	$44.81	1	1
3M 35-WHITE-1/2 Electrical Tape	Digikey Canada	https://www.digikey.ca/product-detail/en/3m/35-WHITE-1-2/3M15557-ND/1818756	$3.60	1	1
Cotton Glove	Peavey Mart		$15.00		1

Budget (CAD)	$600.00
Total Proposed	$421.76
Total Spent (CAD)	$470.48