Iron Man Mouse

ECE 445 Design Document

Team 5: Zhiyuan Yang (zy8), Yayati Pahuja (ypahuja2), Kai Chieh Chang (kcchang3)
TA: Feiyu Zhang
09/26/2021

1. Introduction

1.1 Problem Overview

We often find ourselves tied to the monitor/mouse system while giving a presentation or in a group working session. This could limit our delivery in presentation due to audience boredom and could limit group workflow by blocking the monitor. It would be a lot cooler and more efficient if we could hover our cursor and control the computer system with our hands in the air just like Iron Man. It is also a problem that sometimes we want to have more functionalities packed in our mouse to be more productive. If we can add a depth/3rd dimension to using a mouse by bringing it into the air and expanding user work/input space, we could open up a wide range of possibilities.

1.2 Solution Overview

We aim to create a mouse that works in 3 dimensions in the form of a glove. As visualized in figure 1, The device would have a wrist band portion that acts as the tracker of the mouse pointer (implemented by 3-axis accelerometer), and 3 gyroscopes attached to 3 fingers to collect finger bending data to simulate mouse clicks. A microcontroller (MCU) on the glove would translate these translational/rotational data to imitate a mouse or trackpad movements with possible custom operation. Then the MCU on the glove would send the signal to a PC via bluetooth (BT). The transmitted data would be in the form of Human Interface Devices (HID) reports to have easy interaction with the Windows operating system.

1.3 Visual Aid (Physical Design)

Figure 1 visually shows how our design can solve the above problem. The main PCB (including the MCU, BT, and accelerometer) would be located on the back of the glove near the wrist. As one moves his/her hand, the accelerometer would take in that translational information and convert it into cursor movement to send to the PC through BT. On each of the first 3 fingers, there would be gyroscopes that can detect how fast one is bending his/her fingers. The MCU can then be customized to translate different finger movements as different PC control signals (i.e. a flick on the index finger can be translated to a mouse left click), and to send these data via BT.

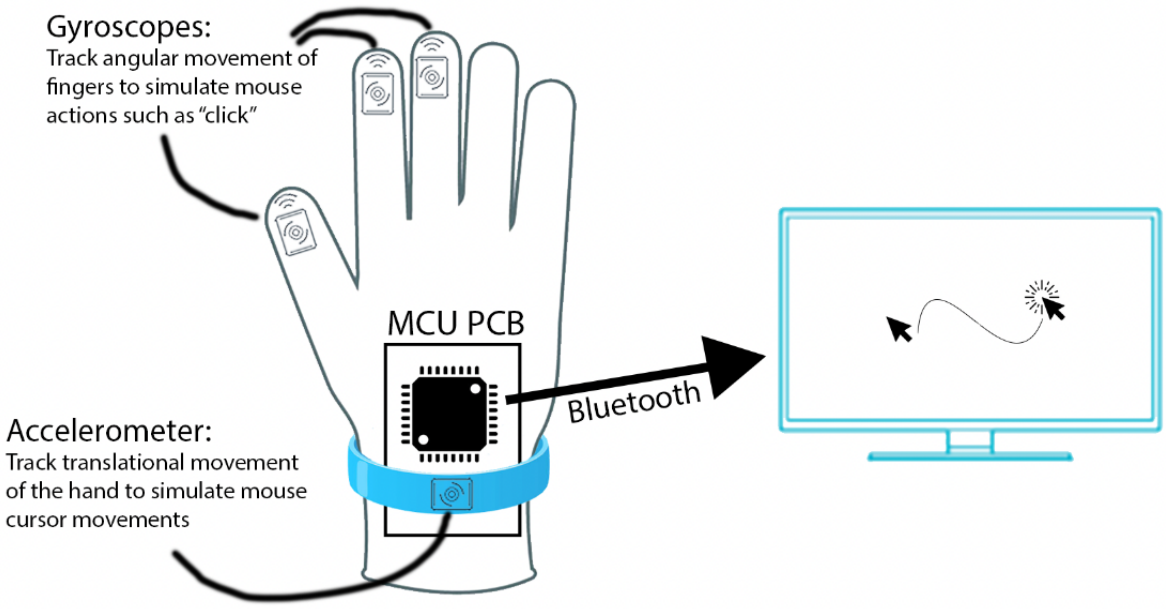

Figure 1. High Level Visual Aid of Iron Man Mouse

1.4 High Level Requirements

1. Support basic mouse functionalities such as click, scroll, cursor movement as well as some advanced functionalities such as zoom in/out, volume control.
2. The mouse should achieve at least 70% target efficiency (total hits/targets) and at least 60% click accuracy (total hits/clicks)
3. The device would have a minimum polling rate of 125Hz, which corresponds to at most 8ms response time.

2. Design

2.1 Block Diagram

Figure 2 shows the high level block diagram of our design. It has a power supply module that is capable of providing 3 separated 2.5V, 3.3V, 3.3V power signals from a rechargeable battery to the accelerometer, gyroscope, and MCU system respectively. This design could reduce noise between the separated systems and improve target efficiency and click accuracy (requirement 2) from the sensor level. The MCU system can then take the data from the sensors, process the data (helps to achieve requirement 2), and convert the information into HID data for the PC to recognize as mouse actions (requirement 1). Then the MCU would send all the HID information via bluetooth to the PC system as output. The algorithm and implementation of all these operations in the MCU system is critical for fast data processing (requirement 3).

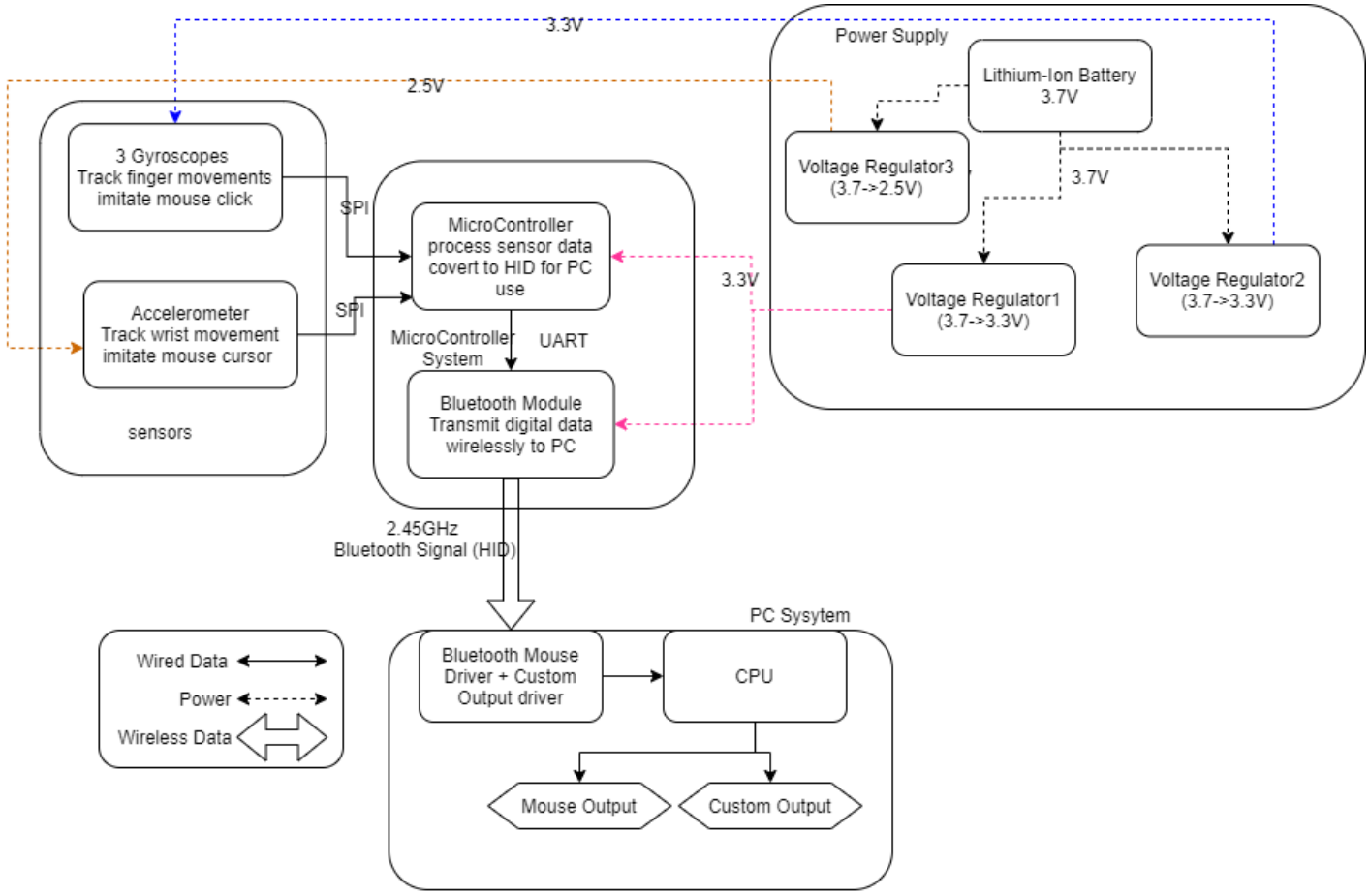

Figure 2. Block Diagram

2.2 Physical Design

2.3 Power subsystem

In order for each sub system to function stably, we would need a stable power supply system. The power subsystem consists of one battery and three voltage regulators. The battery that we chose can provide 3.7V with a capacity of 400mah, and the three voltage regulators would convert the battery voltage down to a stable 3.3V, 3.3V, and 2.5V respectively. We wish to provide stable voltage and current across all devices. Since our devices' normal operation voltages (3.3V) are close to 3.7V, in the event of the battery discharges under the normal operation voltage, the power system should still be able to provide a voltage as close to 3.3V as possible. As a result, we chose to design this subsystem with linear low dropout voltage regulators (LDO). The first voltage regulator will provide 3.3V to the microcontroller and the bluetooth module. The second voltage regulator will provide 3.3V to the three gyroscopes that are the finger sensors. The last voltage regulator will provide 2.5V to the accelerometer. The power subsystem is designed this way to reduce interactions among subsystems, and reduce potential noise from the sensors. This design choice should be able to improve our data accuracy as described in requirement 2.

2.3.1 Polymer Li-ion Rechargeable Battery

The model we choose is DTP502535, which has nominal voltage of 3.7 V and nominal capacity of 400mAh. The battery has a standard discharge of 80mA and maximum discharge of 400mA, which should be enough to power all the devices: the sensors have a maximum current consumption of 8mA, but the microcontroller can drain up to 100mA at maximum and bluetooth module 50mA at maximum (total to around 158mA in the worst case). With a less optimistic approach, on average our devices will drain up about 70 to 80 mA current (158mA/2). The battery should give our mouse 5 hours of use (without any power saving design). Apart from current, we need to ensure the battery can supply enough voltage to keep the devices on even when it is almost discharged. We therefore chose the battery that has a charge limited voltage of 4.2V and discharge cut-off voltage of 2.8V, which is well within the operation range of MCU and bluetooth module.

Requirements	Verifications
1. The voltage range of the battery should be between 2.0V and 5.5V (5.5 V is the highest voltage regulators can take and 2.0V is the lowest voltage for	a. Fully charge the battery at 4.2 V. Connect an ammeter in series with the charger and battery. Wait until the initial charging current of 80mA

sensors and control unit to function.)	reaches 4mA, indicating that the battery is fully charged. b. Apply a variable resistor parallel to the battery and set it to be around 50 Ω. c. Measure the voltage of the battery and the current of the circuit using an oscilloscope and change the variable resistor to maintain a 80mA current. d. Record the voltage values until the battery fully discharges and see if the values are within the specified range.
2. Require that under the average current drain of 80mA, the battery should be able to operate for at least 5 (- 10%) hours.	a. In the previous verification, use a stopwatch to record the time it takes to fully discharge the battery. See if the time verifies the requirement.

2.3.2 Voltage Regulators

In our design, we use 3 LDO regulators and they are the same family model LP5907. The reason to use multiple voltage regulators is to reduce noise of the sensors' data by separating sensors' power supply from the control unit's power supply. The criteria of choosing the regulators include compatibility with the battery and the devices. Each of the regulators has a maximum current output of 250mA which is more than enough for the devices (calculated to be 158mA in 2.3.1). The MCU and bluetooth module uses a maximum of 150mA current and the sensors use much less than that. On the battery compliance side, they have max input voltages of 5.5V which well incorporates the maximum voltage of the battery (4.2V).The LDOs have a typical dropout voltage of 120mV, which means that the input voltage has to drop below 3.42V for the output voltage to drop below 3.3V. When our battery voltage drops below the nominal voltage (3.7V), the low dropout voltage features allow the regulators to provide stable 3.3V voltages for long periods. In addition, the enable pin feature allows for more convenience in power regulation. In figure 3 below, we incorporated decoupling capacitors in the design to further reduce noises.

Requirements	Verifications
1. Provides fixed 3.3V / 2.5V (+/- 5%) in the range of 0 to 200mA (maximum current drain from MCU/BT) output current when input is between 3.6V to 4.0V (within	a. Set up the regulator circuit; Connect a variable power source (initialize to 3.6V) to the regulator input. b. Increase the input voltage by 0.05V

battery range)	c. Measure the open circuit voltage of the output voltage of the regulator (0mA) d. Connect a variable resistor (initialize to Z) between the regulator output pin and ground; measure the current with an oscilloscope. e. Measure the regulator output voltage while decreasing the resistance (until the output current reaches 200mA). Record the numbers f. Repeat step 2~5 until input voltage=4.0V
2. Measure the dropout voltage to be ≤ 250mV, when the output current is between 0~200mA (maximum current drain from MCU/BT).	a. Set up the regulator circuit (battery parallel with a variable resistor), and input voltage of 3.65V. b. Measure the current with an oscilloscope c. Measure the regulator output voltage while decreasing the resistance (until the output current reaches 200mA). See if the output voltage changes. d. Lower the input voltage by 0.02V and repeat previous steps until input voltage is 3.55V.
3. Measure the quiescent current to be below 25uA (typical 12uA) and ground current to be around 14uA when output current is 0mA. (low power consumption)	a. Set up the regulator circuit with an open output circuit. Measure the current going in the regulator and the current going out the regulator using an oscilloscope.

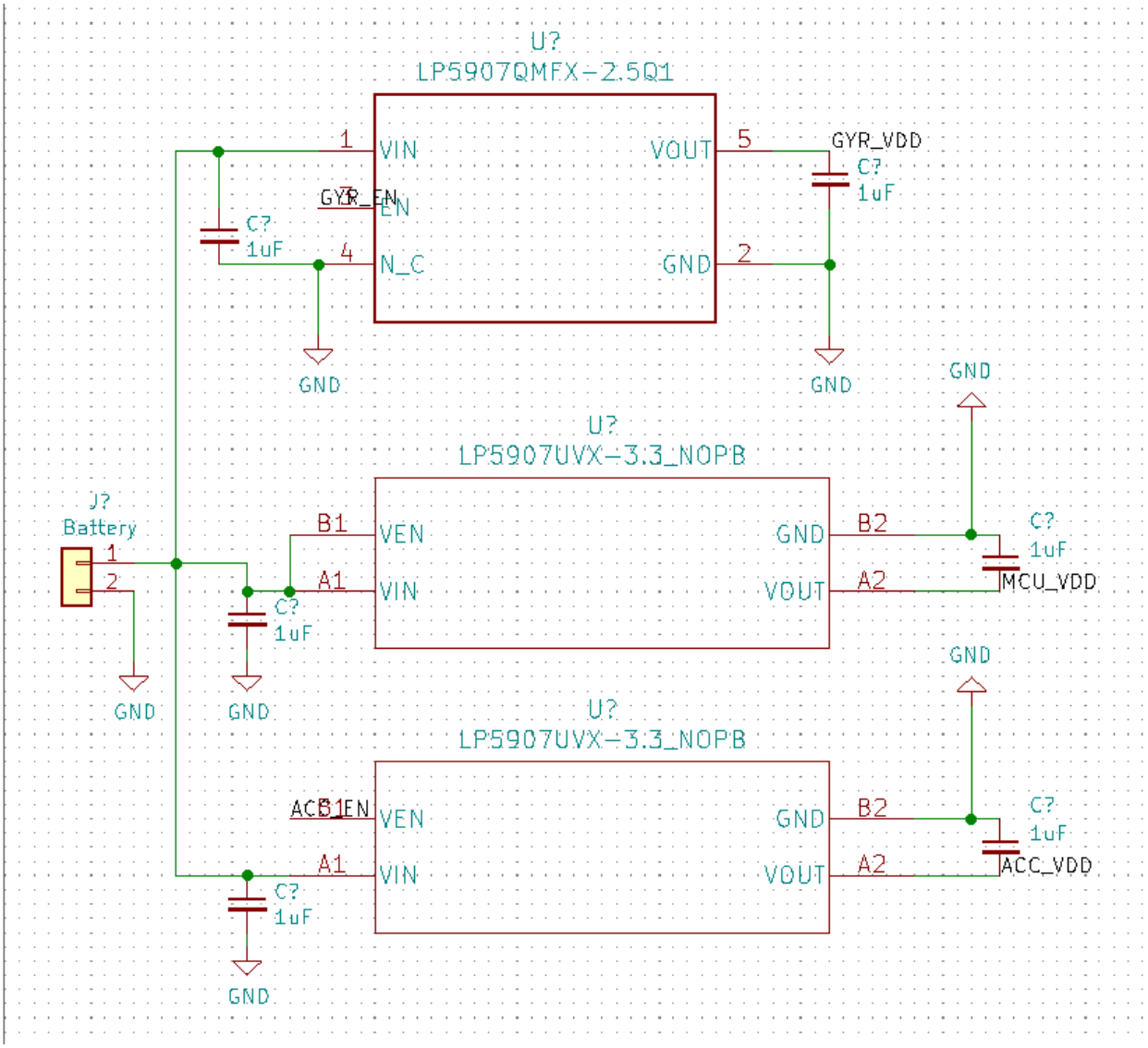

Figure 3. PCB Schematic for Voltage Regulators

2.4 Control System

The control system consists of an STM32L152RET6 microcontroller unit (MCU) along with its peripherals and a bluetooth module. The MCU, powered by the power supply in section 2.3, would be the center to process data from the accelerometer and gyroscope through SPI communication, converting information to useful mouse data. The MCU then transmits data to a bluetooth module (RN-42) via UART, leaving the rest of the work to be completed by the PC driver to visualize our hand movements as a mouse cursor.

2.4.1 Microcontroller Unit (STM32L152RET6)

The STMicroelectronics MCU, STM32L152RET6, is a ultra-low-power 32-bit Arm-Cortex-M3-Based MCU, with on-chip support for various common applications such as USB, SPI, ADC, and DAC. The reason for this choice was a combination of power consumption and performance to the nature of the project. In order for the Iron Man Mouse to work for more than 2 days of active use (~24hr), we need a chip with customizable power to save as much battery life as we can. The MCU has multiple power modes such as 290nA Standby and 11uA low-power run mode that can be customized when the mouse is inactive. In order for superb accuracy and low latency, we need a MCU with high clock/communication speed as well as fast computation ability.

The MCU would be powered by the power supply with 3.3V, take in digital SPI inputs from 1 accelerometer and 3 gyroscope, compute the cursor location and associate commands from user inputs, then transmit data to the bluetooth module. In figure 4, we also show the associated peripherals in the schematic for programmability and usability. This includes:

- Off-chip high speed resonator to boost the clock speed to at least 16MHz
- USB type B interface for programming the chip
- Reset button for calibration
- Open wakeup pins for sleep/operation mode
- General IO connectors to increase scalability

Requirement	**Verification**
1. Operate at a clock speed greater than 8MHz for effective data processing	a. Let the MCU output the system clock signal onto a generic microcontroller clock output pin (MCO) b. Use an oscilloscope to measure the frequency.
2. Minimum of 10 Mbps for SPI communication (both TX and RX) to meet the minimum of 125Hz mouse polling rate in the high level requirements $125Hz * (24\ bps/accelerometer$	a. In the PCB design, have the SPI Master SCLK pin connected to a measurement net b. Connect an oscilloscope to that measurement pin to verify SPI clock speed (which determines SPI data transmission rate)

$+ 16\,bps/gyroscope * 3) = 9Mbps$	
3. Capable of customizing power modes (minimum of 2 modes: sleep/active) to reduce power consumption through software	a. Use an LED on the PCB to indicate active and sleep status of the sensors/MCU b. Have a current meter connected between battery and Vdd on the MCU, and verify the LED does switch current consumption with different modes.
4. At least 1 SPI master and 4 chip select for all the sensors acting as SPI slave	a. Write a simple serial console output of the 3 gyroscope outputs and the 1 output of the accelerometer. b. As long as each sensor can change these 6 values then it is guaranteed that the SPI connection is made and the requirement is met.

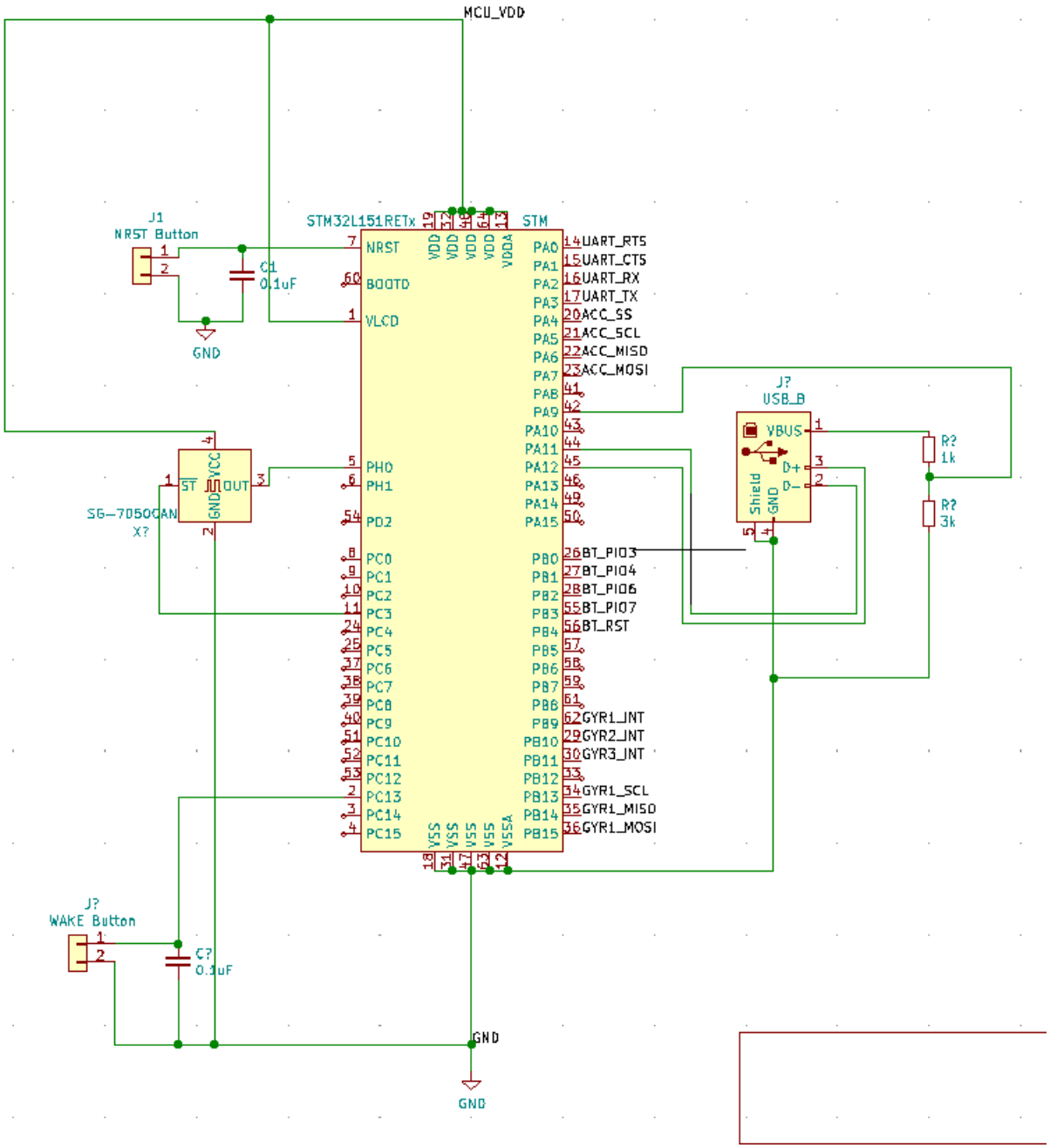

Figure 4. MCU Schematic

2.4.2 Bluetooth Module (Bluetooth SMD Module RN-42)

The Roving Networks RN-42 Bluetooth module is used to transmit the mouse data to the PC using the default HID profile over UART serial communication. The reason for this choice is that the module has multiple user configurable power modes including a sleep mode which uses only 26uA, which would allow for low power consumption while also giving a max range of about 50-60 feet, and a max data transfer rate of 3Mbps. This would satisfy our requirement of a long battery life as well as provide the convenience of controlling the mouse from a decent distance from the PC. Since the module is also HID compliant, and can be programmed to appear as a HID mouse in the discoverability menu, we can eliminate the need of a custom driver and report the mouse functionalities to the PC using HID reports that can be automatically processed by the PC as mouse input.

Requirement	**Verification**
1. Overall mouse latency should be around 8ms +/- 10% considering there might be some added latency due to the bluetooth connection.	a. https://www.vsynctester.com/testing/mouse.html This website can be used to test the input latency of a mouse. b. Perform the test to confirm if the latency is around 8ms.

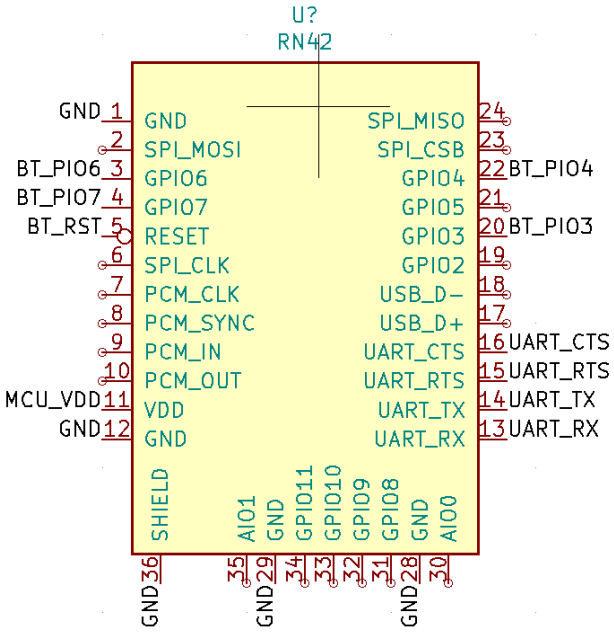

Figure 5. RN42 Schematic

2.5 Sensors

The sensor subsystem consists of an accelerometer that is on the wrist and three gyroscopes that are on the fingers. The accelerometer would collect translational acceleration, which we can use to calculate mouse cursor position. The angular velocity data from gyroscopes can help detect finger bends and simulate mouse operations such as clicks and zoom in. They are all digital sensors; the accelerometer communicates with the MCU using SPI/I2C; the gyroscopes communicate using SPI only. They are powered by the power subsystem described in 2.3. We decided to connect all the sensors to MCU using SPI. With digital interface, we should be able to reduce transmission noise from traditional analog sensors and hence increase our accuracy rating in requirement 2. Three gyroscopes would connect to one SPI master on the MCU and the accelerometer would connect to another SPI master on the MCU.

2.5.1 Accelerometer

The accelerometer chosen is LIS2DE12, which is a digital-output, low-power, and three-axis accelerometer. It is connected to the PC using SPI protocol. The accelerometer has a variable range for the user to select. We are planning on using +/-4g because hands could move rather quickly. It has an eight-bit data output for each axis. Therefore, the sensitivity of the accelerometer is around 31.2 mg/digit, which would give us fairly accurate data. It is capable of measuring data up to 5.3kHz, which is more than enough to satisfy the response time requirement (8ms/125hz). It is low in power: 6uA active use, 0.5uA sleep mode, and this is great on the battery life of the device. The device also has programmable interrupts which can be generated when wake-up/certain movement events (which we can configure) are detected. This is particularly useful when it comes to power consumption and setting a threshold acceleration value for the device to turn on.

Requirements	Verifications
1. The data rate output has to be at least 1kHz. (To achieve 125Hz polling rate of the device, we can maybe average out some acceleration values to be more accurate)	a. Connect accelerometer to MCU through SPI b. Configure the first four bits of CTRL_REG1 to be 1000, which selects the data rate of 1.62kHz c. Read data from the accelerometer and write the data to a usart connector d. Hook up the usart connector to an oscilloscope to display the data and

	check the data rate.
2. The position data measured through the accelerometer at scale (+/- 4g) has to be within +/- 10%.	a. Set up the connection between accelerometer and MCU through SPI b. Move the accelerometer across the same physical distance at 5 different speeds (therefore different accelerations). c. Read data from the accelerometer, and use the conversion algorithm we develop below to calculate the position change. d. Compare 5 position changes and see if they are within 20% of each other.

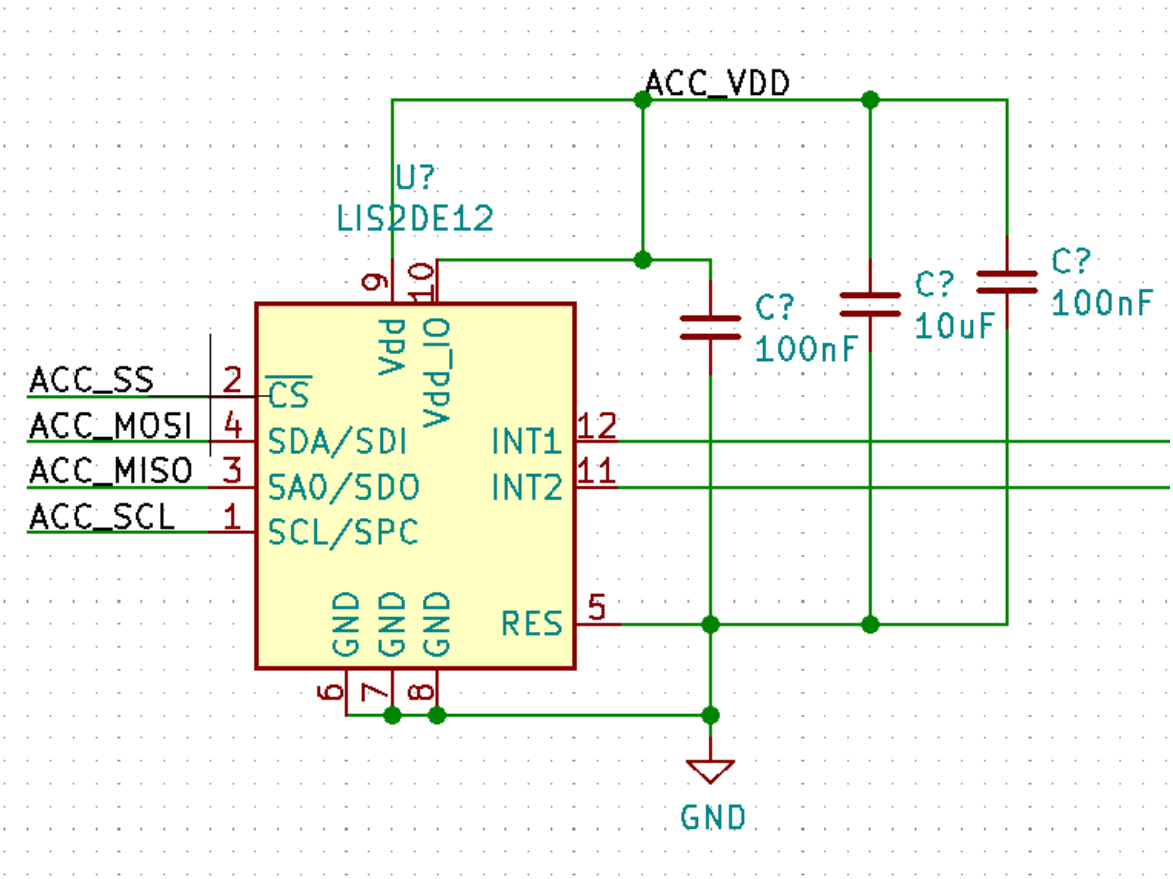

Figure 6. PCB Schematic for Accelerometer

2.5.2 Gyroscope

The gyroscope chosen is ICG-1020S, which is a digital-output, high-resolution, and dual-axis gyroscope. It is connected to the PC using SPI protocol. It has 16-bit adc output word length and a variable measuring range up to +/- 374 degrees/s, which we will use. In the paper “The statistics of natural hand movements”, we found the distribution of angular speeds of the index finger.

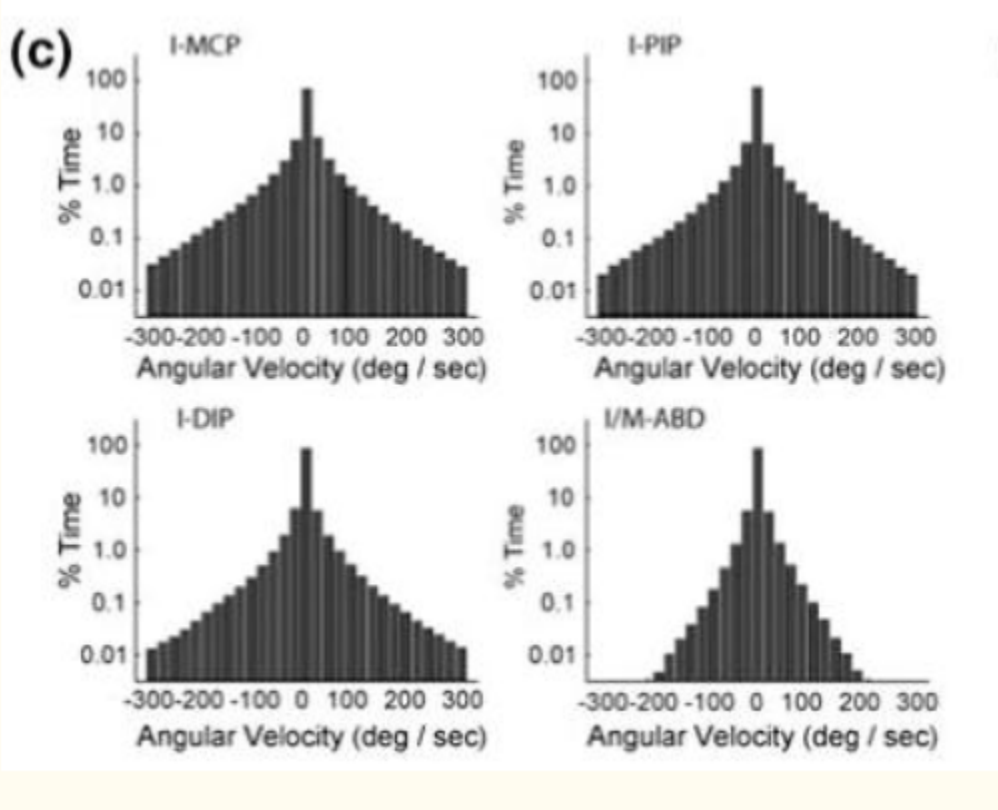

Figure 7. Index Finger angular velocity distribution [5]

We can see that most of the time the velocity is within 100 degrees/sec, but it can go to 300 degrees/sec on rare occasions. The 16-bit adc word length would give us fairly accurate data even in those rare occasions. It is capable of measuring data up to 32kHz, which is more than enough for the fast response time requirement. We can even reduce the sampling rate to save power. It has 2.5 mA current consumption in normal mode and 6uA in sleep mode, which is low enough to not have a big impact on battery life. Since it is a two-axis gyroscope, we can not only measure the clicking movement of fingers but also the wrist rotation movement. This allows us to develop more advanced functionalities.

Requirements	Verifications
1. The data rate output has to be at least	a. Connect the gyroscope to MCU via

1kHz. (To achieve 125Hz polling rate of the device, we can maybe average out some angular velocity values to be more accurate).	SPI. b. Configure register 25, the sample rate divider register to get 1kHz or 2kHz of sampling frequency. c. Read data from the gyroscope and write the data to a usart connector d. Hook up the usart connector to an oscilloscope to display the data and check the data rate.
2. Measure angular velocities at scale (+/- 374 degrees/sec) to the precision of 5% (+/- 2.5%)	a. Set FS_SEL in GYRO_CONFIG to 3 to select the +/- 374 degrees/sec range b. Put the gyroscope on a mechanical device that is rotating at a known and fairly constant angular velocity. We can also calculate the angular velocity ourselves if we do not know the angular velocity. We can take the time of 10 revolutions and calculate the average angular velocity. c. Compare the average value measured by the gyroscope to the real one; see if the precision is within 5%.

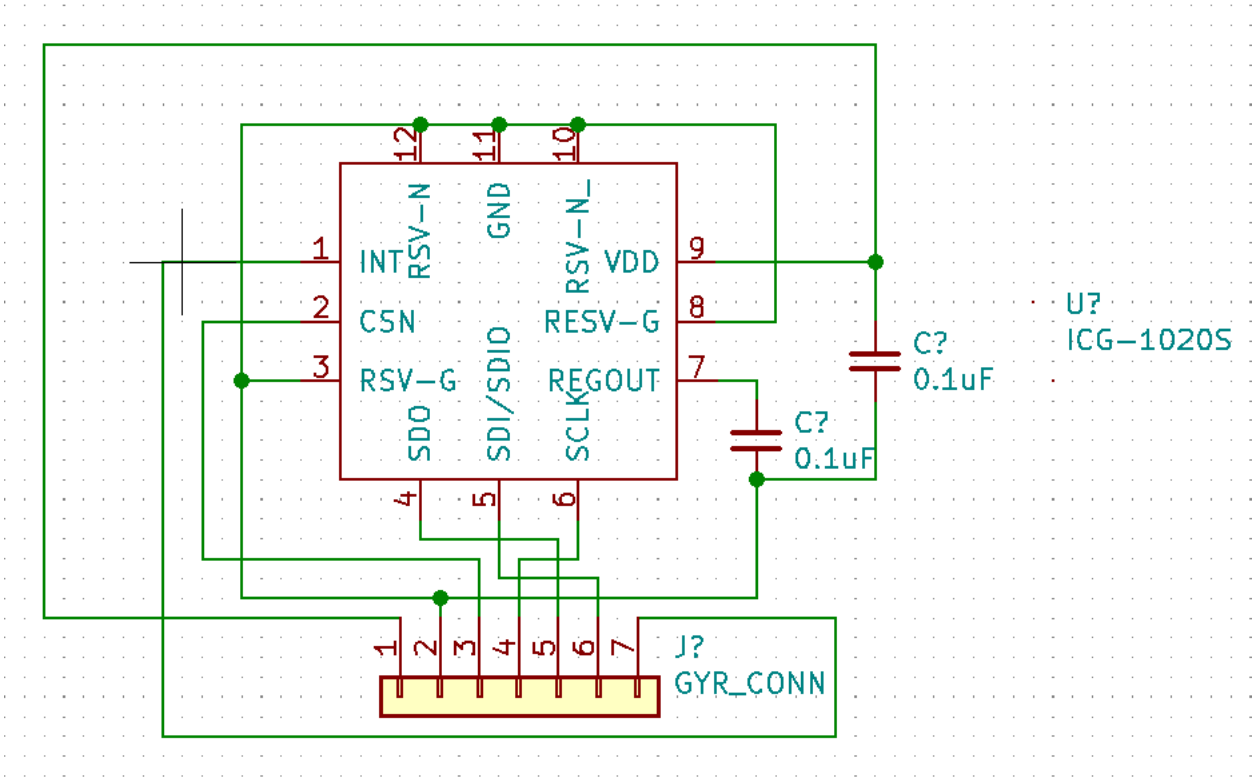

Figure 8. PCB Schematic for a Gyroscope

2.6 Software

2.6.1 PC Driver (HID Protocol)

Since the bluetooth module being used is HID compliant, it can be configured to show up as a Bluetooth Mouse in the discoverable devices on a bluetooth enabled PC. All modern mainstream Operating Systems recognize HID devices without needing specialized drivers. This means we can directly send HID descriptor reports to the PC without the need for any custom drivers. The mouse data can be sent using the HID raw mouse report format, which is as follows according to the RN42 User guide [4]:

0xFD	5	2	Buttons	X-stop	Y-stop	Wheel

Figure 9. HID mouse descriptor report format

Here, the Buttons, X-stop, Y-stop, Wheel are represented by 1 byte each. The Buttons byte signifies what buttons (Left, Right, Middle) were clicked, X-stop and Y-stop bytes signify the delta-x and delta-y values for the cursor movement and the Wheel byte signifies how much the scroll wheel was moved.

Requirements	Verifications
1. Check if the data being sent is correct and complete so that no data packets are being lost. Each report should be 7 Bytes according to the HID report format.	a. Display read data on a Serial monitor to check the number of bytes read and the data itself. b. Confirm format and values.

2.6.2 Overall Data Path

The sensors are used as the input sources by the user to control the mouse. The sensor data is read by the microcontroller, processed into HID reports and sent to the PC through the Bluetooth module. The overall data path through our device is as follow:

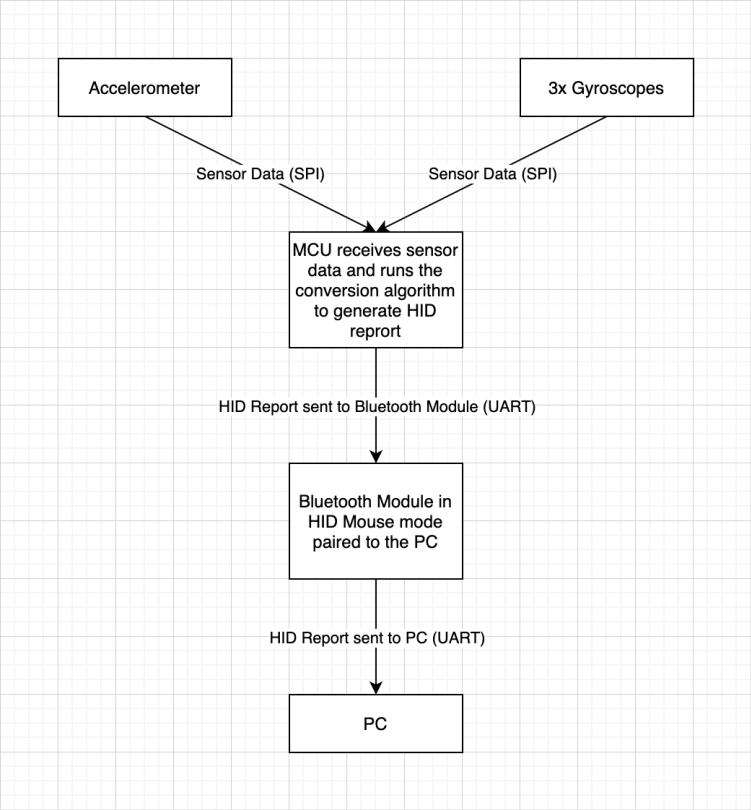

Figure 10. Data Path

2.6.3 Accelerometer Conversion Algorithm

The accelerometer is used to control the cursor movement. To generate the X and Y movement values we need an algorithm to convert the sensor data into the 2 bytes of HID data. We begin by calibrating the sensor to account for any erroneous values generated by the sensor hardware. The sensor data is then read and corrected using the calibration offset values followed by noise removal to account for small motions not intended to act as input. Finally, the resulting acceleration data is used to generate the delta X and Y values using single integration of the acceleration data. The general algorithm is as follows:

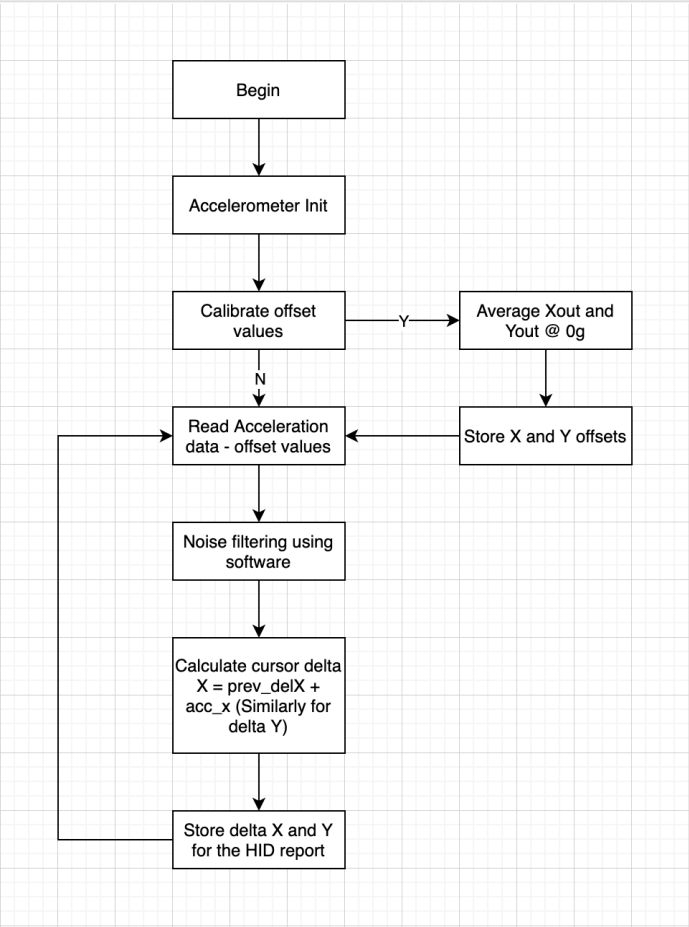

Figure 11. Accelerometer conversion algorithm

2.7 Tolerance Analysis

2.7.1 Background

The most critical part of our design is the accuracy of mouse clicks as described in high level requirement 2. In order to accurately imitate a mouse on a glove, we need:

1. the sensors to accurately capture the analog signal from user movement
2. the MCU to correctly translate this information into HID reports for the PC to understand
3. the BT module to send this information reliably wirelessly.

In our design process, we have partially addressed problem 1 and 3. We have chosen digital sensors to reduce noise potentially introduced in analog transmission. However, there is still risk in the process. Depending on how the MCU processes the raw sensor data, we could potentially include noise that is detrimental to accuracy. This is especially important for the accelerometer data, since it is the only part that requires consistent polling throughout the active time of the glove. All subsequent functionality of clicking, scrolling, etc all depend on our accelerometer to provide the correct cursor location. We are solely using the data from one three-axis accelerometer to calculate the location data for the mouse cursor, which is the most basic functionality of a mouse. Therefore, we need a tolerance analysis on the accelerometer data to figure out how much error we can accommodate from the sensor, while retaining the same accuracy as specified in requirement 2.

2.7.2 Tolerance Analysis Statement

A digital accelerometer can sample the acceleration on 3 axes in the unit of g ($9.81 m/s^2$) with some errors. We wish to calculate the maximum error tolerable of a sensor (while maintaining the 125Hz polling rate / 8ms response time) given the following parameters.

$$w\,[bits] = data\ width\ of\ each\ accelerometer\ sample\ on\ 1\ axis$$
$$\pm\ R\,[g] = sensor\ detection\ range$$
$$e\,[\%] = error\ of\ the\ accelerometer$$
$$f\,[Hz] = sampling\ rate$$

The related variables are:

$$a\,[m/s^2] = acceleration$$
$$v\,[m/s] = velocity$$
$$x\,[m] = position$$

2.7.3 Accelerometer data processing

Mouse is a relative device. Each mouse event reports an update on the old location. Our job is to calculate delta position (Δx) given acceleration values. We plan to implement a position algorithm by Kurt Seifert and Oscar Camacho.

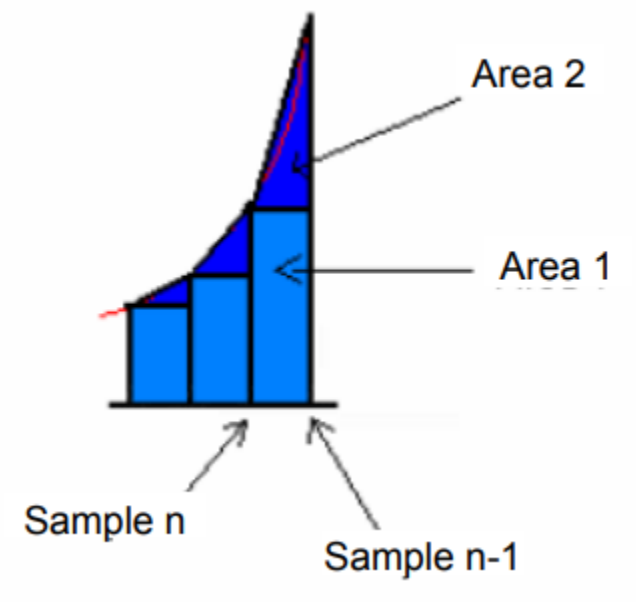

Figure 12. Acceleration integration [2]

Position is acceleration integrated twice. Integration is basically calculating the area below the curve. We are using the rectangle and triangle method in the figure above to calculate the area. $Area_n = Sample_n + \frac{|Sample_n - Sample_{n-1}|}{2}$. The delta time term is omitted, because we assume uniform sampling and delta time is 1.

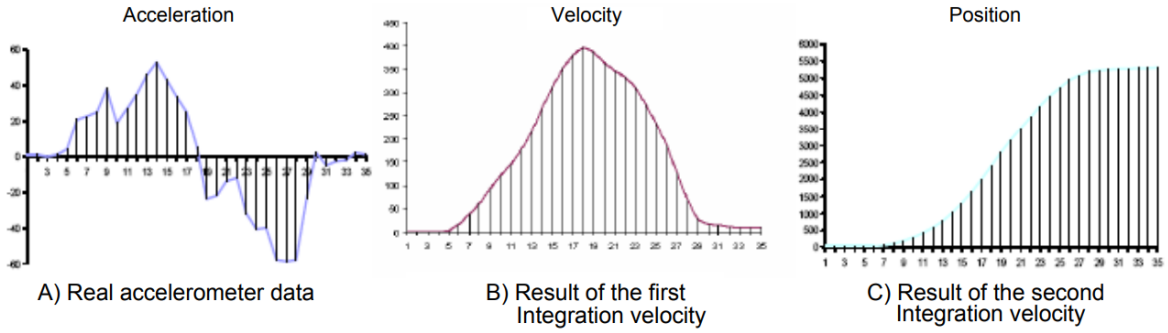

Figure 13. Double Integration steps[2]

Therefore, velocity and position values can be calculated as follows:

$$v[1] = v[0] + a[0] + ((a[1] - a[0]) / 2)$$
$$x[1] = x[0] + v[0] + ((v[1] - v[0]) / 2)$$
$$\Delta x = x[1] - x[0] = v[0] + ((v[1] - v[0]) / 2)$$

An error on acceleration term will contribute to an error on velocity and also an error on position. There might be noise on the sensor. In this case, we want all the noise signals

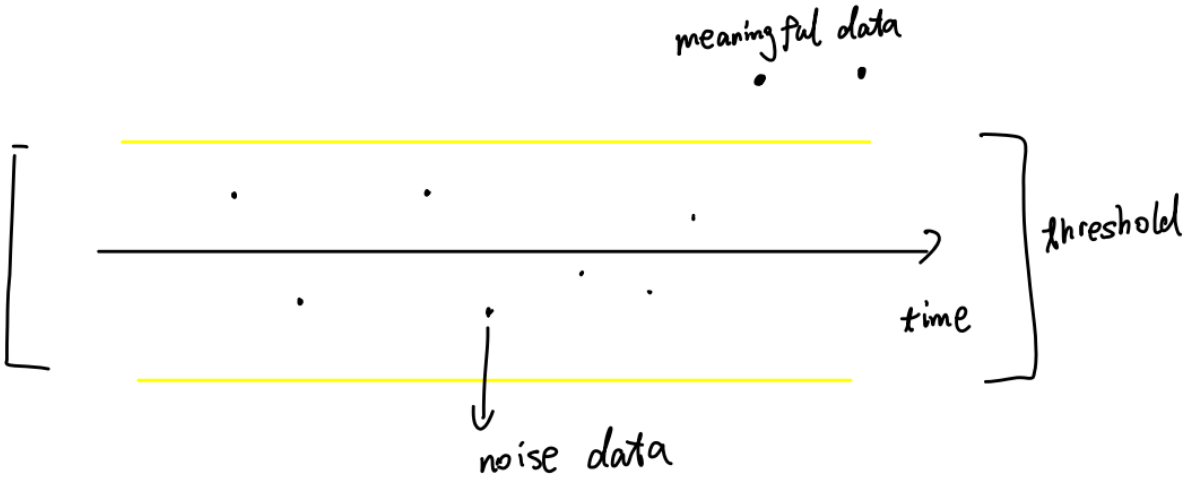

Figure 14. Double Integration steps

contained in a threshold to not affect meaningful data collection. The data inside the threshold should be zeroed digitally. Supposing that the minimum acceleration to detect motion is a quarter of a g ($2.5\ [m/s^2]$), then we need the error on acceleration to be within $\pm\ 2.5\ [m/s^2]$. The absolute error on acceleration values has two parts. The first one is the zero-g offset, and the second one is the error due to the limited sensitivity of the sensor. Therefore, the absolute error can be expressed as

$$E_{max} = MAX(e_{offset}, \frac{R}{2^{w-1}})$$

In order to retain accuracy, the error E of the sensor chosen in mass production needs to be less than $2.5\ [m/s^2]$. In our prototype, the sensor we choose has a zero-g offset (stationary) of +/- 100mg, which is $\pm\ 0.9\ [m/s^2]$. Its noise will be contained in the threshold.

In the second diagram of figure 13, we see that the errors from acceleration build up to a non zero velocity in the end even though acceleration has gone to 0. We plan on accommodating this issue by setting a threshold velocity, so when velocity drops below, we would calibrate the sensors. This approach would make choosing a sensor easier as described below.

In the first line of the algorithm, we would introduce a $\pm\ 2E$error from the accelerometer data. In the second line, this error would double to $\pm\ 4E$. However, based on the central limit theorem, as we sample from the accelerometer enough times, the distribution would approach a Gaussian distribution, averaging the effect we see in Figure 13 diagram B to a small value. After applying a velocity threshold, the issue on accuracy would be minimal.

2.8 Schematics

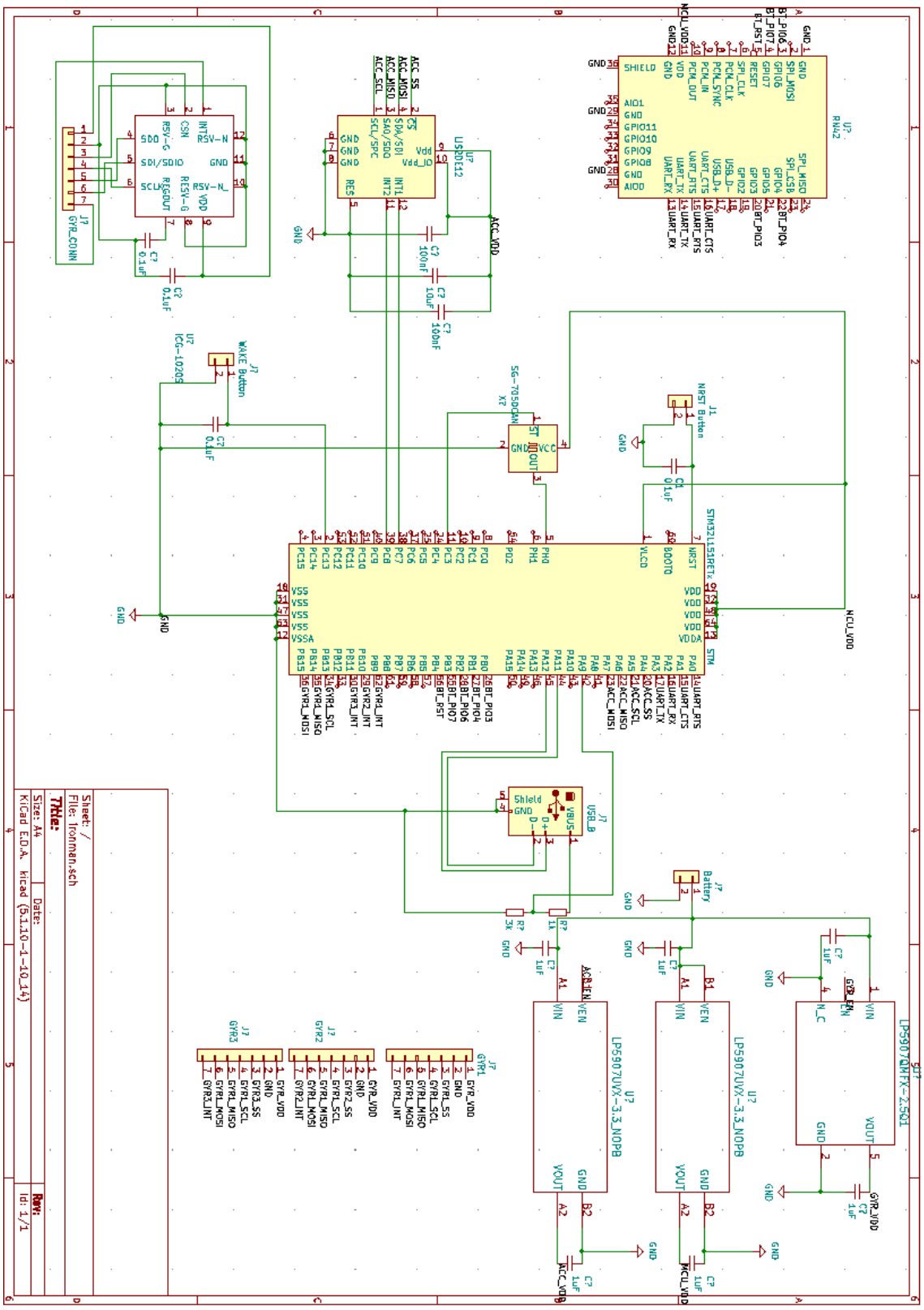

Figure 15. Complete Schematic

3. Cost and Schedule

3.1 Cost Analysis

3.1.1 Labor

$$Cost_{labor} = \$/hour * people * hour/week * weeks$$
$$= \$25/hr * 3\ people * 10\ hr/week * 10\ weeks$$
$$= \$7500$$

3.1.2 Parts

Part	Cost
Microcontroller (STM32L152RET6)	$\$9.55$
Accelerometer (LIS2DE12)	$\$1.55$
Gyroscope (ICG-1020S)	$\$3.99 * 3 = \$11.97$
Bluetooth Module (RN42HID-I/RM)	$\$18.95$
Voltage Regulator LP5907QMFX-2.5Q1	$\$0.70$
Voltage Regulator LP5907UVE-3.3/NOPB	$\$0.68 * 2 = \$1.36$
Battery 3.7V 400mah PRT-13851	$\$4.95$
Total	$\$49.03$

3.1.3 Grand Total Cost:

$$\$7549.03$$

3.2 Schedule

Week	Due	Jeff	Zhiyuan	Yayati
1 (Sep 20)	None	1. Design Doc: (2.a, 2.b, 2.c.ii, 3) 2. Study pin layout (MCU, sensors) 3. PCB design	1. Design Doc: (2.c.i, 2.c.iii, 2.d, 4) 2. Study pin layout (MCU, sensors) 3. PCB design	1. Design Doc: (1, 2.c.iv) 2. Study pin layout (MCU, bluetooth module) 3. PCB design
2 (Sep 27)	- PCB Review - Design Doc - Talk to Machine Shop	1. Develop sensor preprocess algorithm	1. Refine PCB design 2. Order Parts	1. Refine PCB design 2. Get used to STM32Cube Dev-Env
3 (Oct 4)	- PCB order #1 - Sensor data processing	1. Translate Algorithm into C 2. Bluetooth Interface	1. Implement primitive driver code 2. Bluetooth Interface	1. Implement primitive driver code 2. Power System
4 (Oct 11)	- Team Evaluation - BT interface - Minimal assembly	1. Prepare for evaluation 2. Work on driver/BT connection	1. Prepare for evaluation 2. Work on driver/BT connection	1. Prepare for evaluation 2. Power System
5 (Oct 18)	- Driver - Basic Mouse - Self Verification 1	1. PCB redesign 2. Basic mouse functionality	1. PCB redesign 2. Basic mouse functionality	1. PCB redesign 2. Basic mouse functionality
6 (Oct 25)	- PCB order #2	1. Advanced Functions 2. Individual progress	1. Advanced Functions 2. Refine subsystems	1. Advanced Functions 2. Individual progress

			3. Individual progress	
7 (Nov 1)	- Individual progress	1. Refine Subsystems		
8 (Nov 8)	- Assembly - Self Verification 2	1. Refine System 2. Vericate design based on proposal		
9 (Nov 15)	- mock demo	1. Prepare for Mock Demo 2. Final Assembly		
10 (Nov 22)	- Ready to present	1. Presentation preparation 2. Final Paper		

4. Ethics and Safety

Since we are developing a peripheral device, there are relatively less safety and ethical hazards. However, there is still a possibility of having some issues. As our project is a wearable peripheral device, any circuit failure could harm the consumer. Therefore, we will make sure to isolate the circuit in an insulated casing to reduce the potential harm to consumers and use the inbuilt temperature sensors on the MCU as well as the accelerometer to detect overheating situations and power off the device accordingly to uphold IEEE Code of Ethics #1: “To hold paramount the safety, health and welfare…” [1].

Using a lithium Ion battery brings risk that we must consider. Overcharging a battery can generate heat within a cell that could trigger thermal runaway that could lead to explosion [6]. Therefore, when we are charging our battery, we should not exceed the charge limited voltage and charge with the recommended current level.

The United States Consumer Product Safety Commission recommends that “Battery-powered products be designed with a system approach addressing thermal protection, charge and discharge protection” [7]. The battery module we have chosen has overcharge protection, short-circuit protection, heat protection, and collision protection, which fulfills this recommendation.

Since we will also be working in an electronics lab, and carrying out complex procedures such as assembling boards using soldering and powering the device using lab supplies, we have attended lab safety training to learn how to use the required equipment safely with appropriate gear such as eye protection glasses, gloves and lab coats to avoid risk of burns, shorts and shock in adherence to IEEE code #6: “To maintain and improve our technical competence…” [1].

Another large portion of our project encompasses the software programming portion. We will strictly ensure that any content used from outside sources are properly cited and credited to avoid plagiarism while keeping in adherence to IEEE code #5: “... to credit properly the contributions of others” [1].

We will also be adhering to the remaining code of conduct as mentioned in the IEEE Code of Ethics [1] so as to ensure the highest ethical and professional conduct throughout the development of our project.

5. Citation

[1] Ieee.org, “IEEE Code of Ethics”, 2020. Available at: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed 30 September 2021].

[2] K. Seifert and O. Camacho, “Implementing positioning algorithms using accelerometers - NXP,” *Implementing Positioning Algorithms Using Accelerometers*, Feb-2007. [Online]. Available at : https://www.nxp.com/docs/en/application-note/AN3397.pdf. [Accessed 30 September 2021].

[3] Majesticglove.com, 2021. *Product Personalization & Custom Logo Services - Majestic Glove*. [online] Available at: https://www.majesticglove.com/custom-logo-services/ [Accessed 30 September 2021].

[4] Roving Networks. “Bluetooth Data Module Command Reference & Advanced Information User’s Guide”, 2013. Available at: https://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/bluetooth_cr_UG-v1.0r.pdf. [Accessed 30 September 2021].

[5] J. N. Ingram, K. P. Körding, I. S. Howard, and D. M. Wolpert, “The Statistics of Natural Hand Movements,” Experimental Brain Research, vol. 188, no. 2, pp. 223–236, 2008. [Accessed 30 September 2021].

[6] D. H. Doughty and E. P. Roth, “A general discussion of Li Ion Battery Safety,” *Electrochemical Society Interface*, 2012. [Accessed 30 September 2021].

[7] “Batteries,” *U.S. Consumer Product Safety Commission*. [Online]. Available at: https://www.cpsc.gov/Regulations-Laws--Standards/Voluntary-Standards/Topics/Batteries. [Accessed 30 September 2021].