Marching Band Assistant

Design Review

Team Number 6

Wynter Chen (wynterc2)
Alyssa Licudine (alyssal3)
Prashant Shankar (shankar7)

ECE 445
TA: Dhruv Mathur
October 1, 2020

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Objective

One of the primary responsibilities of a drum major is to conduct a consistent tempo during marching band practices and live performances. However, most high school drum majors are tasked with this responsibility with no prior training. While some high schools and colleges have access to drum major camps to train and practice fundamentals, drum majors often do not have practice tools readily available to receive feedback on their conducting.

The Marching Band Assistant (MBA) aims to create a method for drum majors to practice and analyze the consistency and tempo of their conducting. An arm attachment with an inertial measurement unit on it would be used to record acceleration from the conducting arm. The derived acceleration data processed via a microcontroller would be able to calculate the user's average tempo after a short period of recording, as well as the standard deviation of time between conducting motions. The calculations would be transmitted via Bluetooth to a GUI on a laptop, where the user could both have data displayed in real time and view data afterwards to observe any inconsistencies in his or her conducting.

1.2 Background

A drum major is the leader of a marching band, a common entertainment organization that plays musical numbers at the sporting events of most high schools and colleges. A drum major's responsibilities include relaying vocal commands, communicating with band members, and conducting effectively [1]. Conducting establishes the tempo of a musical number, and the drum major is the only visual source of tempo during performances.

Thus, metronome devices such as the Dr. Beat are used in personal or full-band practices to commit tempos to memory [2]. However, there are issues with using an audio source due to the spaced-out nature of marching bands combined with the relatively slow speed of sound. Additionally, there is currently no way to tell if one's conducting motions are on time without the use of a metronome. Software such as SmartMusic serve a similar purpose to our solution; it receives input via microphone and provides correctional feedback to musicians, but this only extends to singing or playing an instrument [3].

To our knowledge, there is no commercial device that records the movements of one's arms to determine the tempo or consistency of conducting. Personal interviews with Metea Valley High School Band Director Glen Schneider, Marching Illini member Daniel Dresser, and University of South Carolina Drum Major Kelley Powell indicate that the tool would be a valuable asset in high school marching bands and practice environments.

Having a tool where drum majors can record their motions would help them verify if they truly committed a tempo to memory, and if they can keep a consistent tempo over a long period of time. Our tool intends to graphically display conducting data both during and after the recording session ends, so that drum majors can analyze their behavioral patterns and catch inconsistencies. Our expectation is to deliver real-time data with a delay of less than 500 ms. We believe this is an appropriate limit because while the user does not need instant feedback since the long-term behavior of conducting is more useful information than a single beat, the user should be able to identify within two measures of music if his or her tempo or consistency has changed.

1.3 High-Level Requirements

- The MBA must be able to identify sudden stops of the arm by polling acceleration data at a rate above 1 kHz.
- The MBA should be able to display the latest tempo data calculations with a delay of less than 500 ms.
- The MBA should be able to be used wirelessly without charging for at least four hours.

2. DESIGN

2.1 Block Diagram

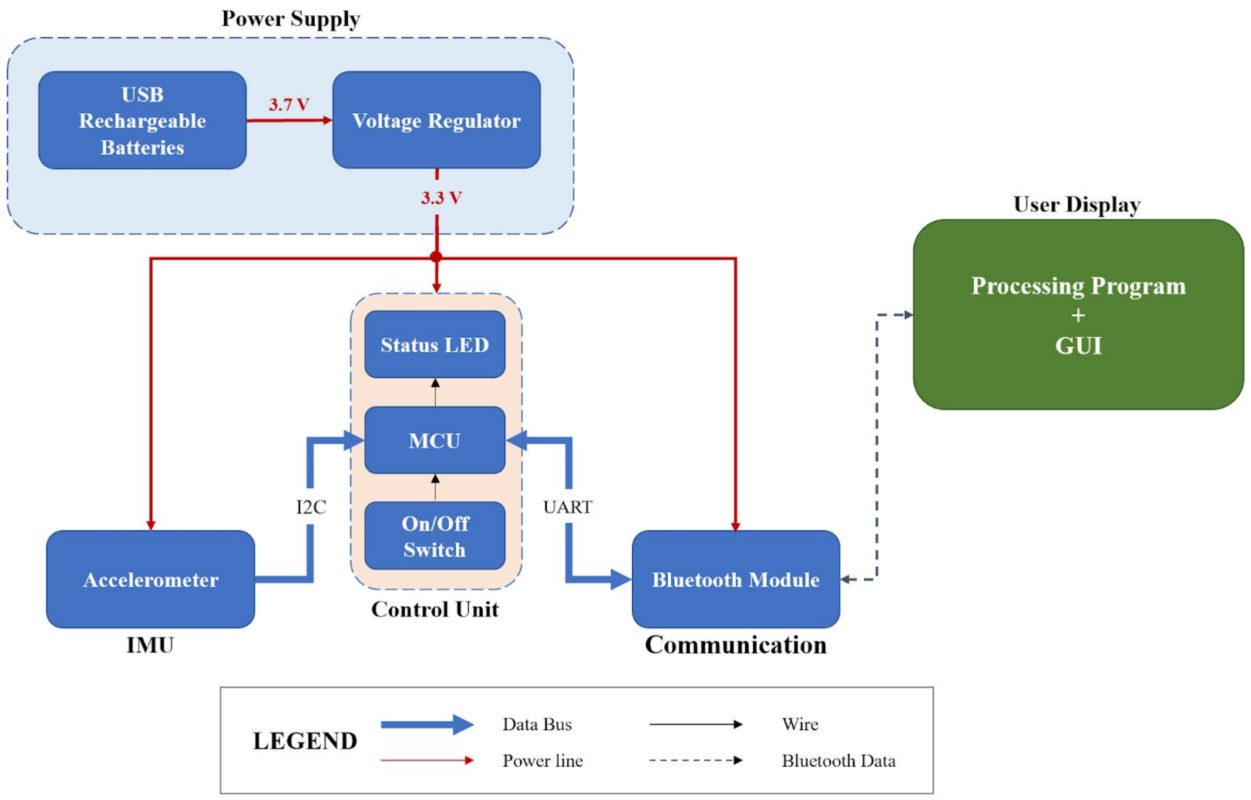


Fig. 1. A comprehensive block diagram of the Marching Band Assistant.

The MBA consists of five main subsystems: a power supply, the inertial measurement unit (IMU), the control unit (CU), communication, and the user display. The power supply drives the steady operation of the sleeve, powering all its components for up to four hours, which is the high-end typical length of a band rehearsal. The IMU contains the accelerometer, which captures the acceleration of the user's arm motions. This data is then passed onto the MCU in the control unit subsystem. The MCU parses and processes the data received from the IMU and sends it to the Bluetooth module in the communication subsystem. The processed data is then transmitted to the computer via Bluetooth. A Python program collects and displays the data on the user's computer.

2.2 Physical Design

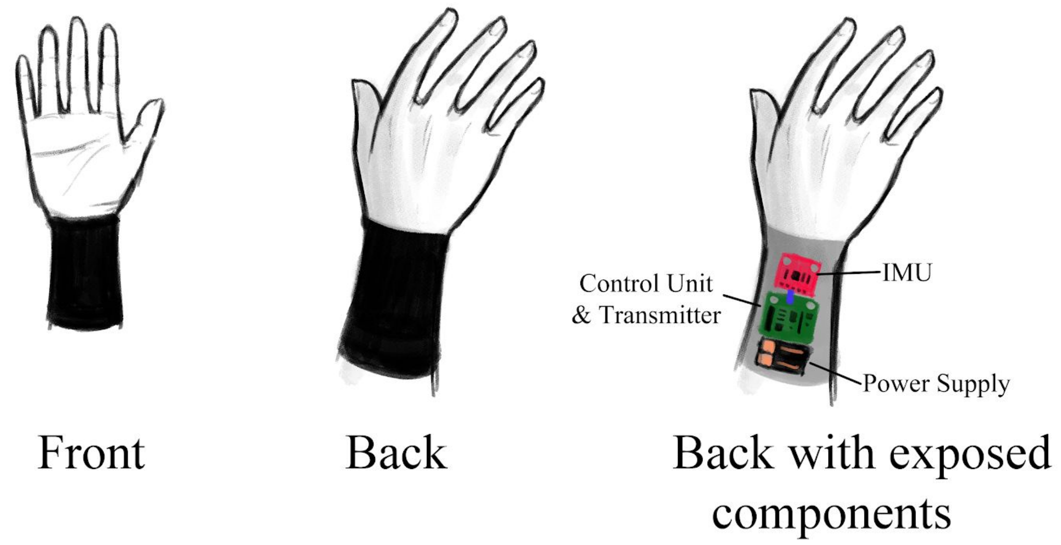


Fig. 2. Artist's rendition of the sleeve component.

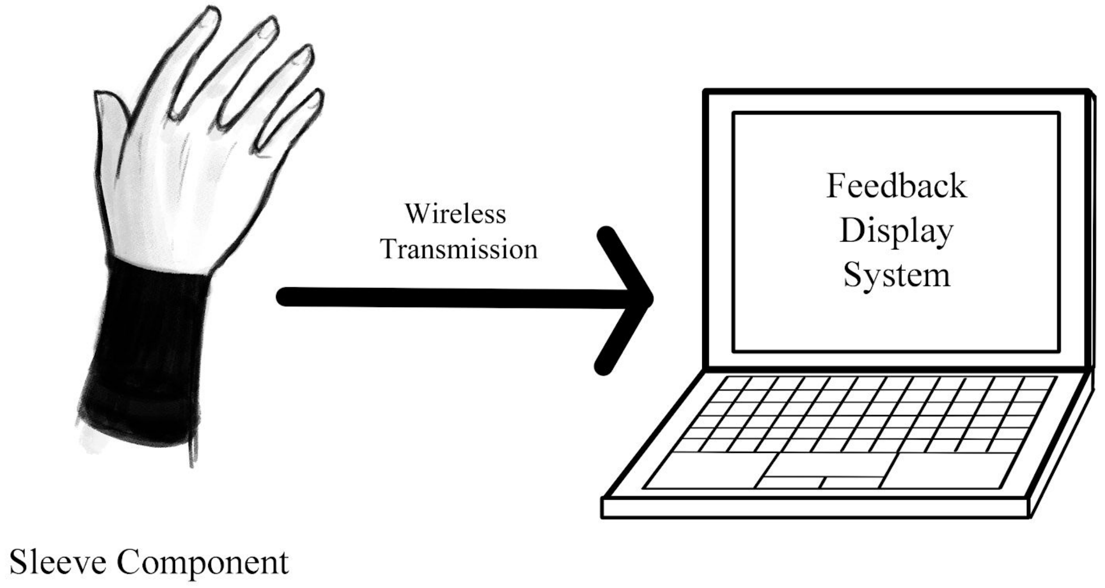


Fig. 3. Usage of the sleeve component with the feedback display system.

Our design allows the user to slip the one-size-fits-most device on his or her arm, even if a marching band uniform is worn. The MBA sleeve will be made out of nylon that wraps around the arm, so that the IMU can be placed on the wrist. Velcro will be used to secure the sleeve to the user's arm. The positioning of the IMU closer to the hand is crucial for recording accurate data, while the control unit and power source can be placed further down the wrist to avoid putting extra weight on the upper arm. The components will be covered by another layer of nylon fabric so that the MBA sleeve can be transported safely and used outdoors. The MBA sleeve contains a Bluetooth module in the sleeve, which sends data to the computer feedback system. The feedback system on the computer will be an executable program, with options to view data in real-time or observe past data sets.

2.3 Block Descriptions | Requirements and Verifications

Subsystem 1: Inertial Measurement Unit

Before explaining the specifications of our inertial measurement unit, we will explain why we are collecting acceleration data in the first place. A fundamental requirement of the MBA is to determine when the user has conducted a beat. A "beat" is conducted when the user stops their arm in a certain place in between motions in order to establish a certain tempo. Regardless of where the arm stops, the common trend is that a beat is observed when there is a sudden decrease in velocity of the arm.

Acceleration data can help determine these sudden shifts in velocity. When an object in motion comes to a sudden stop, the magnitude of acceleration drastically increases, then drastically decreases shortly afterwards since the object is at rest for a non-zero period of time. Therefore, we will need to detect anomalies in acceleration data to determine when a beat is conducted. Our tolerance analysis in Section 2.4 goes into more detail with regards to quantitative results.

The inertial measurement unit (IMU) will extract acceleration information from the user's arm motions. Acceleration along one axis causes displacement on the corresponding proof mass, and the capacitive sensors in the IMU detect the differential displacement. Consumer IMUs such as the ICM-20948 use proof masses for each of its three angular rate axes [4]. The ICM-20948's three sensors are an accelerometer, a gyroscope, and a magnetometer, but we will only use the accelerometer functionality of the ICM-20948 for now. We have no use for measuring magnetic forces, and we currently do not have plans to use the gyroscope due to rotational motions on an arm during conducting being sparse even with slow motions. However, we will consider using gyroscope functionality if we do not get sufficient information from accelerometer data alone, as the ICM-20948 is capable of using both the accelerometer and gyroscope at the same time. Our tolerance analysis and other subsystems will take into account the highest possible amount of current the IMU will draw, which is 3.11 mA when all three sensors are used in low-noise mode.

Each sensor has a sigma-delta analog-to-digital converter (ADC) that produces digital outputs, with an adjustable sensitivity of $\pm 2g$, $\pm 4g$, $\pm 8g$, or $\pm 16g$ for the accelerometer and ±250, ±500, ±1000, and ±2000 °/sec for the gyroscope. The data outputted from the IMU specifies the acceleration in each axis, while the gyroscope output specifies the rotational motion in each axis [4].

A Digital Motion Processor (DMP) computes the data it acquires from the IMU using motion processing algorithms. The data is sent in digital form, utilizing I2C protocol after sampling, digitizing, and packeting the data. The packeted data is 16 bits long. Power is supplied to this module via the analog VDD pin, and its data is transferred to the MCU through its data line pins.

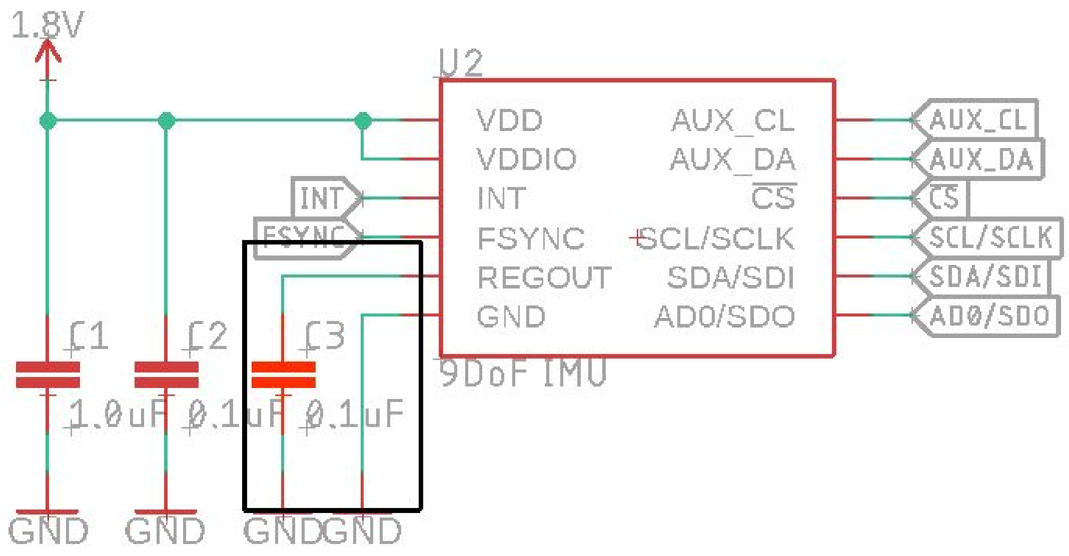


Fig. 4. Typical Operating Circuit Schematic of the ICM-20948.

Instead of using the ICM-20948 on its own, we have opted to use the SparkFun 9DoF IMU Breakout, which contains the specified IMU, level shifters to regulate and supply voltage to the chip, and two Qwicc connectors to more easily facilitate I2C communication between the IMU and ATMega328P MCU [5].

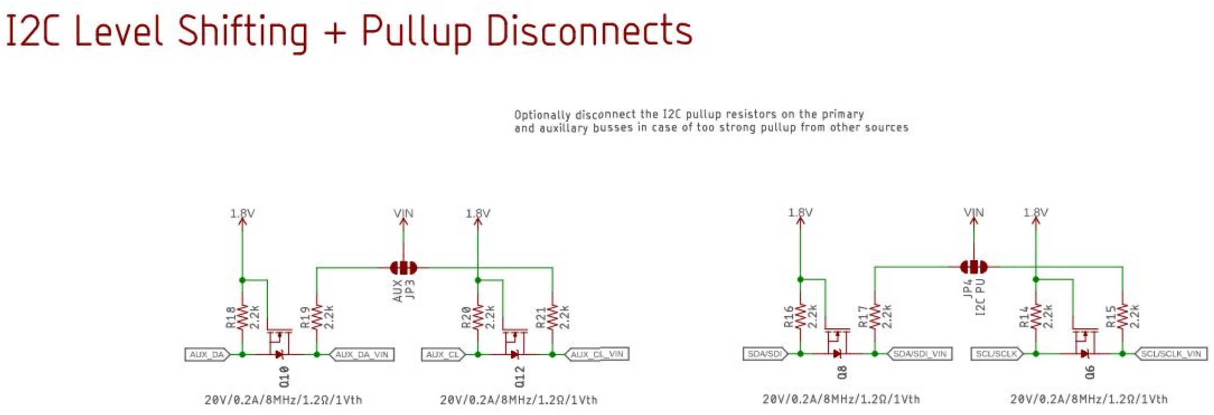


Fig. 5. I2C Level Shifting and Pullup Disconnects Schematic for the IMU Breakout Board.

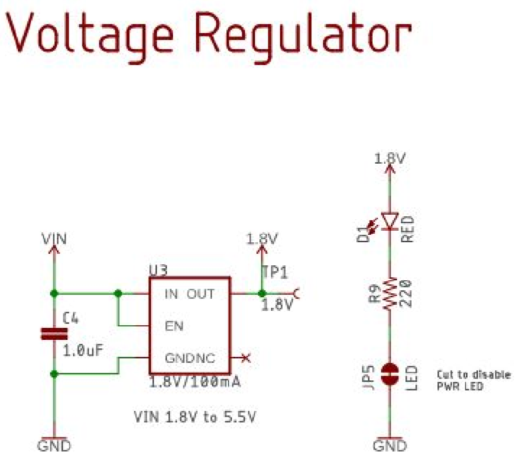


Fig. 6.Voltage Regulator Schematic for the IMU Breakout Board.

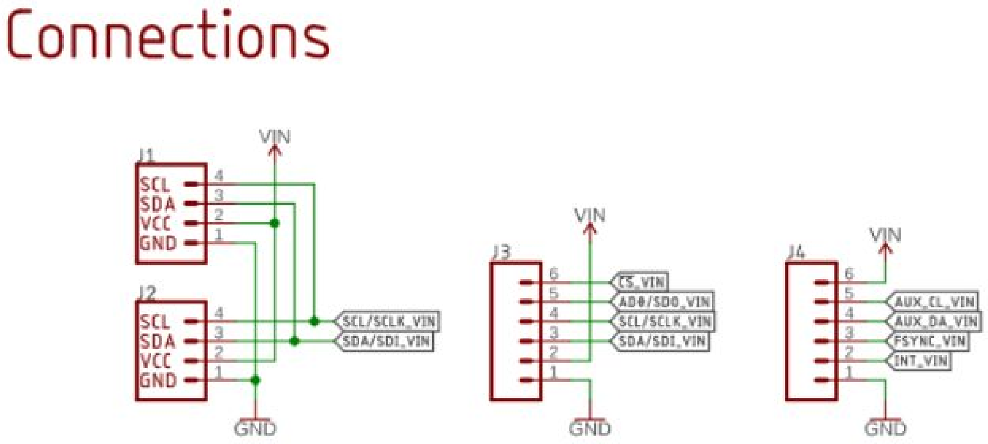


Fig. 7. Qwicc Connectors Schematic for the IMU Breakout Board.

We have analyzed videos of ourselves conducting at 180 beats per minute, which is typically the fastest a drum major is asked to conduct [6]. We found that our arm is stopped in place for around 0.2 seconds, and for as little as 0.1 seconds. We would like our IMU to run at a frequency fast enough which would enable us to guarantee that the IMU can detect a short, steep spike in acceleration. In low-noise mode, the maximum frequency the accelerometer samples at is 1.125 kHz, or 1,125 samples a second. Even with our fastest observed conducting stop of 0.1 seconds, we would have a minimum of 112.5 samples for each stop of the arm, which is more than enough data points to determine if a significant spike in acceleration happened. It is worth noting that the most significant parts of the acceleration peaks will usually be less than 112.5 samples, but our tolerance analysis both acknowledges this fact and verifies that there are more than enough data points to classify acceleration data spikes.

Since the maximum output data rate is 1.125 kHz and each data entry contains 16 bits, the maximum bit rate of the IMU is 18 kbps, while the maximum baud rate of the IMU is 1125 kBd. Given that the ATMega328P MCU can run with a baud rate of at least 1 MBd without issue, we will have no problems running the IMU at the maximum output data rate [7].

Table 1: R&V table for the inertial measurement unit subsystem.

Requirement	**Verification**
Inertial Measurement Unit 1) IMU must be able to send acceleration data to the MCU via I2C. 2) IMU must be able to sample acceleration at a rate of at least 1 kHz.	Inertial Measurement Unit 1) Check that the MCU receives any packet of data from the IMU. *Verification Process:* 1) Load program that pings the ICM-20948 IMU via I2C onto the ATMega328P MCU. 2) Connect the MCU to the IMU. 3) Power on both devices with any appropriate power supply and run the microcontroller script. If any data packet from the IMU is returned through the MCU, the test was successful. 2) Check that we are collecting at least 1,000 samples a second on average from the IMU. *Verification Process:* 1) Load program that records a timestamp of when data is received onto the ATMega328P MCU.

	2) Connect the MCU to the ICM-20948 IMU. 3) Power on both devices and record data for any amount of time. 4) View the data collected by the MCU on a computer. If the number of samples collected is greater than 1,000 multiplied by the number of seconds that data was recorded, the test was successful.
3) IMU must be able to record correct acceleration data in the x and y axes.	3) The IMU must read 0 m*g* ± 25 m*g* when resting, and must be able to read at least 1000 m*g* when moving for the x and y axes. *Verification Process:* 1) Load program that collects acceleration data at a rate of at least 1kHz onto the ATMega328P MCU. 2) Connect the ICM-20948 IMU to the MCU. 3) Lay the IMU on a flat, stationary surface. 4) Collect data over three seconds. 5) Check that the x and y acceleration data does not ever exceed ±25 m*g* while stationary. 6) Hold IMU in the palm of hand. 7) Move hand back and forth to a metronome set at 160 BPM at least 2 feet along the IMU's x-axis. Continue the process for at least 10 seconds. 8) Check that the x-acceleration exceeds |800 m*g*| at least once. 9) Repeat steps 7 and 8 along the y-axis.
4) IMU must be able to record correct acceleration data in the z axis.	4) The IMU must read 1000 m*g* ± 10% when at rest, and must be able to read at least 1800 m*g* for the z-axis. *Verification Process:* 1) Load program that collects acceleration data at a rate of at least 1kHz onto the ATMega328P MCU. 2) Connect the ICM-20948 IMU to the MCU.

	3) Lay the IMU on a flat, stationary surface. 4) Collect data over 3 seconds. 5) Check that the z-axis acceleration data reads 1000 m*g* ± 10% while stationary. 6) Hold IMU in palm of hand. 7) Move hand back and forth to a metronome set at 160 BPM at least 2 feet along the IMU's z-axis. Continue process for at least 10 seconds. 8) Check that the z acceleration exceeds 1800 m*g* at least once.

Subsystem 2: Control Unit

The control unit is driven by a microcontroller and interacts with the user through its input, an on/off switch, and its output, an LED light that indicates whether the sleeve is off or on. The microcontroller will execute all the interpretation and processing of the raw data from the data registers of the IMU. Its flash storage contains the data that will be forwarded to the Bluetooth module via a Universal Asynchronous Receiver/Transmitter (UART). The computer will then receive the final values for the beats per minute to be outputted to the GUI via Bluetooth.

Microcontroller

The microcontroller will be calibrated to receive and parse raw data from the IMU's digital motion processor (DMP) in real-time. It will also be programmed with calculations that will filter out noise, perform various moving averages of the acceleration data, and normalize the data according to the sensitivity scale factor.

The ATMega328P was chosen because its range of baud rates accommodate for the data rate our application needs, which is at least 1.125 kbps. The rates of the ATMega328p range from 2400 bps to 230.4 kbps. It also has considerable documentation and application to electronics projects, which will prove to be useful for reference when designing and constructing the overall circuit for the sleeve.

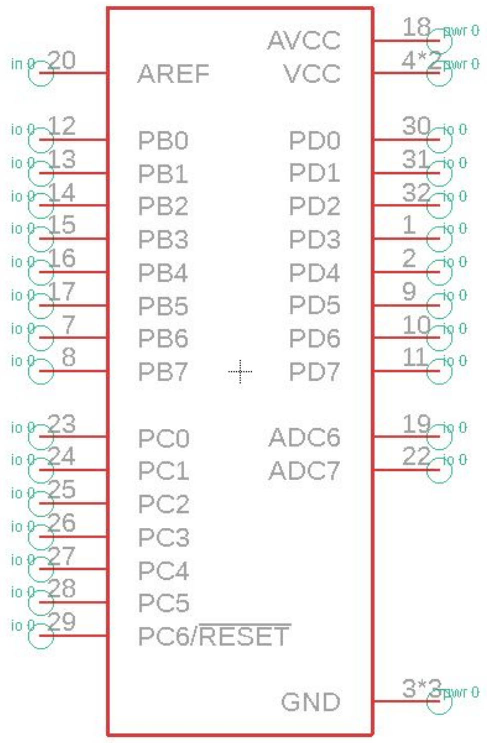


Fig. 8. ATMega328P Schematic.

On/Off Switch

The on/off switch is the user-control that is used to activate and deactivate the circuit. When the user switches the MBA sleeve attachment on, the microcontroller is activated and begins collecting data from the IMU. When the user switches the sleeve attachment off, the voltage regulator stops supplying power to the components in the circuit, meaning the MCU and IMU turn off as well.

Status LED

The LED will turn on or off according to the operation mode of the microcontroller. When the switch is pushed into the “ON” mode with a fully-charged battery, the light will turn on. When the switch is pushed into the “OFF” mode, the LED will not light up because there will be no current.

Table 2: R&V table for the control unit subsystem.

Requirement	**Verification**
<u>Microcontroller</u> 1) Microcontroller must communicate with the Bluetooth module over UART at a speed of 1.125 kBd.	<u>Microcontroller</u> 1) The microcontroller's performance will be evaluated with a USB UART bridge and a computer terminal. *Verification Process* 1) Connect the microcontroller to a USB UART bridge and a terminal such as PuTTY. 2) Set up the terminal at 1.125kBd. 3) Send and echo back 100 characters. 4) Confirm that all characters on the received on the terminal match the characters that were sent.
<u>On/Off Switch</u> 1) Switch must have an operating force above 1000 grams [8, 9].	<u>On/Off Switch</u> 1) Check that the operating force of the switch is above 1000 grams. *Verification Process* 5) Connect switch in series with a resistor and battery. 6) Attach DMM probes parallel to the resistor. 7) Ensure the switch is initially in the "OFF" position by checking that the voltage of the resistor is 0V. 8) Stack weights on top of the switch, until the switch is turned on and a non-zero voltage is measured across the resistor. 9) Check the size of the weights to ensure that it is over 1000 grams.
<u>Status LED</u> 1) LED must be able to turn on.	<u>Status LED</u> 1) Ensure that the LED is able to turn on. *Verification Process:* 1) Attach DMM probes to respective ??? 2) Turn switch "OFF". 3) Measure current going through the LED while the switch is off. 4) Turn switch "ON". 5) Measure current going through LED while switch is on.

	6) Verify that the currents measured are in the correct operating regions on the LED.
2) LED must be visible from two feet away.	2) The lux of the LED will be measured while in a room with under 15 lux. The measured lux of the LED must exceed 30. *Verification Process:* 1) Connect the two ends of a 1% tolerance photoresistor to the DMM probes. 2) Turn off lights in verification lab 3) Measure lux of the lab when lights are off, by placing the photoresistor two feet away from the LED and measuring the resistance. 4) Turn switch "ON" to turn on the LED 5) Measure lux of the lab while LED is on and lights are off, by placing the photoresistor two feet away from the LED and measuring the resistance. 6) Subtract the lux of the lab when lights are off from the lux of the lab when LED is on and lights are off to obtain the lux of the LED. 7) Check that the lux measured is above 30.

Subsystem 3: Communication

The communication subsystem includes one Bluetooth module inside the MBA sleeve attachment. The Bluetooth module allows for wireless data collection and transmission to the computer, eliminating the need for wires that may limit the user's movement. The module on the sleeve will communicate with the microcontroller via UART. The "Record Data" user display button being clicked will trigger the Python program to start collecting data from the Bluetooth module's TXD pin via Bluetooth. When the user clicks "Stop Recording Data", the program will communicate a signal that prompts the MCU to stop collecting data from the IMU via Bluetooth; the Bluetooth module receives this signal through the RXD pin [10].

Range and data transmission rate were factors that were considered in deciding between the two widely known wireless communication methods: Wi-Fi and Bluetooth. The distance between the sleeve and the user display is dependent on the user's visual acuity and the computer's placement. Because the user will likely prefer to have the display nearby so it is easy to view and pause the program if needed, this distance will not exceed five meters. The bits per second needed for continuous data transmission from a consumer MCU such as the ATMega328P is no more than 1 Mbps [7]. Compared to Wi-Fi, Bluetooth is more energy and space-efficient for our application because it is used for transferring small bandwidths of data at speeds of about 3 Mbps in a 10-20 meter range with low-power consumption.

The HC-05 Bluetooth Transceiver Module was chosen because its default baud rate (9.6 kBd) is greater than that of the ICM-20948's maximum output data rate, 1.125 kBd [10]. Compared to other Bluetooth modules, it has an expansive collection of documentation on interfacing with Python and its integration with the ATMega328P. These resources will prove useful should any bugs occur during construction and testing of the system.

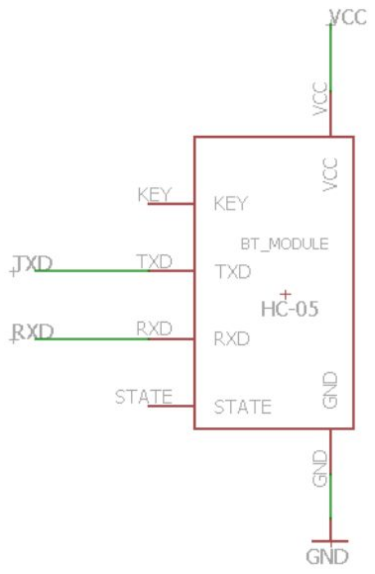


Fig. 9. HC-05 Bluetooth Transceiver Schematic.

Table 3: HC-05 Bluetooth Transceiver Pin Connections.

Pin Name	**Description / Connection**
VCC	Powers the module. Connect to +3.3V supply voltage.
KEY	Toggles between Data Mode (set low, default) and AT command mode (set high). In data mode, the module can send and receive data from other Bluetooth devices. To enable data mode, this pin will not be connected to anything on the PCB.
TXD	Transmits serial data. Everything received via

	Bluetooth will be sent through this pin as serial data. This pin will connect to the microcontroller's RX pin.
RXD	Receives serial data. Every serial data given to this pin will be broadcasted via Bluetooth. This will be connected to the microcontroller's TX pin.
STATE	The state pin is connected to an LED. It can be used as feedback to check if Bluetooth is working properly. We will not be using this pin, as the Bluetooth module already has its own status LED.
GND	Ground pin of module. Connect to PCB system ground.

Table 4: R&V table for the communication subsystem.

Requirement	**Verification**
1) The Bluetooth module must have a baud rate greater than 1.125 kBd, the output data rate of the MCU.	1) The Bluetooth module's baud rate will be obtained from and displayed by the Arduino IDE. *Verification Process:* 1) Insert Bluetooth module header pins into breadboard. Make the following connections from the module to an Arduino Uno with jumper wires (connect wire in same row as respective pin on the breadboard): HC-05 GND ---> Arduino GND Pin HC-05 VCC (5V) ---> Arduino 5V HC-05 TX ---> Arduino Pin 10 (soft RX) HC-05 RX ---> Arduino Pin11 (soft TX) HC-05 Key (PIN 34) ---> Arduino Pin 9 2) Load and compile the Arduino program HC-05.ino (Appendix A) in the IDE by clicking "Verify" [11]. 3) Before connecting the Arduino to the USB, remove the VCC wire from the HC-05 so it is not getting any power from the Arduino. 4) Connect the Arduino Uno to the USB

	cable extended from a PC. Upload the HC-05.ino program to the board by clicking “Upload” in the IDE. 5) Reconnect the Arduino Uno 5V wire to the HC-05’s VCC pin. The HC-05 LED will blink on and off at about 2 second intervals, indicating that the HC-05 is in AT command mode and ready to accept commands. 6) Open the Serial Monitor from the Arduino IDE, type "AT", and click SEND. “OK” should appear on the terminal to confirm the HC-05 is properly connected to the PC via Bluetooth. 7) Type “AT+UART” in the Serial Monitor to see the baud rate the module is operating at. 8) Confirm the baud rate is greater than 1125 Bd. If not, type “"AT+UART=9600,1,0" to explicitly set the baud rate (9600 is the default). Upload the program to the Arduino again and repeat steps 6 and 7. 9) Confirm the baud rate is greater than 1125 Bd.

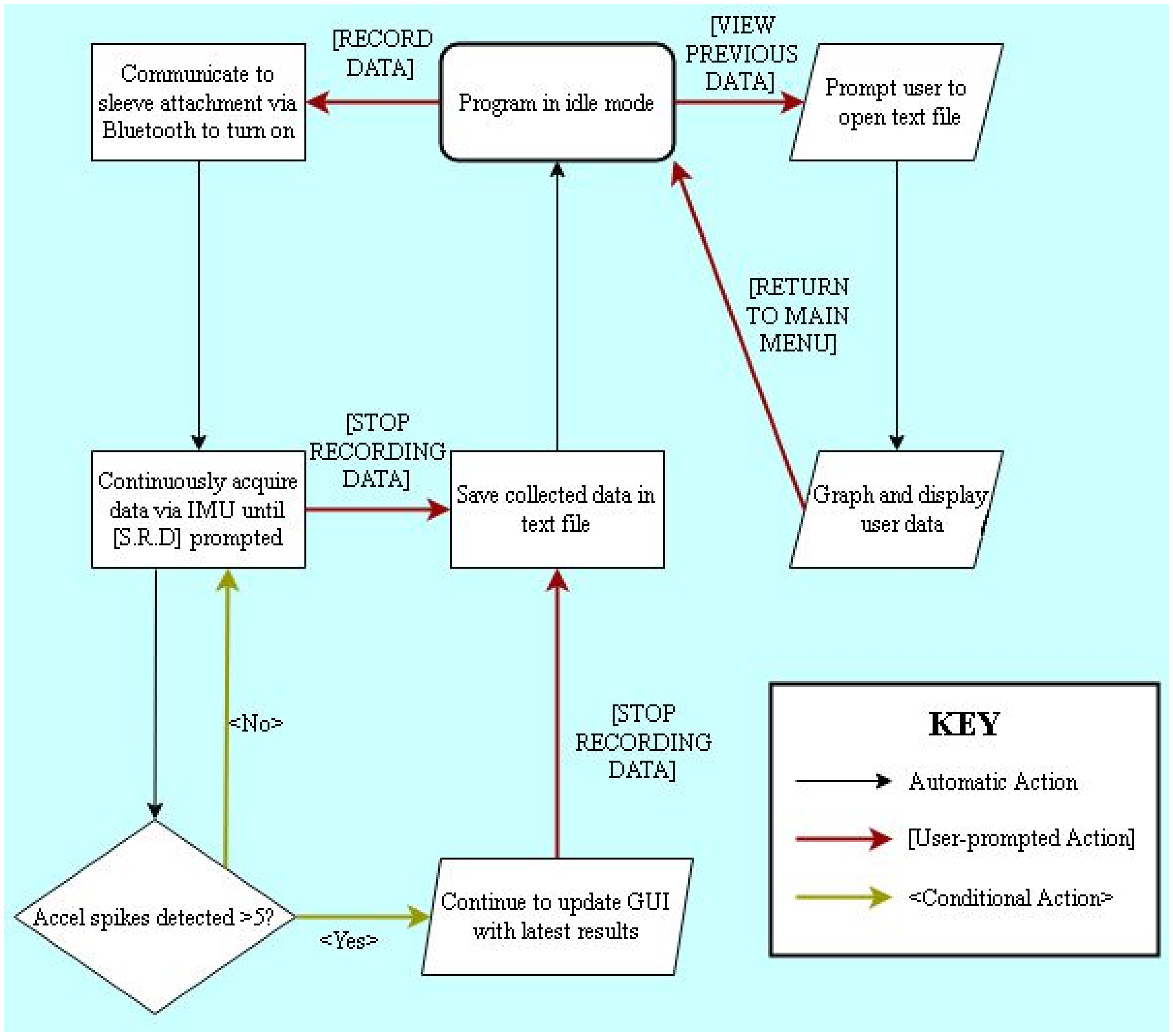
Subsystem 4: User Display

The user’s computer will display the MBA’s GUI. The GUI will be a simple executable run on Python. The frontend will contain an option to start or stop recording data, which will display the latest MCU calculations in real time after enough data is collected (e.g. five spikes in acceleration detected by the IMU). There will also be an option to observe graphical data after the recording ends, and if time permits, an option to practice conducting to a sample piece of music. The backend will consist of a Python script that receives the calculations and displays it live on a GUI. The GUI will be run using the Tkinter library, but similar libraries such as Kivy and PyQT are possible alternatives [12]. To receive data from the Bluetooth module, the PyBluez module can be used [13].

The GUI would look relatively simple, with a button to either start or stop recording data, as well as a button that enables the user to look at previously recorded data. If the user selects the option to record data, the program will command the IMU to start collecting data via Bluetooth. After the IMU records five spikes in acceleration, the program will continuously display the average tempo over the past five beats and the standard deviation of time between the past five beats, as well as the averages for tempo and standard deviation since the recording session started. All four pieces of data would be updated as soon as the IMU records a new spike in acceleration, meaning the data is updated, calculated and displayed in real time.

When a recording session ends, the program tells the MCU via Bluetooth to stop collecting data from the IMU. The program will then store necessary data in a text file that will be read using the Pyplot module in Python. The user will be prompted to pull up the appropriate data file, and graphs will be available with respect to time for both rolling and cumulative averages for both tempo and consistency. Using alternative data storage methods such as CSV and Microsoft Excel spreadsheet are plausible and easily implementable, but they will only be explored if our text file method is insufficient.

Fig. 10 demonstrates the flowchart of our planned software program. If the user chooses the option from the main menu to record data, the program prompts the sleeve attachment via Bluetooth to turn its modules on, and the MBA will continue to display the latest data until it is prompted to stop by the user. Additionally, the user can also choose to pull up past data from a text file, and return to the main menu when necessary.



[RECORD DATA]
Communicate to sleeve attachment via Bluetooth to turn on
Program in idle mode
[VIEW PREVIOUS DATA]
Prompt user to open text file
[RETURN TO MAIN MENU]
[STOP RECORDING DATA]
Continuously acquire data via IMU until [S.R.D] prompted
Save collected data in text file
Graph and display user data
<No>
[STOP RECORDING DATA]
Accel spikes detected >5?
<Yes>
Continue to update GUI with latest results
KEY
Automatic Action
[User-prompted Action]
<Conditional Action>

Fig. 10. A flowchart demonstrating our user display.

Table 5: R&V table for the user display subsystem.

Requirement	**Verification**
<u>User Display</u> 1) The user display contains a toggleable option to connect to the MCU via Bluetooth.	<u>User Display</u> 1) Check if the backend of the user display recognizes connection to device via Bluetooth. *Verification Process:* 1) Create a Python script that attempts to connect to a specified Bluetooth module and returns a print message (e.g. "Success") when connection is successful. 2) Load script onto computer that supports Bluetooth capability. 3) Power on HC-05 Bluetooth module within 5 m of the host computer. 4) Run the Python script. If the specified print message is outputted, the test was successful.
2) Data recorded by the IMU is visible on the user display in any numerical form.	2) Check if raw MCU data is displayable by GUI. *Verification Process:* 1) Create a Python script that initializes collecting IMU data, prints the raw output data of the ATMega328P MCU onto a GUI as soon as data is received. 2) Connect MCU to ICM-20948 IMU, and connect MCU to host computer via Bluetooth. 3) Run the Python script for at least 10 seconds. If the raw data received from the MCU is displayed on the GUI in numerical form, the test was successful.
3) User Display screen updates with new data at least two times a second.	3) Check using a counter if GUI is updating at an acceptable rate. *Verification Process:* 1) Repeat Step 1 of Requirement 2 of the User Display, and add a counter that keeps track of how many times the GUI has updated each second (counter does not have to be visible on GUI). 2) Connect MCU to ICM-20948 IMU,

	and connect MCU to host computer via Bluetooth. 3) Run the Python script for at least 10 seconds. If the counter is at least equal to 2 for every second the script was run, the test was successful.

Subsystem 5: Power Supply Unit

Input: +3.7V Lithium Ion Polymer (Li-Poly) Rechargeable Battery

Outputs: +3.3V ± 5% for the ICM-20948 9DoF IMU Breakout Board
+3.3V ± 5% for HC-05 Wireless Bluetooth Module
+3.3V ± 5% for ATMega328P Microcontroller

The power supply will provide the power necessary for steady operation of the IMU, the control unit, and the communication subsystems on the sleeve. With the convenience of the user in mind, we will be using a lightweight Li-Poly battery that can be recharged via mini-USB [14]. Primary lithium batteries are lighter than other primary chemistries and are suitable for low current applications [15]. On the sleeve, the battery and voltage regulators should be compact and be able to fit in the encasing with the control unit and microcontroller. Each component on the sleeve requires +3.3V. Because the battery provides +3.7V, a voltage higher than what is needed for each module, a linear voltage regulator will be used to scale the voltage down to +3.3V, as shown in Fig. 11.

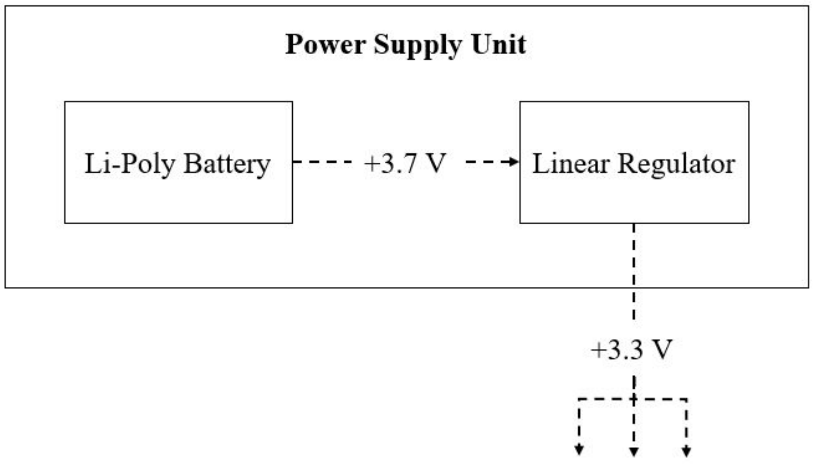


Fig. 11. Power Module Block Diagram.

Power Requirements
The battery must be able to supply enough power to ensure full functionality of the three subsystems simultaneously for at least four hours. The operating voltage and the current drawn by each component were used to calculate the voltage and current delivery requirements of the power supply. The power requirements for all the components on the sleeve are shown below in Table 6.

Table 6: Current and Supply Voltage Requirements for Sleeve Components.

Component	**Current Requirements**	**Voltage Requirements (Typical Operating Voltage)**
ICM-20948 9DoF IMU Breakout Board	Max: 3.11 mA	3.3 V
Bluetooth Module (HC-05)	30 mA	3.3 V
ATMega328P Microcontroller	1.5 mA	3.3 V
1-position DIP Switch	Max: 25 mA	Max: 24 V
Status RGB LED	R: 30 mA (max) G: 25 mA (max) B: 30 mA (max)	R: 2.2 V G: 3.3 V B: 3.3 V
LP2985 Low-Noise Low-Dropout Voltage Regulator	Output: 150 mA	Supply Input Voltage: Min: 2.2 V Max: 16 V Relevant Fixed Output Options: +3.3V
Lithium Ion Polymer Rechargeable Battery	Typical: 2500 mAh	Nominal: 3.7 V

To calculate the battery capacity in mA-hours needed for the subsystems on the sleeve, we add the maximum currents drawn by each subsystem:

$$3.11mA + 30mA + 1.5mA + 25mA + 30mA = 89.61mA$$

If we want the fully charged MBA sleeve to last 4 hours, we utilize the following equation:

$$\frac{BatteryCapacity(mA \cdot hours)}{CurrentDraw(mA)} = BatteryLife(hours)$$

It is good practice to assume our battery will have less than ideal battery life. This can be compensated with the assumption that a real-world battery life will be 75% of its theoretical value. The parameters used for calculating the battery capacity on the sleeve are 89.61 mA for the current draw and 5.5 hours (approximately 1.5 hours more than we would expect) for battery life. This results in a battery capacity requirement of at least 492.86 mAh. Because the largest voltage needed for these components is 3.3 V, we would need a battery that can supply at least this amount as well. We choose a Li-Poly battery with a nominal voltage supply of 3.7 V and a typical supply current of 2500 mAh to extend the lifetime of the sleeve to approximately 27 hours before charging is needed.

Voltage Regulators

Because the IMU breakout board, the Bluetooth module, and the microcontroller require the same voltage inputs of 3.3V, only one voltage regulator will be used to satisfy the voltage needs of each component. To ensure the components are powered safely, the voltage regulator will adjust the voltage from the Li-Poly battery to the 3.3V. Because the voltage regulator is enclosed with the IMU, Bluetooth module, and microcontroller, it will supply each with this voltage via power tracks on the PCB interface.

A linear regulator scales its input voltage down to a fixed voltage at its output terminal maintained by a feedback loop. The LP2985 low-dropout (LDO) and low-noise regulator will be used because it has stable behavior despite the output voltage being close to the input voltage value, improving its power efficiency. It is able to output our desired voltage quantity of 3.3V, and its low-noise output ensures that the incoming power supply or transients in the load will not affect the stability of the voltage being supplied to the subsystems [16].

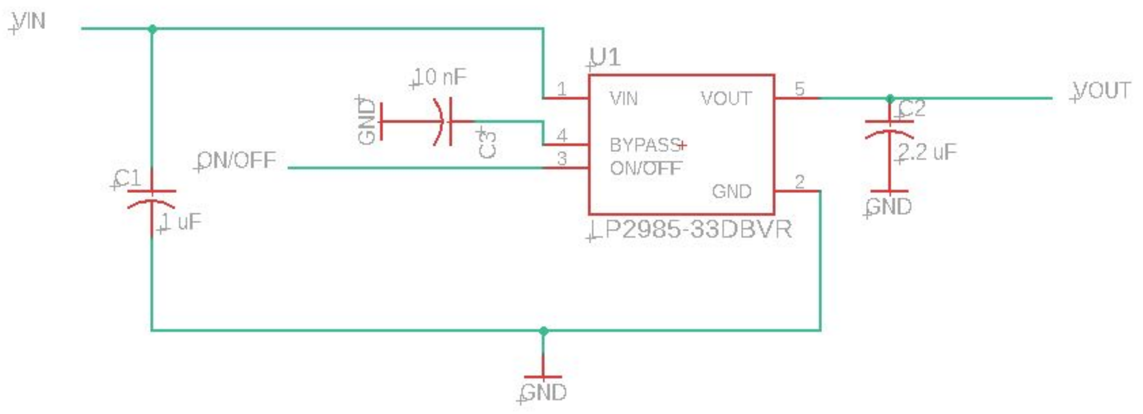


Fig. 12. LP2985 Linear Regulator Typical Application Schematic.

Table 7: Pin Connections for LP2985 Linear Regulator.

Pin	**Function**	**Connection**
1	VIN	Supply Input (3.7 V). This will be connected to the battery.
2	GND	Ground to PCB ground.
3	ON/OFF	Active-low shutdown pin. Tie to VIN if unused.
4	BYPASS	Attach a 10 nF capacitor to improve low-noise performance.
5	VOUT	Voltage Output (3.3 V). The control unit, IMU breakout board, and the Bluetooth module will be connected to this pin in parallel.

Table 8: R&V table for the power supply unit subsystem.

Requirement	**Verification**
Li-Poly Battery 1) Supply +3.7V ± 5% power and stores 2500 mAh ± 5% of charge.	Li-Poly Battery 1) The current output will be confirmed to be within the acceptable range with a digital multimeter. A stable voltage output within the acceptable range will be

	confirmed with an oscilloscope waveform. *Verification Process:* 1) Ensure the battery has been fully charged (reads $+3.7V \pm 5\%$). 2) Attach DMM probes to respective JST connector pins of the battery. 3) Measure and record I and V at 5 minute intervals for 60 minutes. 4) Perform midpoint Riemann summation on current measurements. 5) Confirm that at least 2375 mAh was extracted.
2) Lifetime per charge of the sleeve should be at least 4 hours of usage.	2) The lifetime of the sleeve will be measured while it is turned on. *Verification Process:* 1) Ensure the battery has been fully charged (reads $+3.7V \pm 5\%$). 2) Attach DMM probes to respective JST connector pins of the battery. 3) Turn on the sleeve and make Bluetooth connection to the user's computer so that the sleeve is in full operation mode. 4) Measure and record I and V at 2 hour intervals and at half hour intervals as the hours in operation reaches 25 hours (calculated lifetime of 1 charge is 27 hours). 5) Terminate verification process when the $V_{battery} < +3.3$ V and record the time this occurs.
3) Must not exceed an operating temperature that can damage the user's skin (44℃) [17].	3) The temperature of the sleeve will be measured with a thermocouple thermometer while it is turned on. *Verification Process:* 1) Ensure the battery has been fully charged (reads $+3.7$ V $\pm 5\%$). 2) Attach DMM probes to respective JST connector pins of the battery. 3) Turn on the sleeve and make Bluetooth connection to the user's computer so that the sleeve is in full operation mode. 4) Measure and record the temperature at 30 minute intervals. 5) Terminate verification process when the

	fourth hour is reached.
<u>Linear Voltage Regulator</u> 1) Voltage must be regulated to +3.3V ± 5% for the Bluetooth module, microcontroller, and the VDD pin on the IMU	<u>Linear Voltage Regulator</u> 1) Stable voltage outputs at the desired values will be measured and observed through oscilloscope waveforms. *Verification Process:* 1) Attach the oscilloscope GND probe to GND of the PCB and the signal probe to the VDD input pin of the Bluetooth module. 2) Supply regulator with 3.7V DC from a power supply. 3) Ensure output voltage remains 3.3V. 4) Repeat steps 1-3 for the microcontroller and the VDD pins on the IMU.

2.4 Tolerance Analysis

The most important component of our project is our IMU, since our high-level design relies entirely on collecting clear acceleration data. As stated before, if an object in motion comes to a quick and sudden stop, the magnitude of the acceleration of the object will sharply increase, then sharply decrease after a short period of time. Therefore, if we calculate the 3-dimensional hypotenuse using the 3 directional components of acceleration the IMU records, we should be able to observe that there are short periods of time where the magnitude of acceleration is significantly greater than its resting point.

$$|a| = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

Fig. 13. The equation used to acquire the magnitude of three-dimensional acceleration.

At rest, we would expect an object's magnitude of acceleration to be 1*g*. The is because the x and y components should be 0*g*, but the z component will be 1*g* because of gravitational force. When an object is moving in any direction, we can expect certain directional components of acceleration to shift towards higher negative or positive values. However, we can be certain with conducting motions that the magnitude of acceleration will increase to values higher than usual. Below, two separate examples are shown of this to demonstrate that despite having different magnitude peaks, different tempos produce similar spikes in acceleration.

We do not have access to simulating ICM-20948 data, but we have collected data via an Android phone that uses an LSM6DSO accelerometer, which ran at a slower sampling rate of approximately 392 Hz. Despite the low sampling frequency, we were able to plot the total acceleration of arm motions conducting at a tempo of 180 BPM for 16 beats as seen in Fig. 14, and found that the magnitude of acceleration had clearly identifiable spikes, with every stop of the arm registering a value of at least 6*g*.

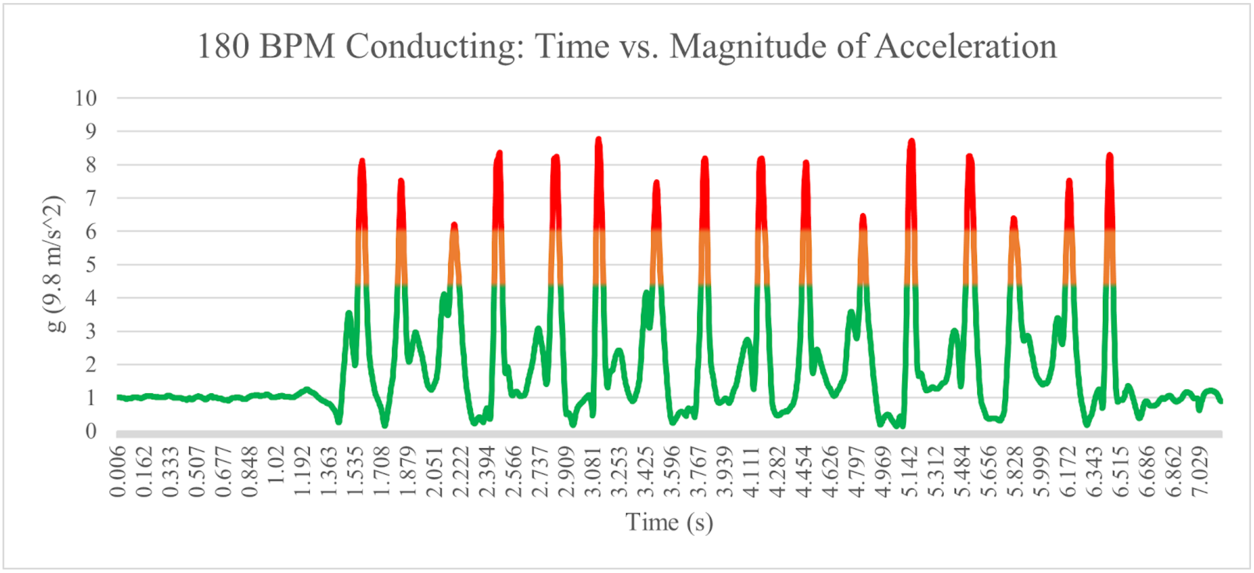


Fig. 14. A graph of the magnitude of acceleration recorded by the LSM6DSO over time when conducting at 180 BPM.

For Fig. 14, data values with a value of above 6*g* are shown in red, while data values between 4.5*g* and 6*g* are shown in orange. The lowest number of samples registered from a peak above 6*g* is 4 samples for the 3rd beat. Note that the high-pass limit does not have to be 6*g* in this instance, and could reasonably be as low as 4.5*g* since there does not seem to be any noise anomalies in data above the 4.5*g* mark. Using the new threshold, the lowest number of samples registered from a peak above 4.5*g* is 15 samples from the 11th beat. Extrapolating for the ideal frequency our ICM-20948 will run at (1.125 kHz), we can estimate that we will get at least 43 samples of clear beat data when conducting at 180 BPM.

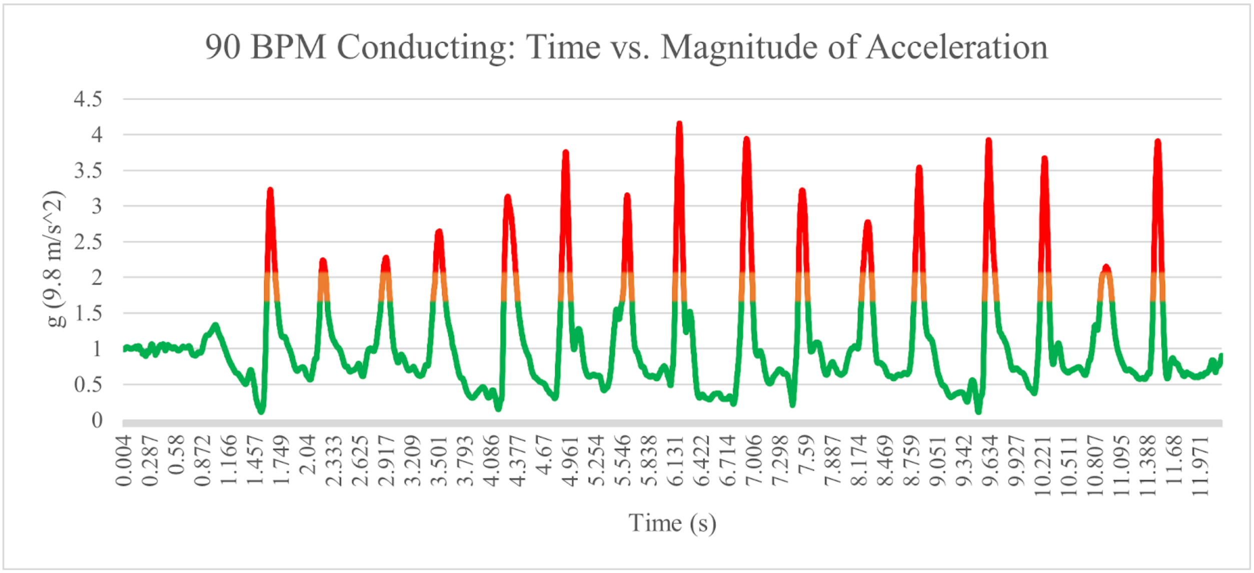


Fig. 15. A graph of the magnitude of acceleration for conducting at 90 BPM.

Fig. 15 uses the same method as Fig. 14, but at a lower tempo of 90 BPM to demonstrate the differences in conducting motions. For Fig. 15, data values with a value of above 2*g* are shown in red, while data values between 1.6*g* and 2*g* are shown in orange.On one hand, the data is not only clear, but has more samples each acceleration spike due to the halved conducting speed. The shortest peak above 1.6*g* is the second conducting motion, which lasts for 34 samples. Extrapolating for the ICM-20948's frequency, we can estimate that we will get at least 97 samples of clear beat data when conducting at 90 BPM.

Unfortunately, the spikes in acceleration are considerably smaller, with the highest value in Fig. 15 (4.2*g*) being less than the high-pass limit we imposed on Fig. 14. Therefore, we cannot expect to calculate when a beat is conducted by detecting when our acceleration exceeds a fixed amount. However, what we can do to detect beats is create an algorithm that stores the past 100 samples, find the maximum value of total acceleration, apply a high-pass filter dependent on that maximum value to filter out noise, and use the values surrounding the local maximum to verify that the user indeed conducted a beat. While the data we received in both edge cases produced different values, the similar trends and data quality lead us to believe that calculating when spikes in acceleration is possible.

It is fair to claim that we will not receive the exact same measurements from the ICM-20948. However, we have reason to believe that the LSM6DSO accelerometer has provided us with valuable reference data. The ICM-20948 and LSM6DSO share the same supply voltage, full-scale range options, and even linear acceleration sensitivities across full-scale ranges [18]. The only significant difference between the two IMUs is that the LSM6DSO requires less current to run, but that is insignificant given that their performances would not differ if the proper current is applied to both IMUs. We are confident that we will be able to collect and isolate the data we need at at least 1 kHz if our test data with a similar IMU at a lower frequency produced clear results.

2.5 COVID-19 Contingency Plan

In the case we transition to a completely online Senior Design semester, we are confident that we will still be able to work as a team and coordinate to finish the project. All three members of our team will be on campus the entire semester, even if the university shuts down in-person instruction due to COVID. As a result, we will be able to hand off physical parts to each other for building and testing. We also have access to basic electrical equipment in our homes; this includes, but is not limited to, Arduinos, multimeters, and soldering equipment.

Our most significant concern for a transition to an online semester due to a COVID outbreak is whether it will affect our ability to test the PCB design effectively. In the event of a fully online transition, we should still be able to solder the parts onto our PCB with our personal soldering equipment, as long as all the parts are in our possession. Necessary parts can be shipped to our personal addresses if needed. However, we do not own high-grade equipment and tools such as the ECE lab oscilloscopes, so we may run into issues while testing our project at home.

In the event that we don’t have the sufficient tools or materials to construct a PCB, we would breadboard our circuit. Creating a model that can be fit on an arm is possible, though it would not be as neatly packaged as originally intended. The breadboard model would also not be as secure as a PCB model, so parts could come loose while the product is in use. Even if we are able to create a functional test unit before a COVID outbreak, we will not be able to test the device on anyone other than ourselves, who all lack true drum major or general conducting experience.

If we are able to construct a functional circuit and acquire all the necessary parts, we do not anticipate that our requirements or verifications will have to change. Most of our verification processes do not require high-grade equipment, and are able to be completed in our personal homes. The only exception to this would be our battery testing. The battery testing will require high-grade equipment and tools, such as the oscilloscope and lithium safety bags, that we only have access to in the lab. In addition, the battery testing will require the help of the power-centric TAs in-person.

The biggest hurdle for a fully online semester would be properly testing our PCB. In the event that we are unable to design a functional PCB, we would create a breadboarded version of our intended design.

3. COST ANALYSIS

3.1 Costs of Labor

Our research shows that the average salary nationwide for a Consumer Electronics Engineer is $75,591 as of September 24, 2020 [19]. Assuming an engineer worked 40 hours a week for 52 weeks, their hourly wage would be $36.34. We will assume that we will work on the project on 1 device for 10 hours a week for 10 weeks, and that we will need 2.5 times the amount of cost as originally expected to account for overhead. Therefore, we estimate the total labor costs to be 3 people * 36.34 dollars an hour * 10 hours * 10 weeks * 2.5 = **$27,255**.

3.2 Costs of Parts

Table 9: Cost and quantity of parts.

Name	Manufacturer	Part Number	Quantity	Unit Price ($)
Rechargeable Li-Poly Battery 3.7V	Adafruit	LIPO785060	1	11.49
USB Li-Ion/Li-Poly Charger v1.2	Adafruit	MCP73833/4	1	19.42
Mini B to USB Adapter Cable	Amazon	N/A	1	6.28
LDO Voltage Regulator	Texas Instruments	LP2985-33DBVR	1	0.51
Inertial Measurement Unit	Invensense	ICM-20948	1	5.91
HC-05 Bluetooth Module	DSD TECH	HC-05	1	8.99
ATMega328P MCU	Microchip	ATMega328P	1	2.08
1-Position DIP Switch	CUI Devices	DS04-254-SMT	1	.70
Status RGB LED	Kingbright	WP154A4SEJ3VBD ZGC/CA	1	1.99

Velcro Heavy Duty 4”x2” Fastener Strips	VELCRO	N/A	1	5.79
Nylon Fabric 60”x36”	Emmakites	N/A	1	9.95

3.3 Total Costs

Table 10: Total costs of project, including labor and parts.

Section	**Cost**
Labor	$27,225.00
Parts	$73.11
Total	**$27,298.11**

4. SCHEDULE

Table 11: Weekly project schedule for each member.

Week	**Wynter**	**Alyssa**	**Prashant**
10/04/20	Work on designing PCB, order breadboard parts	Confirm use of subsystem modules after Design Review	Work on designing PCB
10/11/20	Research design techniques for wearable consumer electronics	Work on breadboard prototype	Finish PCB design and order via PCBWay
10/18/20	Work on packaging design, construct PCB if possible	Work on microcontroller program, work on data collection algorithm	Start work on user display, start testing PCB if possible
10/25/20	Work on user display, work on microcontroller program if necessary	Work on microcontroller program, test PCB	Gather test data from PCB, work on user display
11/01/20	Verify Bluetooth capability	Finalize microcontroller program	Verify Bluetooth capability
11/08/20	Connect functional subsystems together	Optimize data collection algorithm on MCU program	Finish user display
11/15/20	Final Preparations for Demo	Final Preparations for Demo	Final Preparations for Demo
11/22/20 (Thanksgiving Break)	Final Preparations for Presentation	Final Preparations for Presentation	Final Preparations for Presentation
11/29/20	Final Preparations for Presentation	Work on Final Paper	Work on Final Paper
12/06/20	Work on Final Paper	Work on Final Paper	Work on Final Paper

5. ETHICS AND SAFETY

This device aims to incorporate technology into musical training in a new and innovative way, in order to “contribute to society and to human well-being,” as stated in the ACM Code of Ethics [20]. The goal of this device is to assist drum majors by providing a learning tool. Both drum majors and marching band members may benefit from it.

The IEEE Code of Ethics states that members have a responsibility “to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies” [21]. To help users understand the capabilities of this new technology, an instruction manual will be provided. It will be made clear in the manual how the device is used.

The project makes use of a Bluetooth module to transmit data. There are potential security risks associated with Bluetooth usage, including data theft [22]. While the sleeve does not transmit sensitive or personal data, the host computer could have personal data vulnerable to a Bluetooth security breach. Therefore, it would be recommended that the device be used in a trusted place without significant wireless interference. Additionally, the Bluetooth module should be powered off when not in use.

The safety of the user is the top priority of this device, as the IEEE Code of Ethics clarifies that it “holds paramount the safety, health, and welfare of the public” [21]. Since the device is placed directly on the user’s arm, it is important that the electrical components do not overheat or shock the user. The lithium-ion battery is flammable, so additional precautions must be taken; there will be regulators to prevent the battery voltage from decaying below 3.0 V/cell or exceeding 4.2 V/cell [23]. No universal standards for wearable technology involving fabric could be found, but precautions will be taken to protect the user. Heat-resistant and insulating fabric will be used for the sleeve to protect the user’s arm. The electrical components will also be placed in a box, and therefore will not touch the fabric directly.

There are a number of safety precautions to take into account when we perform testing and verification on our device. One concern is testing for the battery. The Li-Poly battery is flammable, so it is important to keep the testing bench clear while testing. We will take fire safety and fire extinguisher training before testing occurs. Charge and discharge tests will be performed while the battery is inside a specifically designed lithium safety bag [23]. We will also work with power-centric TAs to verify our battery conditions before implementing it into our circuit.

Another safety concern is soldering. Goggles will be worn while soldering, and soldering will be done in a well ventilated area due to the fumes produced from soldering [24]. The soldering iron reaches very high temperatures, so it is important to not touch the tip and return the iron to its stand when not in use. Non-lead solder will be used to avoid lead exposure.

6. REFERENCES

[1] R. Raymond and C. Aliga. Today's Drum Major | Central Iowa Color Guard and Drum Major Camp 2016. (2016). Accessed: Oct. 1, 2020. [Online]. Available: http://www.centraliowacolorguard.com/upload/398755/documents/Drum%20Major%20Handbook%202016.pdf

[2] *DB-90: Dr. Beat*. BOSS. Accessed: Oct. 1, 2020. [Online]. Available: https://www.boss.info/us/products/db-90/.

[3] *Smartmusic*. MakeMusic. Accessed: Oct. 1, 2020. [Online]. Available: https://www.smartmusic.com/.

[4] TDK. *ICM-20948*. (2017). Accessed: Oct. 1, 2020. [Online]. Available: https://product.tdk.com/info/en/documents/catalog_datasheet/imu/DS-000189-ICM-20948-v1.3.pdf

[5] Sparkfun. *SparkFun 9DoF IMU (ICM-20948) Breakout Hookup Guide*. (2019). Accessed: Oct. 1, 2020. [Online]. Available: https://learn.sparkfun.com/tutorials/sparkfun-9dof-imu-icm-20948-breakout-hookup-guide

[6] J. Kopstein. "Marching Speeds." Altissimo! Recordings. https://militarymusic.com/blogs/military-music/13516233-marching-speeds (accessed Oct. 1, 2020)

[7] Components101. *AtMega328P Microcontroller.* (2018). Accessed: Oct. 1, 2020. [Online]. Available: https://components101.com/microcontrollers/atmega328p-pinout-features-datasheet

[8] CUI Devices. *DIP Switch | DS01-254*. (2019). Accessed: Oct. 1, 2020. [Online]. Available: https://www.cuidevices.com/product/resource/ds01-254.pdf

[9] Omron. *Technical Explanation For Basic Switches*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.ia.omron.com/data_pdf/guide/29/microswitch_tg_e_3_2.pdf

[10] Components101. *HC-05 Bluetooth Module*. (2018). Accessed: Oct. 1, 2020. [Online]. Available: https://components101.com/wireless/hc-05-bluetooth-module

[11] “Modify the HC-05 Bluetooth Module Defaults Using AT Commands.” Instructables Circuits. (2013). https://www.instructables.com/Modify-The-HC-05-Bluetooth-Module-Defaults-Using-A/ (accessed Oct. 1, 2020).

[12] Python. *Tkinter - Python interface to Tcl/Tk*. (2020). Accessed: Oct. 1, 2020. [Online]. Available: https://docs.python.org/3/library/tkinter.html

[13] *PyBluez*. (2019). Accessed: Oct. 1, 2020. [Online]. Available: https://readthedocs.org/projects/pybluez/downloads/pdf/latest/

[14] PKCell. *Li-Polymer Battery Technology Specification*. (2019). Accessed: Oct. 1, 2020 [Online]. Available: https://cdn-shop.adafruit.com/product-files/328/LP785060+2500mAh+3.7V+20190510.pdf

[15] A. Udanis. “How to Choose the Best Battery for Your Next Project.” All About Circuits. (2015). https://www.allaboutcircuits.com/technical-articles/how-to-choose-the-best-battery-for-your-next-project/ (accessed Oct. 1, 2020).

[16] A. Veeravalli, S.M. Nolan. “Introduction to Low Dropout (LDO) Linear Voltage Regulators.” Design & Reuse. https://www.design-reuse.com/articles/42191/low-dropout-ldo-linear-voltage-regulators.html#:~:text=A%20low%20dropout%20(LDO)%20linear,voltage%2C%20improving%20its%20power%20efficiency (accessed Oct. 1, 2020).

[17] B. Christopher. “Design Safe Wearable Technology with Heat Transfer Modeling.” COMSOL. (2016). https://www.comsol.com/blogs/design-safe-wearable-technology-with-heat-transfer-modeling (accessed Oct. 1, 2020).

[18] STMicroelectronics. *LSM6DSO*. (2019). Accessed: Oct. 1, 2020. [Online]. Available: httpsp://www.st.com/resource/en/datasheet/lsm6dso.pdf

[19] “Consumer Electronics Engineer Salary.” Ziprecruiter. (2020). Accessed: Oct. 1, 2020. [Online]. Available: https://www.ziprecruiter.com/Salaries/Consumer-Electronics-Engineer-Salary

[20] D. Gotterbarn. *ACM Code Of Ethics And Professional Conduct*. (2020). Accessed: Oct. 1, 2020. [Online]. Available: https://www.acm.org/code-of-ethics

[21] IEEE. *IEEE Code Of Ethics*. (2020). Accessed: Oct. 1, 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[22] “Wireless Connections and Bluetooth Security Tips”. Federal Communications Commission. https://www.fcc.gov/consumers/guides/how-protect-yourself-online (accessed Oct. 1, 2020).

[23] Spring 2016 Course Staff, Champaign, IL, USA. *Safe Practice For Lead Acid And Lithium Batteries*. 2016. Accessed: Oct. 1, 2020. [Online]. Available: https://courses.engr.illinois.edu/ece445/documents/GeneralBatterySafety.pdf

[24] “Soldering.” Stony Brook University. https://ehs.stonybrook.edu/programs/laboratory-safety/laboratory-equipment/soldering (accessed Oct. 1, 2020).

APPENDIX A: HC-05.ino Program

```
/*
AUTHOR: Hazim Bitar (techbitar)
DATE: Aug 29, 2013
LICENSE: Public domain (use at your own risk)
CONTACT: techbitar at gmail dot com (techbitar.com)
*/

#include <SoftwareSerial.h>

SoftwareSerial BTSerial(10, 11); // RX | TX

void setup()
{
  pinMode(9, OUTPUT);  // this pin will pull the HC-05 pin 34 (key pin) HIGH to switch module
to AT mode
  digitalWrite(9, HIGH);
  Serial.begin(9600);
  Serial.println("Enter AT commands:");
  BTSerial.begin(38400);  // HC-05 default speed in AT command more
}

void loop()
{

  // Keep reading from HC-05 and send to Arduino Serial Monitor
  if (BTSerial.available())
    Serial.write(BTSerial.read());

  // Keep reading from Arduino Serial Monitor and send to HC-05
  if (Serial.available())
    BTSerial.write(Serial.read());
}
```