# Snowboarding Data Logger

Revised 4/23/2023

Corbin Fleming-Dittenber (corbinfd), Wynn Kaza (wynnkaza)
Alexander Guido (ajguido), Aiden Ascioti (aascioti)

**Customer:** We sent out a survey to members of the Michigan Snowboarding Club and got 17 responses. Of the people we surveyed, 13 said they would be interested in a product that could automatically track days and runs on the mountain. However, only 47% said they currently track their speed down the mountain. From there only 2 people said they track their speed more than once or twice on a new mountain. The people that don't already track their speed said they were worried about draining their phone's battery or that they had never thought about it. This suggests that there is a market of people that would be interested in tracking their run data at least every once in a while, especially if it is automatic and easy. The fact that people track so infrequently right now might suggest that people won't be as willing to buy our product because they might be worried about not using it often enough. Despite this, 16/17 people said tracking their path down the mountain might be interesting to them, and 13/17 said it might be useful.

At the end of the survey, we described the potential product and asked what they might be willing to pay for such a device. The answers were rather split with 7/17 people saying they would pay less than $50 or that they weren't interested, and the other 10/17 people saying they would be willing to pay $50-100 or more. Given the answers given in other parts of the survey, it would be best to shoot for the $50-100 price range despite 4 people saying the most they would pay is $100-200. Earlier answers did not indicate that people would be confident they will use the product a lot, so a high price point might not be possible from the start at least.

**Value:** As collected from a google survey that was run, many individuals who snowboard would like a device capable of tracking themself and friends while on a mountain. This device allows a customer to track their speed and location using a GPS as they go downhill. Also, the IMU will allow the user to see how steep a given run is. Having this data will help provide information about the different speeds they have on different parts of the mountain, what slopes they went down, and various other data that would be useful to a snowboarder. Furthermore, while there are mobile applications that accomplish a similar goal, they drain the user's battery very quickly, so by utilizing an external low-power device it will save cell phone battery life while on the mountains where it is difficult to charge your phone.

**Approach:** Our project is a snowboard data tracker that is meant to track position on mountains while also tracking speed. The device is about the size of a boot and is meant to be mounted to a board in between the boarders feet. The logger has the ability to communicate with the user via the user's cell phone and a low power screen on the device housing. We researched various technologies to improve the battery life of the device. During demonstration, we wanted to provide demonstrations of the device in action but it was not possible due

to time constraints, but we were able to let people use the device because it was able to get a GPS fix with the windowed ceiling in the demonstration area.

**Proposed Major functions of our project**:

1. Location and speed tracking
    a. IMU for tracking speed and location of user

We want to be able to track the maximum speed of users, so we will use an IMU. Specifically, we are going to use the LSM6DSO breakout board from Sparkfun. This board is low-power which is ideal for our application. It includes an accelerometer, gyroscope, and temperature sensor which when used in combination can help us track speed and also location. We plan to use the IMU to track location and then to use the GPS to help adjust for drift.

    b. GPS for tracking the location and speed of the user

We are going to use a breakout board from Adafruit because it is relatively low power and will take less time to set up than a custom board. The Adafruit GPS module uses a CD-PA1616S GPS patch antenna module which has a current consumption of 25mA for acquisition and 20mA for tracking which is lower than alternatives from Sparkfun and Amazon.

    c. Software for gathering and storing data

After a day out skiing the user will need to be able to see their data somehow. We will create a piece of software that will take a user's data from the data logger and turn it into interesting statistics and graphs. It will display information like maximum speed, maximum altitude, vertical distance traveled, run location data, and highlighted locations. Data will be stored on an SD card via the Adafruit MicroSD card breakout board. Data will also be transferable to a computer via USB.

    d. Wireless Communication

The wireless system must have a range of at least 10 feet and be able to continuously communicate with a phone or other device for at least 8 hours. Ideally, it will act as a wireless UART so that a simple command set can be developed to convey data information from the IMU and GPS to a phone. This will be done using an Adafruit Bluefruit LE UART Friend - Bluetooth Low Energy (BLE).

2. Data display
    a. Low-power character display to display basic information

We want users to be able to see if the recording started or what their maximum speed was on the last run and etc. This will be displayed on an LCD screen from Digikey with a backlight so it can be seen at night or day. Without the backlight, the screen only draws 3 mA, but the backlight current is 20 mA. The screen and backlight will turn off when not in use. The backlight will also only turn on when needed as detected by the light sensor.

3. Power management
    a. Onboard 2+Ah lithium battery to provide a full day of operation without failure.

The Logger will feed energy from the onboard lithium battery through a 3.3V output Switching Regulator to provide stable power to the microcontroller and various sensor modules. A switching regulator was selected to maximize efficiency and utilize the entire battery output voltage range. The input and output of the battery will be fused. The microcontroller will be able to control charging, with an additional IC to prevent overcharging/discharging. The exact size of the cell is to be determined by March 10th once the components are finalized.

    b. Temperature sensor to track battery efficiency

The IMU chip has a temperature sensor, which will be used in conjunction with stored battery data to provide accurate real-time %SOC measurements to the user and disable charging in emergency situations.

    c. Charging capabilities via USB port

The Logger will direct the Nucleo Devkit's on-board 5V output to the STBC08PMR Linear Battery Charger IC. The IC provides constant voltage/constant current charging to the lithium battery housed within the module.

    d. Qi-Compliant Wireless Charging

The Logger will be capable of Qi-Compliant wireless charging. We plan to utilize the BQ51013 Qi Compliant Wireless Power Receiver chip from TI and a set of inductive charging coils rather than using a breakout module. We opted to utilize the chip and coils directly in order to minimize the cost and packaging, but will include the ability to utilize a given module should our implementation not function reliably.

    e. Solar power harvesting to improve lifetime

While our initial proposal investigated the viability of piezoelectric vibration power harvesting, we found the gain negligible when compared to the complexity. By contrast, adding a solar panel to the power distribution system is simpler and likely to yield much higher power gain. However, due to anticipation that the Logger will remain in pockets for much of its lifespan, solar power is deemed to be of secondary importance. Ideal diode ICs

will provide protection for individual circuit components while minimizing power consumption, critical for our design.

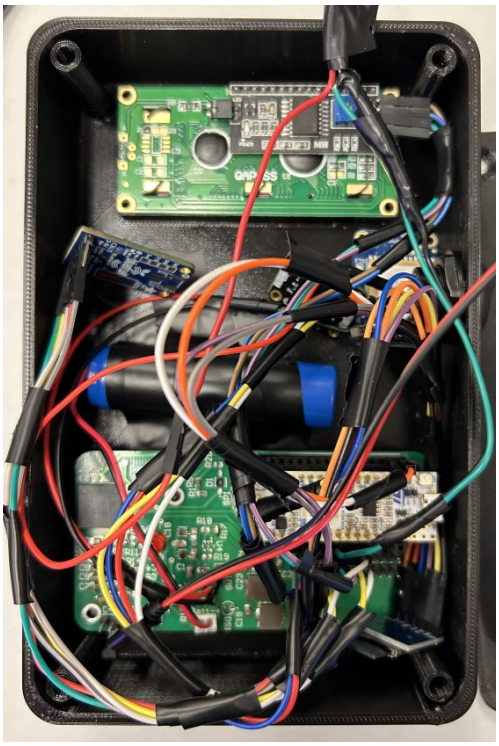

Figure 1: The Inside of our device

**Essential System Components:**

1. Sensors
    a. IMU LSM6DSO
    b. GPS Adafruit PA1616S
    c. Temperature sensor LSM6DSO
2. Power Management
    a. Battery
        i. Linear Battery Charger STBC08PMR
        ii. Battery Protection IC AP9101CK
    b. Wireless Power
        i. Qi Receiver IC BQ51013
        ii. Receiver Coils
    c. Miscellaneous
        i. 3.3V Switching Regulator ZXLD1615
        ii. Ideal Diode LM66100
3. Processing and Output
    a. ARM Microcontroller NUCLEO-L432KC
    b. Low-Energy Bluetooth Module Adafruit Bluefruit LE
    c. Onboard Memory for logging Adafruit MicroSD card breakout

4. Miscellaneous
    a. USB-compatibility (either the MCU or via breakout module)
    b. Casing
    c. LCD display NHD-0208AZ-FSW-GBW-33V3

**Optional System Components:**

1. Sensors
    a. Ambient Light Sensor to adjust display outputs
2. Power Management
    a. Solar Power
        i. Boost Converter AP3015A
        ii. 277mW Solar Panel Array SM14K09L
3. Processing and Output
    a. Buzzer for enhanced user experience CC07MP025M16-2700
    b. Button for User to switch between modes SS314MAH4-R

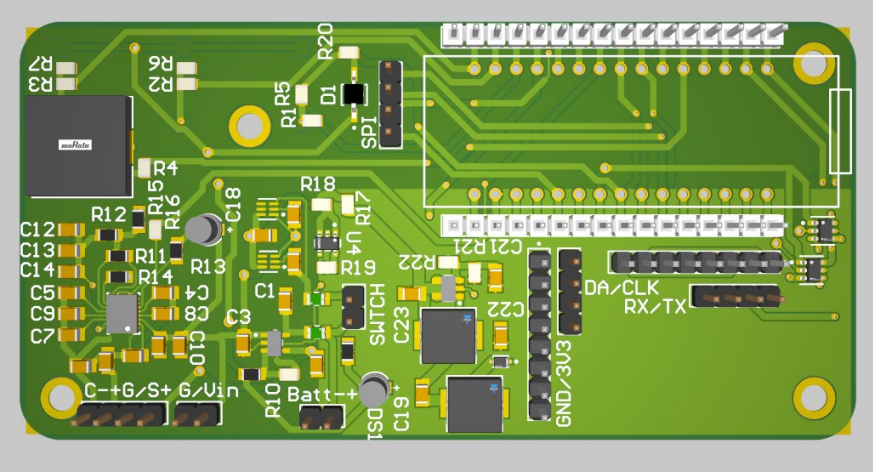


Figure 2: Our custom, 2-sided PCB

**Timeline**

Board Design for power systems [March 10th]

- As one of the goals of our project is to implement simplicity, having a lot of the small components used to power the system can be compressed and simplified by including them on a PCB. This board should pass any board design constraints and be fully built within Altium to be ready for ordering on March 10th. This early deadline is set to ensure proper timing for soldering and testing of the board. Additionally, all components for the PCB should be ordered alongside the deadline.

Wireless Communication [March 10th]

- The interface between the system and the cellphone/device should be prototyped and ready to work by this period. The hardware for this will be accomplished by using the Adafruit BLE UART Friend, which leads the majority of the difficulty up to debugging the software to work with communication. This device must work from at least 10 ft. By the March 10th deadline, the phone should be able to receive correct communication from the system.

Display [April 4th]
- Our system will include a character display that is integrated into the system and is capable of displaying data directly on the device. While this is not as high priority as other components, this is an essential part of our project. This display will communicate with the chip using a 4/8-bit MPU interface based on the required characters.

Various System Components [April 4th]
- There are many smaller parts of the system that must include software and hardware. This includes the buzzer, 3-way switch (for switching between different modes on the logger), and the firmware for the chip should be written at this point. Furthermore, all firmware for the chips should be finalized, including communication, to give time for debugging in the weeks before demonstration.

Mobile Application
- If time exists before the demo, and the rest of the system has been designed and tested, another goal would be to implement a visual mobile application which can display the data from the system in a more human-friendly format. This is not a focus of the project at all, and only an additional feature that will be implemented if time permits.

**Claim Difficulty**
IMU 0.5 (Complete)
GPS 0.5 (Complete)
SD Card 0.25 (Complete)
Combined Positioning Algorithm 0.25 (Abandoned)
Bluetooth Communications 0.5 (Complete)
LCD Display 0.5 (Complete)
Light Sensor 0.25 (Abandoned in return for battery percentage, both are ADCs)
Temperature Sensor 0.25 (Complete)
User interfacing (buttons, buzzer, packaging) 0.25 (Complete)
Custom PCB: 1.0 (Complete)
Added: Sophisticated Driver Code: 0.5 (Complete)

Total: 4.5

**Explanations for Parts:**

**IMU:** In the final product, the IMU was used to measure the slope during a run, and maintain current slope and final slope. This was accomplished by tracking the system's orientation at startup, and subtracting the measurement during the run to find what angle the device is. The benefit of this is that the device can be at any initial position and it will measure slope. Furthermore, many additional functions were coded for the IMU such as the ability to save orientation for calibration when the device was powered on/off. Additionally, we used a timer interrupt to signal when to receive data from the IMU.

**GPS:** The GPS was used to find location, altitude, and speed. The GPS was very hard to get working correctly due to the way that the data was being sent from the device. For the GPS, many functions were required, such as parsing the data, validating the data with the checksum, and interrupts for when the data was being sent to the STM.

**SD Card:** The SD Card was mainly added with the FatFS file system. There was not much difficulty in adding the library using online tutorials, and therefore only received 0.25 points of difficulty.

**Combined Positioning Algorithm:** The original intent of the algorithm was to have the ability to track location and speed with the IMU while the GPS was not being used. However, testing different filters (Kalman, Complementary), we were unable to get accurate measurements of velocity that did not have significant drift. Some factors that could have led to the failure of this device include the sampling rate of the IMU, the drift errors from the filters, and the overall difficulty of linking IMU and GPS data together. Overall, recognizing the difficulty of fusing the GPS and IMU data after many hours spent working on it, we decided on switching to measuring only the angle of the slope.

**Bluetooth Communication:** For bluetooth communication, we decided upon using the Adafruit App to send data over. With more time, we would have worked on building an app that could parse the bluetooth data received and graph it for the user. However, in the current state, the bluetooth module is able to send data over UART to the bluetooth module, and the application is able to display the data received in a command line interface.

**LCD Display:** The LCD Display took multiple days to get working due to a lack of library support. Once sufficient modulation was built, the LCD Display was used with the FSM to display useful data to the user on the screen.

**Light Sensor:** The light sensor was abandoned due to the Nucleo Board only having one ADC input left on it. As we wanted to be able to measure the battery percentage of the device, we opted for using that ADC input for the battery percentage instead.

**Temperature Sensor:** The temperature sensor was overall simple to integrate and did not have many issues. The biggest issue was calculating the current temperature and making sure that the device was displaying an accurate temperature with respect to the room temperature.

**User interfacing (buttons, buzzer, packaging):** There were not too many difficulties with the user interface. Included inside this point category include the push buttons, buzzer, power on/off switch and the 3D printed case. Overall, all of these features helped to enhance the interface that a user would see and simplify how the user would interact with our device.

**Custom PCB:** The custom PCB was used to easily distribute power to the system and reduce complexity of the overall system. The PCB was used to mount our Nucleo board, providing connections for power and communications to all the breakout modules. It also included a battery charging IC and a Sepic Boost Converter to utilize the entire voltage range of our lithium battery. The PCB also included designs for a wireless charging receiver, though we were unable to get it working due to soldering tolerances. Wireless charging was instead implemented with a breakout module.

**Sophisticated Driver Code:** Another 0.5 points were added for sophisticated driver code due to the FSM that was used in our system. Developing and debugging the FSM took a while, and therefore we believe it deserves some amount of points on its own. Furthermore, within this category includes the combination of all the various systems, and the overall system collaboration between the different sensors and breakout boards included in the system.

**Team Member Contributions**

Corbin Fleming-Dittenber:

1. Worked on all of the functionality for the GPS. Obtaining useful data from the GPS was difficult so it took up most of the time on the project. This includes parsing GPS data, setting up sampling rates, and turning on/off the GPS for power saving.
2. Worked on designing and printing the CAD that was used for the box and push buttons.

Wynn Kaza:

1. Worked mainly on the IMU. Having difficulty with obtaining useful data from the IMU, so had to switch to simply displaying only the slope after many trials returned unsuccessful results.
2. Worked on adding timer interrupts to sample IMU data on accurate intervals, reduce push button buoyancy and timeout the IMU calibration.
3. Obtained functional data for temperature readings. This was later displayed on LCD and also used for battery percentage calculations.
4. Worked on system code, adding interrupts and remodeling the code to be more legible for others to read and use.
5. Used PWM to have the buzzer sound whenever the user pressed a push button. The idea behind this was so that if the user was wearing globes, they would still be able to tell that they had pressed a button.

Alexander Guido:

1. Constructed a finite state machine for displaying the correct data on the LCD.
2. Worked on interrupts for the push buttons to cycle between FSM states.
3. Worked on writing and reading data from an SD card via SPI. This results in the log being sent from the STM to the SD Card whenever a user stopped a log.
4. Worked on sending saved data to a mobile phone through Bluetooth via UART. This encompassed sending data whenever the user stopped a log similar to the SD Card.

Aiden Ascioti:

1. Worked on the PCB schematic and layout. Furthermore, worked on building the PCB and testing it after it arrived.

2. Added wireless charging capabilities to the system
3. Added an ADC input on the Nucleo that tracked battery percentage based on temperature of the system.

System Integration: Every member worked together on system integration. Testing functionality for different key parts of the system working in conjugation was a team effort and required every member of the team. Similarly, writing of the main driver code (with several thousand lines of code) on Github was a team effort that everyone on the team contributed equally to.

Weighted Member Contribution:
Corbin-Fleming Dittenber: 25%
Wynn Kaza: 25%
Alexander Guido: 25%
Aiden Ascioti: 25%

**Block Diagrams**

Figure 3: Functional Diagram

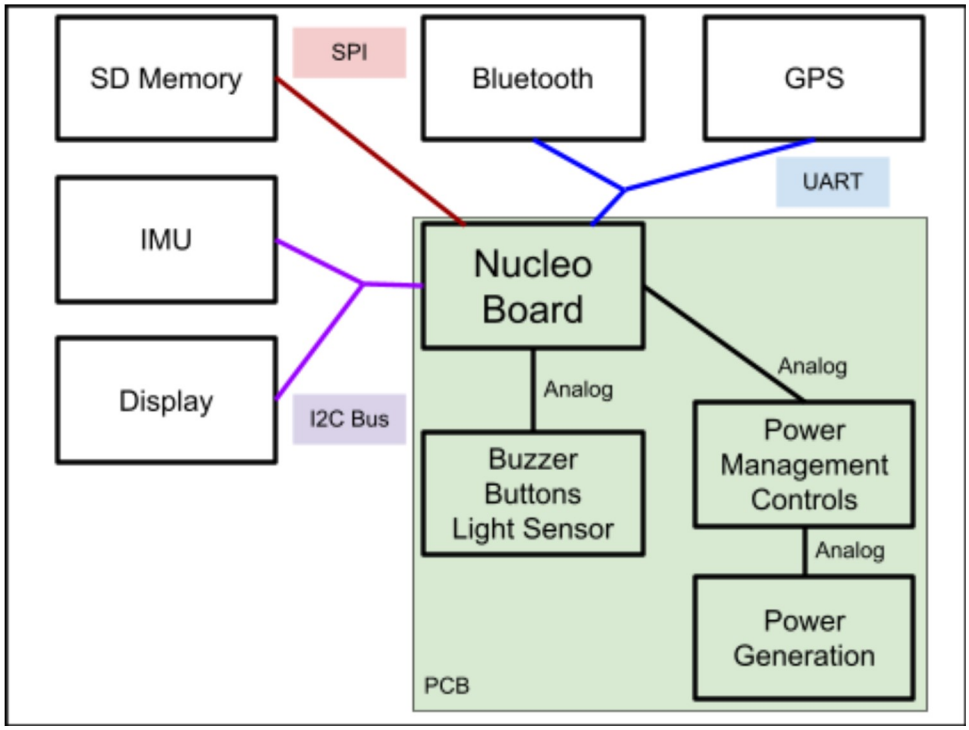

Figure 4: Components Diagram

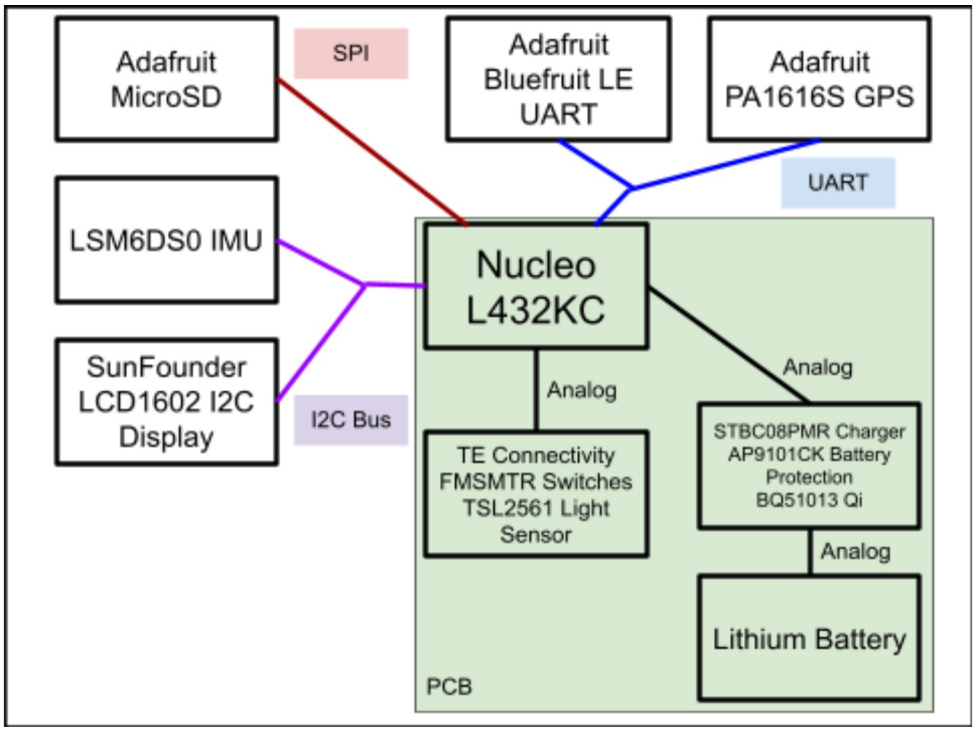


Figure 5: Finite State Machine for Display