Handheld Rocket Tracker
ECE 445 Design Document - Spring 2024

Project # 16

Benjamin Olaivar, Maxwell Kramer, Manas Tiwari

Professor: Arne Fliflet

TA: Sanjana Pingali

Contents

Introduction:

Problem:

For rocket platforms of any size, recovery is a major issue. With the amount of physical hardware costs typically involved with a rocket, recovery and reuse of these resources is an ever increasingly important task. Therefore, systems are needed to be able to recover a rocket. Two cases arise. One, when the rocket is fully intact and can be directly reused. Two, when the rocket suffers a catastrophic failure. Here, some system of tracking the major components is needed so engineering teams can retrieve the surviving physical components to analyze in order to pinpoint the error that occurred to cause the failure.

This isn't a new issue in amateur rocketry, and many solutions have already been developed to address this problem. Radio beacons, altimeters, and similar handheld devices have been created, however they all suffer from being clunky, unintuitive, or expensive. Affordable solutions such as radio beacons don't send out their exact location, and are tracked by following the strength of their signal [1], which only gives the general direction of the beacon. Altimeters send out their exact location, but are costly ($380+) and often require a laptop to receive their position [2], which is inconvenient to carry during a search. A few handheld trackers exist, however they are costly ($475+), difficult to reconfigure, and unintuitive [3].

Solution:

Our system aims to tackle several of the issues with current trackers on the market. We'll be implementing a 2-part tracking system consisting of a tracking beacon ("beacon"), which will be placed on the rocket, as well as a handheld tracking device ("tracker"). The beacon will transmit its GPS location to the handheld tracker. Once received, the tracker will compare its

current location with the beacon's transmission, and guide the user towards the beacon. This will be done by displaying a compass needle, which points in the direction of the rocket. Ideally, this solution will be more intuitive, more affordable, and more enjoyable to use.

High Level Requirements:

1. Successfully transmit positional and state data from the beacon to the handheld tracker, and handheld tracker should successfully transmit commands to the handheld beach. See outlined in the "Packet Breakdown" under Software Design.
2. Have the capability for the user to switch the frequency of both the beacon and the handheld tracker via user input on the handheld tracker device.
3. Accurately show the distance from the beacon within 5 meters, and point the user in the correct direction of the beacon within 2 degrees. This information should be shown via the screen in the User Interface, and behave similar to a compass, however pointing towards the beacon instead of pointing North.

$$Compass\ Direction\ =\ current\ heading$$

$$Angle\ =\ tan^{-1}\left(\frac{|\Delta Latitude|}{|\Delta Longitude|}\right)$$

2. Design:

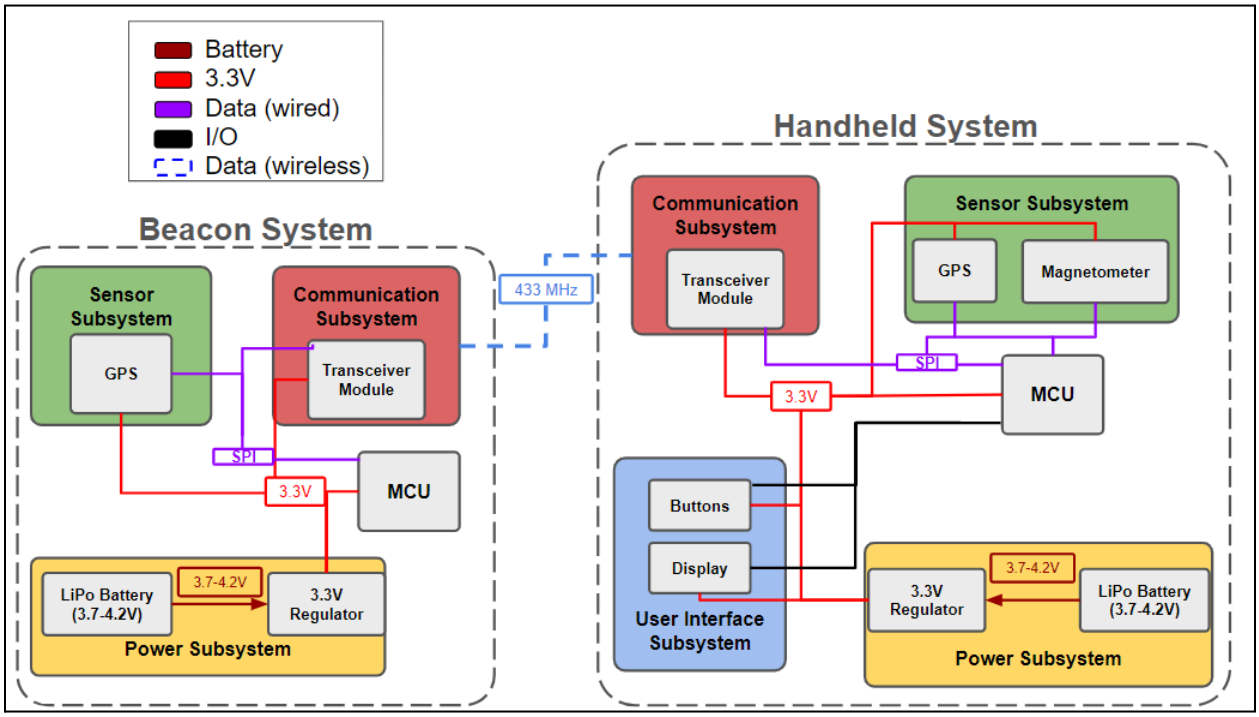

Figure 1: Block Diagram

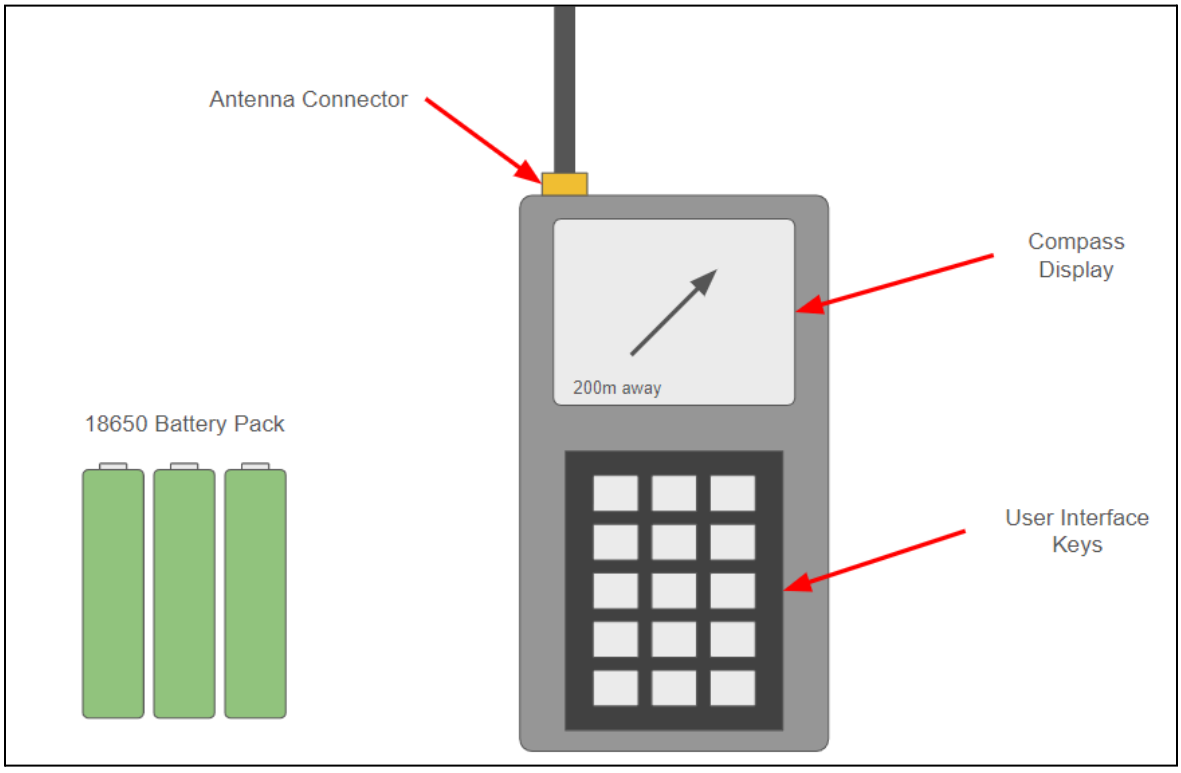

Figure 2: Handheld Tracker Concept Design

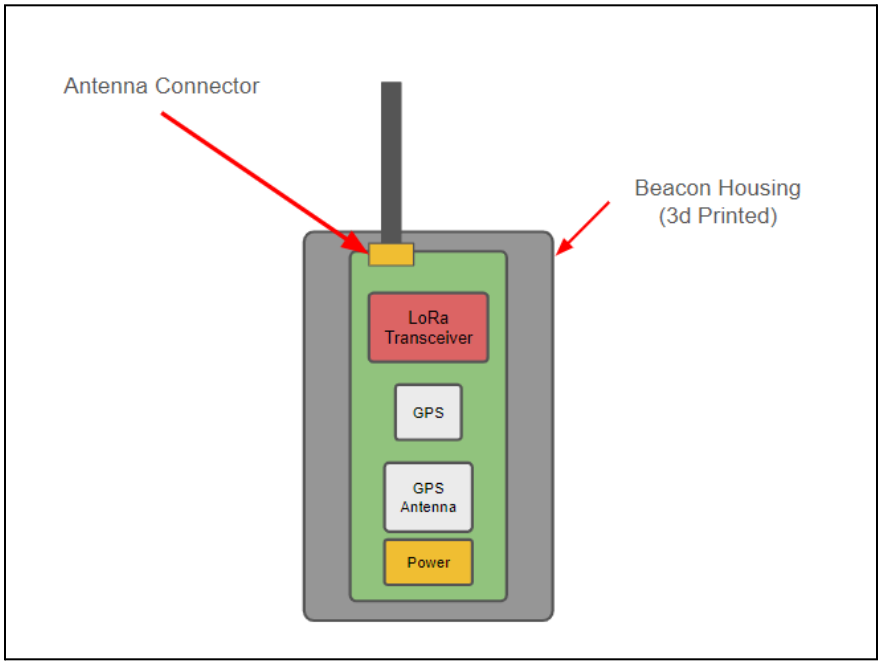

Figure 3: Beacon Concept Design

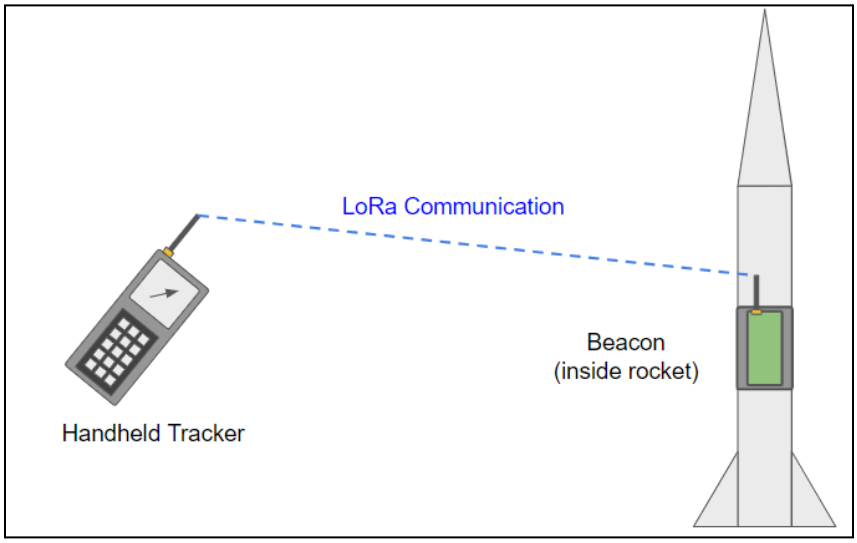

Figure 4: General Overview Diagram

Subsystems:

MCU:

The MCU is responsible for managing communication between the various subsystems. Both the beacon and handheld tracker have a copy of the same processor, however the code running on each device will be slightly different.

On the beacon, the MCU will be responsible for transmitting positional data and handling commands received from the handheld tracker. To do this, the MCU will retrieve the GPS coordinates from the Sensor subsystem via I2C, and send them to the Communication subsystem for transmission. Commands, such as “Frequency Change”, may be received from the handheld tracker, and are outlined in Table 1 below.

The handheld tracker will have an identical processor, however in addition to the tasks performed by the beacon, it will manage the User Interface subsystem (details outlined in the User Interface section). The handheld tracker will compare its own location & heading (from the Sensor subsystem via I2C) to the received beacon location (from the Communication subsystem via wireless transmission), and calculate the distance & direction from the user. Using this data, the MCU will tell the display (in the User Interface) which direction to point the arrow to guide the user to the beacon. See the behavior in Figures 5 and 6.

Table 1: MCU - Requirements and Verification

Requirements	Verification
● The MCU must successfully receive positional and directional heading data via I2C from the Sensor Subsystem.	● Ensure handheld tracker is powered on and in idle state. ● Go outside to a relatively clear area (few trees or tall buildings). ● Connect the MCU to a computer with a USB cable and open the Arduino IDE. ● Print the GPS coordinates and heading (“heading” being the degrees from north, going clockwise). Confirm the GPS coordinates read from the GPS match the real-world location.
● Upon user input, both the tracker and beacon must be able to change to the desired frequency (complying with LoRa spec of 433.05MHz - 434.79MHz).	● Ensure the beacon and handheld device are powered on and in idle state. ● Verify a communication link has been established between the devices (will appear on the handheld tracker display) ● Navigate to the “Frequency Change” command in the User Interface menu, and input the new desired frequency. ● Verify both devices have switched to the new frequency (handheld tracker will display the current frequency, as well as the successful communication link with the beacon). ● Under the hood (Part 1): The handheld tracker will transmit a data packet via the Communication subsystem, then switch to the new frequency. ● Under the hood (Part 2): Upon

Requirements	Verification
	receiving this data packet, the beacon will switch to the desired frequency, and send out a “handshake” message to confirm it has switched. ● Under the hood (Part 3): Once the “handshake” has been confirmed on both devices, the communication will be considered successful.
● The User Interface Subsystem must display accurate heading and distance to guide the user to the beacon. Accuracy must be within 10-15 meters (accuracy of GPS readings), and the arrow must point within 5 degrees of actual heading. See behavior in Figures 5 and 6	● Verification process can be found under the User Interface subsystem.

User Interface Subsystem:

The User Interface subsystem, only located on the handheld Tracker, is responsible for displaying relevant information, and receiving inputs from the user. The User Interface has 2 main components: the screen and the user input. The screen is the primary interface for the user, and must intuitively display the direction and distance from the beacon. This will behave similar to a compass, but point towards the beacon instead of pointing north. Additionally, the user should be able to view and modify the current frequency of the transmitters. The location and directional data will be received from the MCU, which will handle all the necessary calculations (see the MCU section for details). As the user moves around, the screen should update the distance and heading accordingly. See Figures 5 and 6 below.

The user input will be made of basic pushbuttons, which will be used to navigate a basic menu on the display ("show compass" or "frequency change"... menu still TBD). The user may change the frequency by entering the desired channel on the keys.

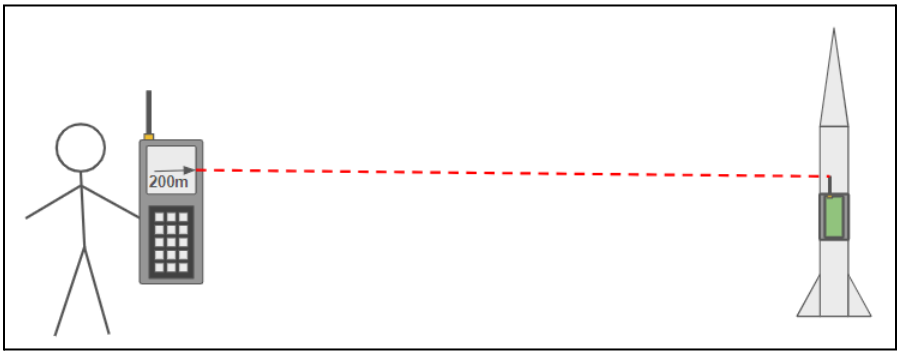

Figure 5: User is standing "left" of the beacon, 200 meters away. The display guides the user by showing the distance and direction they must travel to reach the beacon.

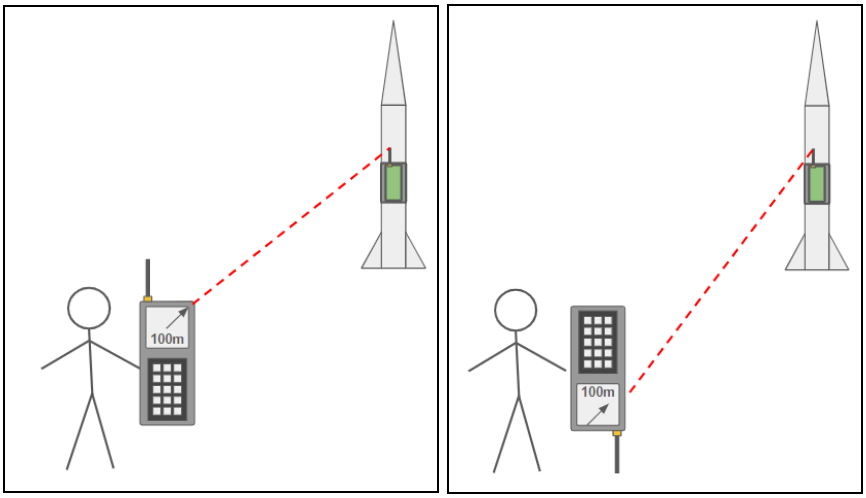

Figure 6: User moves closer, but travels “down”. The display updates, showing the new distance and direction to reach the beacon, including when the device is rotated.

Table 2: User Interface Subsystem - Requirements and Verification

Requirements	**Verification**
• The User Interface Subsystem must display accurate heading and distance to guide the user to the beacon. Accuracy must be within 10-15 meters (accuracy of GPS readings), and the arrow must point within 5 degrees of actual heading. See behavior in Figures 5 and 6	• Pick 2 arbitrary coordinates within 0.5 mile range. Designate one as the handheld tracker (point A), and the other as the radio beacon (point B). • While at point A, hold the beacon and handheld tracker next to each other. Ensure both the beacon and handheld tracker are powered and in idle state. • Ensure radio communication between devices (will be displayed on the screen), and validate the GPS coordinates of both devices. They should both roughly match the coordinates of point A. • Keep the handheld tracker at point A, and place the beacon at point B. • Use hand-calculations to find the expected distance & directional readings from the handheld tracker to the beacon. Compare these calculations to the output given on the

Requirements	Verification
	display. ● Repeat for 1-2 additional points, ensuring the compass on the handheld tracker points in the correct direction, and gives accurate distance measurements.
● When pressed, the input keys should generate interrupts and control the user-interface accordingly (details on user interface layout/features are still TBD).	● Push various keys on the keypad. Confirm that the device responds to inputs, such as entering frequencies and navigating the menu.

Sensor Subsystem:

The Sensor subsystem is responsible for gathering positional data from the sensor suite, and sending it to the MCU. Conveniently, we have found a sensor that has everything we need built into 1 unit: the SAM-M8Q. The SAM-M8Q has a builtin GPS module and GPS antenna for receiving coordinates, as well as a magnetometer to determine the heading of the user.

The beacon only requires the GPS coordinates, which will be sent to the MCU via I2C, and then transmitted to the handheld tracker over the communication subsystem.

Once the handheld tracker obtains the GPS coordinates of the beacon, it can compare against its own coordinates and heading and find the distance and direction of the beacon in reference to itself. To get the most accurate readings, the device must be held with the back facing the ground (like how someone would hold a compass). This ensures the most accurate reading from the magnetometer.

Table 3: Sensor Subsystem - Requirements and Verification

Requirements	Verification
● The sensor subsystem should be able to get accurate GPS coordinates within 5 meters, and heading accuracy within 2°.	● Pick an arbitrary location outdoors, ideally in an open area with few tall buildings and trees (similar to what would be experienced in the field). ● Obtain the coordinates of the location using map data. ● Ensure both the handheld tracker is powered and in idle state. For best heading results, hold the handheld tracker with the back parallel to the ground facing due North. ● Compare the received GPS coordinates with the expected value from map data. Ensure they roughly match (within 5 meters). ● Rotate about the point 360°, taking note roughly every 20°, ensuring the GPS is within 2° of the expected angle. ● Repeat with the beacon (results should match the handheld tracker).

Communication Subsystem:

This Communication subsystem is responsible for the interaction between the tracker and the beacon. The two devices will be using LoRa (Long Range) transmitters. The term LoRa itself stands for “Long Range Radio”. It is a wireless communication technology specifically designed to transmit data over long distances while utilizing minimal power consumption. The desired range of operation of our project is to cover a range of 0.5 miles to 5 miles, which fits within the range which LoRa operates in. The data transmitted between the two devices will be using the I2C bus of the MCU (see Transceiver communication for more details).

The communication would be allowed to take place over a predefined range of frequencies. The user will choose the desired frequency for communication via the handheld tracker. Once the frequency is chosen, both devices will confirm the chosen frequency and begin operation.

The tracker and the beacon will be equipped with a PCB LoRa 433 MHz antenna and a LoRa SMA rubber duck 433 MHz antenna respectively for their suitability to our needs (see Telemetry for more details).

Table 4: Communication Subsystem - Requirements and Verification

Requirements	Verification
● Antennas used should be capable of transmitting at 433 MHz.	● Mount the respective antennas onto the handheld tracker and beacon. ● Ensure that the antennas are upright, and not close to large metal objects or water bodies. ● Antenna feeder cable used should be as small as possible to reduce signal attenuation.

Requirements	Verification
	● Positioning and direction of the antennas play an important factor. A few different set ups will need to be tested to make sure no unexpected issues come up. ● Initial distance for testing can be 4 feet.
● Antennas should be capable of transmitting over a minimum range of 0.5 miles (or available space on UIUC campus quad). Maximum range of transmission would need to be obtained via field testing.	● Place the beacon at a location 0.5 miles from the handheld tracker. ● Make sure that signal and data transmitted by the beacon antenna is received, without any glaring issues, to the tracker. ● Repeat the above set up should be repeated with increments of 0.1 miles to the distance between the transmitter and beacon antenna. Once the transmitted signal strength goes below

The Power Delivery Subsystem:

The Power Delivery Subsystem is responsible for delivering power to all other subsystems on the boards. The subsystem should be able to regulate the variable input voltage (which will gradually fall) to a consistent 3.3V. For more information on the physical design of this system, refer to the Hardware Design section below.

Table 5: Power Delivery Subsystem - Requirements and Verification

Requirements	Verification
● The Power Delivery Subsystem must take a variable input (3.7V-4.2V) from the LiPo battery pack and consistently regulate down to 3.3V. This 3.3V must be sent out to all corresponding electronics, including everything in the Sensor, Communication, and User Interface subsystems.	● Connect the voltage regulator to a power supply. Set the power supply to the maximum LiPo battery charge of 4.2V. ● With a multimeter, check the output voltage. Connect the positive prong to the output voltage from the regulator, and the negative prong to ground. Ensure the regulator is outputting 3.3V. ● Repeat the first 2 steps, decreasing the input voltage in intervals of 0.1V until reaching the minimum battery voltage of 3.5V.

Hardware Design

Operating Voltage and Regulation

The current selection of components requires an input voltage of 3.3V to properly operate. To achieve this a buck converter has been selected to convert the input voltage from the battery source to the desired 3.3V.

All major components also require some minimum drawn amperage to operate. The SAM-M8Q GPS requires 29mA [6]. The Adafruit SHARP Memory Display requires 4 micro Amps [7]. The ATMega needs anywhere between 0.05mA to 0.40mA when operating at 0.1MHz to 1MHz respectively [8]. The LoRa module needs 95mA [9]. All other components either do not need a minimum current draw or will be negligible. This totals to 124.404 mA drawn by the circuit from the battery. The chosen buck converter, the Adafruit LM3671 3.3V Buck Converter, can handle load draws of up to 600 mA [10]. These values are drawn from the cited datasheets. In the case of the ATMega we assume maximum current draw for safety purposes as to make sure we can handle the largest possible minimum draw needed to function.

We will be using 18650 LiPo batteries for this project. This is chosen specifically since LiPo batteries can be recharged. The 18650 variety are ones we have easy access to, allowing us to cut the cost on acquiring more expensive batteries. The charge capacity is between 4.2-3.5V, both of which can be stepped down to 3.3V when connected to the chosen regulator system [24].

The Battery storage system will be a basic battery case with connected power and ground wires, such as 18650 Battery Holder Case [11]. This will allow basic and easy access to the battery for needed replacement or disconnection whenever needed.

Telemetry:

[12]

There are a number of antennas available in the market to choose from. In an outdoor environment, it is recommended to use a glass fiber LoRa antenna. Fiberglass antennas have good electrical insulation to ensure minimal disturbances while transmitting its signals, and good wave transmission [12].

The basic performance of an antenna is determined by a number of factors- spreading factor (SF), impedance, voltage standing wave ratio (VSWR), gain, frequency, bandwidth, and other physical parameters.

1. Spreading Factor (SF):

The spreading factor (SF) refers to the speed at which the signal frequency changes across the bandwidth of a channel. The SF allows for better resource management and optimization of energy consumption. The larger the SF factor, the more resistant it is to local noise, which improves the reliability and range of the antenna. Hence, we would prefer an antenna with a higher SF [18].

2. Bandwidth:

Bandwidth of an antenna refers to the frequency range in which the antenna operates. Higher bandwidth leads to higher data rates and more power efficiency. However, this corresponds to more congestion and less range. Most countries have restrictions on usage of frequency bands for communication for commercial purposes. These restrictions include specifying the allowed frequencies, maximum power, and maximum packet size. Our antenna would need to operate within the band of 433.050 MHz to 434.790 MHz. This bandwidth is used in the Amateur Radio Service, the Industrial/Business Radio Service, the Public Safety Radio Service, and for other household devices.

`The 433 MHz band experiences less attenuation due to atmospheric absorption, rain, and other environmental factors due to its lower frequency. This makes it suitable for long range communication while consuming low power. Hence, for our project, the 433 MHz bandwidth would be the most suitable for our antenna [20].

3. Impedance:

The input impedance of an antenna refers to the ratio of the input voltage to the input current at the feet side of the antenna. Most antennas on the current market operate on a 50 Ω impedance [12].

4. Gain:

The gain of an antenna is used to measure the ability of an antenna to send and receive signals in a particular direction. It describes the amount of power transmitted in a certain direction. Antennas with a higher gain are preferable [12].

5. Voltage Standing Wave Ratio (VSWR):

The voltage standing wave ratio (VSWR) refers to an operation on the coefficient of reflection that describes the power reflected from the antenna. The smaller the VSWR is, the more power efficient the antenna. The smallest possible VSWR is 1.0, which refers to no power being reflected from the antenna. We aimed to choose an antenna with a small VSWR and ended up with one which has its listed value as <= 2.0 [21] [22].

6. Mounting of Antenna:

Most antennas are available with various options of mounting such as magnetic, adhesive, clamp-bracket mounting etc. As a general rule, LoRa antennas should be placed as far from the ground as possible. Since the antennas we are using would be mounted on the handheld tracker and the beacon on the rocket, this factor would be taken into account already. The antennas we have chosen have PCB mounting for the handheld tracker and SMA connector mounting on the beacon [22].

Software Design

Transceiver Communication

The RFM96W transceiver on the handheld tracker will communicate with the RFM96W transceiver on the beacon. The RFM96W utilizes the LoRa (Long-Range) communication protocol, which separates packets into 5 sections: the Preamble, Header, Header Cyclic Redundancy Check (CRC), Payload, and Payload CRC, as seen in Figure 7 and the Packet Breakdown below. Packet data from the beacon will be primarily focused on the state of the device (coordinates, elevation, errors), while the handheld tracker will primarily be sending commands. We have confirmed the need for one command, Frequency Change, however the list of commands is easily expandable via software updates.

Frequency Change Command - Implementing the "handshake":

In special cases, such as a competition or launch event, the team may be required to tweak the frequency of the transceivers so as to not overlap with other team's communication systems. To do this, we are implementing a "Frequency Change" command, which will change the frequency of both the handheld tracker and the beacon by simply entering a new frequency on the handheld tracker (see the Frequency Change Steps below).

Figure 7: LoRa communication packet layout [15]

PHY Frame	Preamble	Header	Header CRC	Payload	Payload CRC
Size	Min. 4.25 symbols	2 bytes	2 bytes	Max. 255 bytes	2 bytes

Packet Breakdown [13] [14]:

Preamble: The preamble consists of 12.25 “symbols”, aka radio “chirps”, which identify the transmitter and mark the beginning of the transmission.

Header (2 Bytes): The first byte defining the payload size (11-255 bytes), and the second byte defining information regarding the CRC (number of bits per parity bit). CRC config is directly related to payload size.

Header CRC (2 Bytes): The error-detecting code for errors in the header. Each parity bit represents the *sum of X bits modulus 2* in the header, with X being defined in the 2nd byte of the header. If there are an even number of ‘1’s, the parity bit is 0, otherwise it is 1. If there is a discrepancy between the number of ‘1’s and the parity bit, there is an error in the transmission. Error checking is handled internally by the RFM96W. If an error is detected, a “resend" request is sent to the transmitting device.

Payload (11-255 bytes): The actual data sent. We’ll be transmitting 2 basic data structures outlined below:

- beacon_transmission_struct (13 bytes), seen in Figure 8:

- 8-bit error mask, with each bit corresponding to an error. 0 = no error, 1 = error.
  - Bit (0) = unsuccessful GPS lock
  - Bit (1) = unresponsive GPS communication (error with I2C bus/sensor not responding)
  - *Bits (2-7) = TBD as new bugs are found*
- Longitudinal coordinates of the beacon (4 bytes)
- Latitudinal coordinates of the beacon (4 bytes).
- Elevation of the beacon (4 bytes)

Figure 8: beacon_transmission_struct

```
struct beacon_transmission_struct {
  uint8_t error_mask;
  float coords_longitude
  float coords_latitude;
  float elevation;
};
```

- handheld_transmission_struct (8 bytes), seen in Figure 9:
  - Commands will be assigned numbers (ex: “Frequency Change” may be #1), and will have corresponding data (4 bytes)
  - The corresponding data (ex: the new frequency “433.12 MHz”) (4 bytes)
  - *Structure subject to change as more commands are added*

Figure 9: beacon_transmission_struct

```
struct handheld_transmission_struct {
  int command_number;
  float command_data;
};
```

Payload CRC (2 bytes): Similar to the header CRC, this section contains error-finding "parity" bits. Each parity bit represents the s*um of X bits modulus 2* in the payload, with X being defined in the 2nd byte of the header. This also marks the end of the transmission. Error checking is handled internally by the RFM96W. If an error is detected, a "resend" request is sent to the transmitting device.

Frequency Change Steps:

1. *(User):* While holding the handheld tracker, navigate to the "Frequency Change" section of the menu. Ensure the handheld tracker and the beacon have a secure link.

2. *(User):* Enter the new desired frequency.

3. *(Software-Handheld Tracker):* The handheld tracker changes the command_number (Figure 9) to 1, representing the "Frequency Change" command. It also changes the command_data (Figure 9) to the user-entered frequency.

4. *(Software-Handheld Tracker):* The handheld tracker transmits the 'handheld_transmission_struct' to the beacon. Once the beacon has received the message, the handheld tracker will switch to the new frequency.

5. *(Software-Beacon):* Upon receiving the "Frequency Change" command, the beacon will switch to the new frequency and transmit a blank packet, initiating the "handshake".

6. *(Software-Handheld Tracker):* The handheld tracker, having received the “handshake” packet on the new frequency, will consider the frequency change “successful” and continue transmitting on the new frequency.

3. Cost and Schedule:

Cost Analysis

The total cost for parts before shipping is $234.53, as seen in Table 6 below. 5% shipping cost + 10% sales tax adds another $269.70. We can assume a salary of roughly $50/hr × 2.5hr ×60 hours to complete = $7,500 per team member. Accounting for all 3 members, that brings a total labor cost of $22,500. Adding this all together, we get cost before shipping + shipping cost + sales tax + labor per member × 3 members = $22,769.70.

Table 6: Bill of Materials

Item	Price	Quantity
SAM-M8Q GPS breakout	$43.00	2
Adafruit SHARP Memory Display Breakout	$24.95	1
Processor	$1.63	2
AS0805KKX7R0BB104 0805 104 pF Capacitor	$0.32	10
RC0805FR-1310KL 10 kOhm resistor	$0.10	2
ABLS-16.000MHZ-B2-T 16MHz crystal oscillator	$0.27	2
Adafruit RFM96W LoRa Radio Transceiver	$20.00	2
SMA ANT connector	$3.74	2
XLMDK12D LED(s)	$0.34	5
TS04-66-55-BK-160-SMT Pushbutton	$0.18	20
18650 LiPo batteries	$6.62	5
Battery holder	$0.60	1
LM3671 3.3V Buck Converter	$4.95	2
ANT-433-CW-RCS beacon Antenna	$10.00	1
FLEXI-SMA90-433 Handheld Antenna	$10.00	1
Boards	$0.00	1
Housing	$0.00	1

Schedule

Table 7: Schedule

Week	Task
February 26th - March 4th	Order parts for prototyping
	Start prototyping with existing components
	Research Transceiver communication
	Start PCB design
March 4th - March 11th	Begin 3D print designing
	Successfully establish transceiver communication
	Finish 1st iteration PCB design
	PCB Order
March 11th - March 18th	Print first versions of 3D printed case prototypes
	Finish the baseline transceiver communication code
	Finish baseline user interface menu
	Range testing with wire antennas
March 18th - March 25th	Finalize 3D prints
	Prototype user interface menu, controlling with Arduino Uno
	Revisions to PCB
	Revisions to User interface software
	PCB Order
March 25st - April 1st	Revisions to PCB
	Revisions to User Interface software
	PCB Order
	Range testing with 1st ordered antennas
April 1st - April 8th	Revisions to 3D design
	PCB Order
	Finalize 3D prints
	Order new antennas (if necessary)
April 8th - April 15th	Range testing with new antennas (if necessary)
	Fix existing bugs
April 15th - April 21st	Finalize Assembly
	Fix existing bugs
April 22nd	Final Demo

Tolerance Analysis:

A main area of risk within this project is the proper function of the RF components. Specifically the proper communication of the transmitters/GPS. The range and accuracy are paramount. This fact will be discussed further under ethics contexts. The Adafruit transceiver datasheet currently lists 12 miles and 1 mile of range based on attached antenna and wire [4]. This component should therefore give us the range of signal transmission we need to establish connections for locating the rocket or debris. Beyond the transceiver we also have a need for reliable GPS and antennas. For the SAM-M8Q Sparkfun GPS we are currently considering 2.5m Horizontal Accuracy, with 26s cold first fix time [6]. These datasheet stats show that the system should be able to locate the rocket or debris fairly accurately with short connection time when first activated. These times and ranges will be tested extensively when the system is built to verify the datasheet claims.

These datasheet values, if accurate, show that we should be able to build an accurate and reliable tracking system. The GPS's built in antenna and magnetometer give it extra reliability as student work connecting these components will not risk connection breakdown over wire or circuit board.

Another main area of risk is the voltage regulator. As we require all major components to operate at 3.3V, failure of the regulator, in this case our chosen buck converter, will result in a total failure of the overall system. Thus, reliability is paramount. Power dissipation of the converter can first be found by the following equation.

$$P_d = i_{out} * (V_{in} - V_{out})$$

Here our output current is equal to roughly 125mA at minimum and up to 600 mA at maximum based on converter datasheet specs [10]. Our desired output voltage is 3.3V. Our input

voltage could be between 3.5V to 4.2V. This results in a minimum power dissipation of 0.025W with min voltage difference and min current draw, and a maximum of 0.54W with max voltage difference and max current draw.

$$\mathrm{T_j} = \mathrm{i_{out}}(\mathrm{v_{in}} - \mathrm{v_{out}})(\Theta_{\mathrm{jc}} + \Theta_{\mathrm{ca}}) + \mathrm{T_a}$$

Furthermore, we can use the above equation to estimate junction temperature. From the buck converter datasheet, junction to ambient thermal resistance is listed as 130 Celsius/Watt [10].

$$\theta_{jc} + \theta_{ca} = 130$$

At 60 celsius ambient temperature, the junction temperature would rise to 63.25 Celsius at minimum and 130.2 Celsius at maximum temperature.

$$63.25C = 0.0025W * 130C/W + 60C$$
$$130.2C = 0.54W * 130C/W + 60C$$

We should not reach much more operating current than what is minimally required. Estimating a twice than needed current draw of roughly 250mA, we get power dissipation of at most 0.225W with 4.2 to 3.3 volt drop. Using this we find we have a junction temperature of 89.25 Celsius at 60 Celcius ambient. This represents a temperature rise of only 29.25C in an extreme environment given what we can assume to be an above average current draw.

$$89.25C = 0.225W * 130C/W + 60C$$

The datasheet also states the converter is 500 mW draw at ambient 60 celsius. This is a temperature near the high end of typical environments. Given this information, we can see that the buck converter can handle most high ambient heat environments as well as significant current draw from the actual board. It is not, however, impervious. Significant current draw or ambient

heat can cause damage to the buck convertor. Thus design choices within the rest of the board must be monitored for minimizing current draw when possible to lighten power dissipation on the buck converter.

Ethics and Safety:

Our project aims to improve on the affordability and usability of current trackers on the market. Our project operates on 433 MHz, which is used by many household devices. Hence, by testing our project in open fields, we wish to minimize the amount of interference caused, which falls under IEEE's Code of Ethics Section 1.1 by aiming to " hold paramount safety, health, welfare of the public" [16].

Furthermore, other ethical and safety issues with this system come primarily with the tracking portion of our project itself. It could be potentially used for malicious purposes by certain people. We do not advocate for the misuse of our project's tracking capabilities. By ensuring that our tracker requires a line of sight between the tracker and the beacon, along with a minimal mile radius requirement to operate the software, we aim to minimize the potential malpractices that our project can be used for. This falls under IEEE's Code of Ethics Section 1.1 by aiming "to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment" [16].

References

[1] I. Apogee Components, “Rocket locators : Apogee rockets, model rocketry excitement starts here,” Model Rockets & How-To Rocketry Information, https://www.apogeerockets.com/Electronics_Payloads/Rocket_Locators (accessed Feb. 8, 2024).

[2] I. Apogee Components, “Rocket locators : Apogee rockets, model rocketry excitement starts here,” Model Rockets & How-To Rocketry Information, https://www.apogeerockets.com/Electronics_Payloads/Rocket_Locators (accessed Feb. 8, 2024).

[3] I. Apogee Components, “Rocket locators : Apogee rockets, model rocketry excitement starts here,” Model Rockets & How-To Rocketry Information, https://www.apogeerockets.com/Electronics_Payloads/Rocket_Locators (accessed Feb. 8, 2024).

[4] A. Industries, “Adafruit RFM96W lora radio transceiver breakout - 433 mhz,” adafruit industries blog RSS, https://www.adafruit.com/product/3073 (accessed Feb. 8, 2024).

[5] Ti, https://www.ti.com/lit/ds/symlink/lm350-n.pdf?HQS=dis-dk-null-digikeymode-dsf-pf-null-wwe&ts=1707432444534&ref_url=https%3A%2F%2Fwww.ti.com%2Fgeneral%2Fdocs%2Fsuppproductinfo.tsp%3FdistId%3D10%26gotoUrl%3`Dhttps%3A%2F%2Fwww.ti.com%2Flit%2Fgpn%2Flm350-n (accessed Feb. 9, 2024).

[6] M. #167436, Greggler, M. #873985, and M. #985133, “SparkFun GPS breakout - chip antenna, Sam-M8Q (Qwiic),” GPS-15210 - SparkFun Electronics, https://www.sparkfun.com/products/15210 (accessed Feb. 8, 2024).

[7] “RECORDS OF REVISION REVISED No,” 2014. Accessed: Feb. 22, 2024. [Online]. Available: https://cdn-shop.adafruit.com/product-files/3502/Data+sheet.pdf

[8] “ATmega328PB AVR® Microcontroller with Core Independent Peripherals and PicoPower® Technology.” Available: https://www.mouser.com/datasheet/2/268/40001906C-3323958.pdf

[9] “SX1231 SX1231 Transceiver Low Power Integrated UHF Transceiver,” 2013. Accessed: Feb. 22, 2024. [Online]. Available: https://cdn-shop.adafruit.com/product-files/3076/sx1231.pdf

[10] “LM3671/-Q1 2-MHz, 600-mA Step-Down DC-DC Converter Datasheet.” https://cdn-shop.adafruit.com/product-files/2745/P2745_Datasheet.pdf (accessed Feb. 20, 2024).

[11] “8650 Battery Holder Case -2 Slot with Switch.” Accessed: Feb. 22, 2024. [Online]. Available: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/338/114090053_Web.pdf Accessed: Feb. 17, 2024. [Online]

[12] “How to Choose the Best Antenna for Lora?,” *www.linkedin.com*. https://www.linkedin.com/pulse/how-choose-best-antenna-lora-ct-rf-antennas-inc/ (accessed Feb. 23, 2024).

[13] The Things Network, “Lora Physical Layer Packet Format,” The Things Network, https://www.thethingsnetwork.org/docs/lorawan/lora-phy-format/ (accessed Feb. 22, 2024).

[14] “What are Lora® and Lorawan®?,” LoRa Developer Portal, https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/#:~:text=The%20name%2C%20LoRa%2C%20is%20a,areas%20(line%20of%20sight) (accessed Feb. 22, 2024).

[15] “Maximum packet size,” Maximum Packet Size - an overview | ScienceDirect Topics, https://www.sciencedirect.com/topics/computer-science/maximum-packet-size#:~:text=4.2.&text=LoRa%20offers%20a%20maximum%20packet,and%20a%20quarter%20of%20up chirp (accessed Feb. 22, 2024).

[16] IEEE, “IEEE Code of Ethics,” *ieee.org*, Jun. 2020. https://www.ieee.org/about/corporate/governance/p7-8.html (accessed: Feb. 22, 2024)

[17] “Spreading Factor (SF) | FAQ | IoT Journey,” *iotjourney.orange.com*. https://iotjourney.orange.com/en/support/faq/spreading-factor-(sf) (accessed Feb. 22, 2024) [Online]

[18]T. Tulka, “LoRa Spreading Factor Explained by Tomas Tulka,” *blog.ttulka.com*. https://blog.ttulka.com/lora-spreading-factor-explained/ (accessed Feb. 22, 2024) [Online]

[19] “Antenna-Theory.com - Bandwidth,” *www.antenna-theory.com*. https://www.antenna-theory.com/basics/bandwidth.php Accessed: Feb. 22, 2024 [Online]

[20] “433MHz Antennas: 412-440MHz: UHF VHF,” *www.data-alliance.net*. https://www.data-alliance.net/433mhz-antennas/ (accessed Feb. 23, 2024) [Online]

[21] “VSWR,” *antenna-theory.com*. https://antenna-theory.com/definitions/vswr.php Accessed: Feb. 22, 2024 [Online]

[22] “Lora Antenna 433 MHz SMA Rubber Duck Antenna Manufacturer | C&T RF Antennas Inc | Antenna Manufacturer.” https://lcantennas.com/lora-antenna-433/ (accessed Feb. 23, 2024). [Online]

[23] ctrfantennas, “How To Choose The Best Antenna For Lora? | C&T RF Antennas Inc | Antenna Manufacturer,” Dec. 25, 2021. https://lcantennas.com/how-to-choose-the-best-antenna-for-lora/ Accessed: Feb. 22, 2024 [Online]

[24] “Lithium-ion Battery DATA SHEET Battery Model : LIR18650 2600mAh.” Available: https://www.ineltro.ch/media/downloads/SAAItem/45/45958/36e3e7f3-2049-4adb-a2a7-79c654d92915.pdf