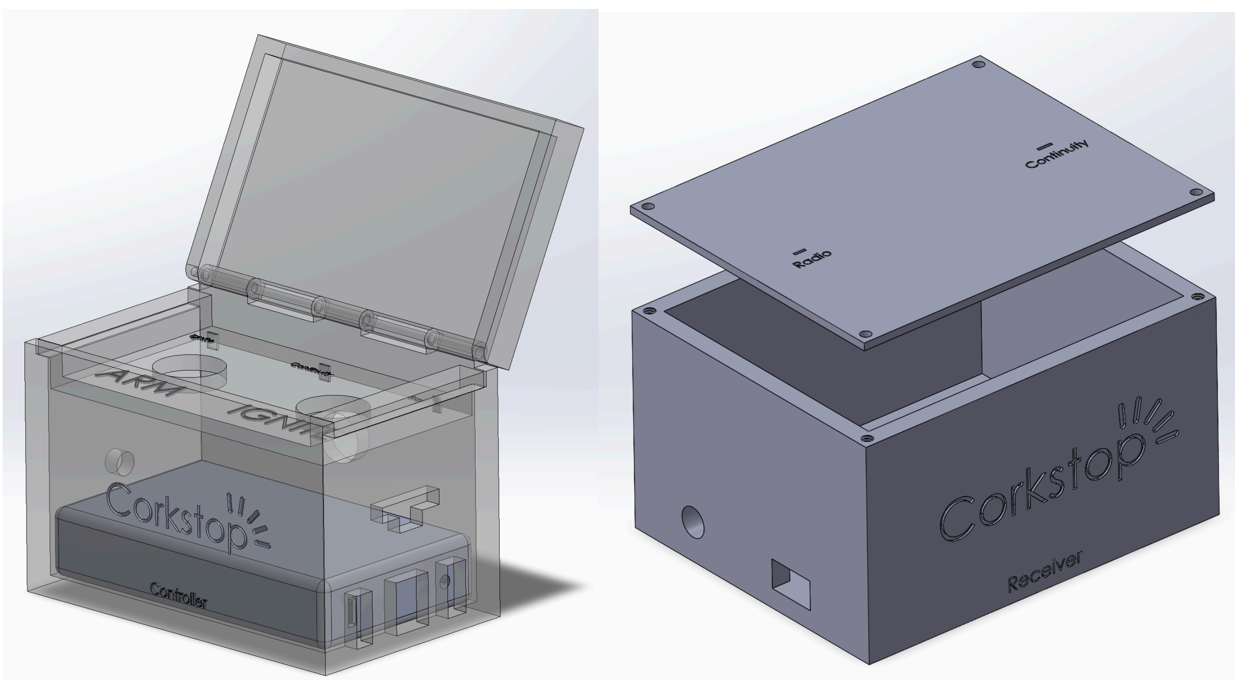
# Corkstop

Remote Ignition System

Corkstop is a single channel ignition system that operates alongside an e-match or igniter to remotely launch a rocket or activate a stationary rocket motor.

This project was made for use on the [UMN rocket team] as an SRAD (student research and developed) remote ignition system for test rockets, static fires, and launches.

Corkstop consists of a controller box and receiver box, each equipped with a microcontroller, radio, and portable power source.

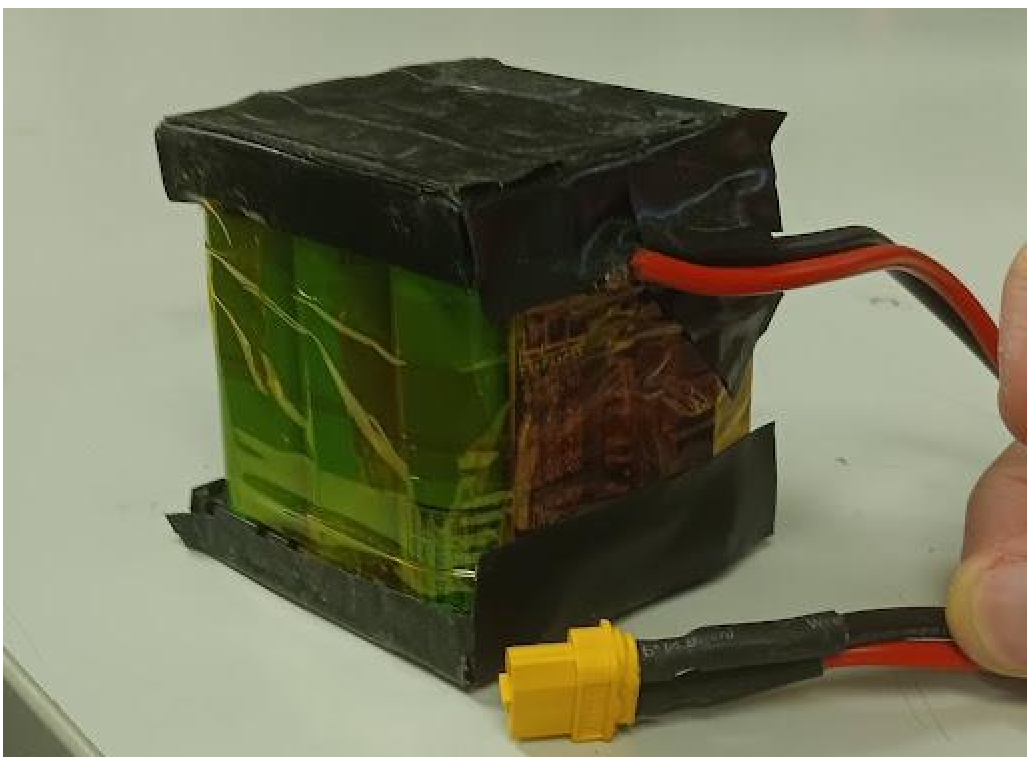



All the code and CAD can be found here.

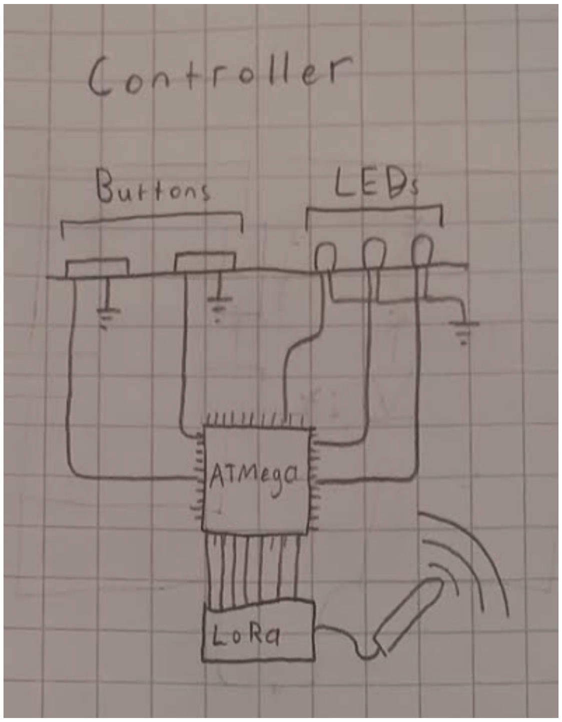
## Design and Implementation

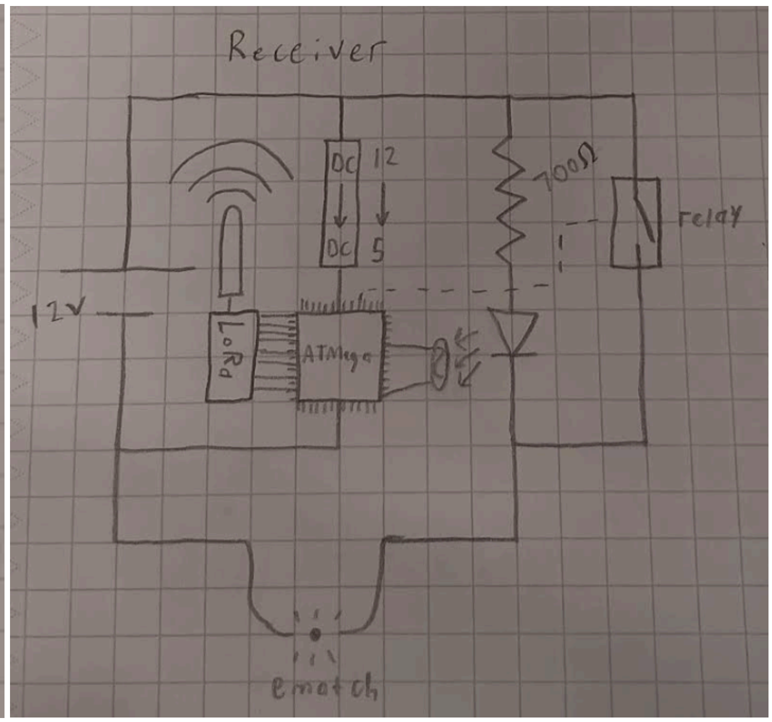
Here is the full BOM for this project:
1x Portable power source

1x AVR-BLE board
1x ItsyBitsy 5v board
2x LoRa breakout
2x 433MHz antenna
1x XT60 Splitter
2x XT60 connector
1x DC Buzzer
1x DC-DC converter
1x 5v Relay
1x 700Ω resistor
2x Arcade button
2x Alligator Clips
1x Photoresistor
1x Blue LED
2x Cyan LED
3x Multicolour LED
1x 10kΩ resistor
∞x E-matches
1x UFC battery (12.7v, pictured below)



## Circuit:




The basic idea is that a large amount of current (> 300mA) will cause the ematch to ignite. That current is supplied by the UFC battery on the receiver’s end. When the relay is triggered, the battery is shorted, causing a large amount of current to flow through the ematch. While the relay is not triggered, there is a small amount of current (about 15 milliamps) flowing through the ematch. This current is not enough to blow the ematch, but it is enough to light an LED (pictured directly below the 700Ω resistor). The light from the LED is detected by a photoresistor, and the information is recorded by the ATMega on the receiver.

If the ematch is connected improperly, or not connected at all, then the LED is not glowing, as the circuit is not complete. This is called **continuity**. When the circuit is closed it is said to “have continuity”, when it is open it “does not have continuity”. Both the receiver and controller boxes have a continuity status light, which is green to indicate continuity, and red to indicate loss of continuity. The receiver box uses the photoresistor along with its ADC to gather this continuity information. It then sends that information to the controller box over the LoRa radio.

## Radio:

Both boxes use an [SX1276](#) LoRa radio module. These radios interface over SPI and make use of an interrupt pin for incoming messages. The Corkstop™ custom radio protocol works via request response. The controller box, if on, periodically sends the ASCII string “cork” once a second. The receiver box responds to this by sending either “stop” or “stal”. “stop” indicates that it has continuity on the circuit, and “stal” indicates no continuity. When either radio receives a message, it will turn their cyan radio LED on with a timer to expire the LED in half a second. This means that the cyan radio LED should be blinking every half second if both radios are connected. If either radio disconnects both radio lights remain off.

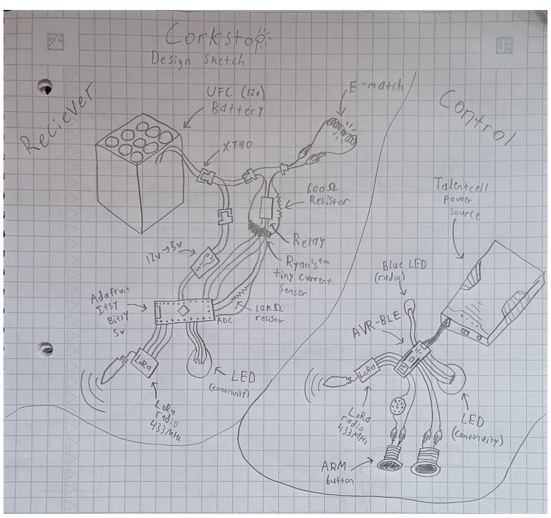
When the controller wants to send the ignition message, it sends the ASCII string “IGNITE”. The receiver will, after verifying continuity, trigger the relay to short the circuit. If continuity is not met when the ignition signal is received, the receiver will not trigger the relay and will send back “cant” in response. Otherwise, the relay will be triggered for 0.1 seconds and the receiver will send back “done” in response. The closing of the relay should ignite the ematch.

No CRC or checksums are used in the Corkstop™ custom radio protocol. Any malicious actor could send the ASCII strings over 433MHz frequency to attempt to trigger ignition. This also means that multiple Corkstop™ controller -> receiver pairs are not currently possible.

Additionally, libraries [RadioHead ]and [AVR_LoRa] were used in this project. Thanks to [Paul Stoffregen], [Kurt E], and [kpierzynski].

## Challenges:

This device is relatively simple, the main challenge was limited time. The initial design concept was created a month before the final product was complete, which is impressive if not a little irresponsible for a device that has a number of safety considerations.



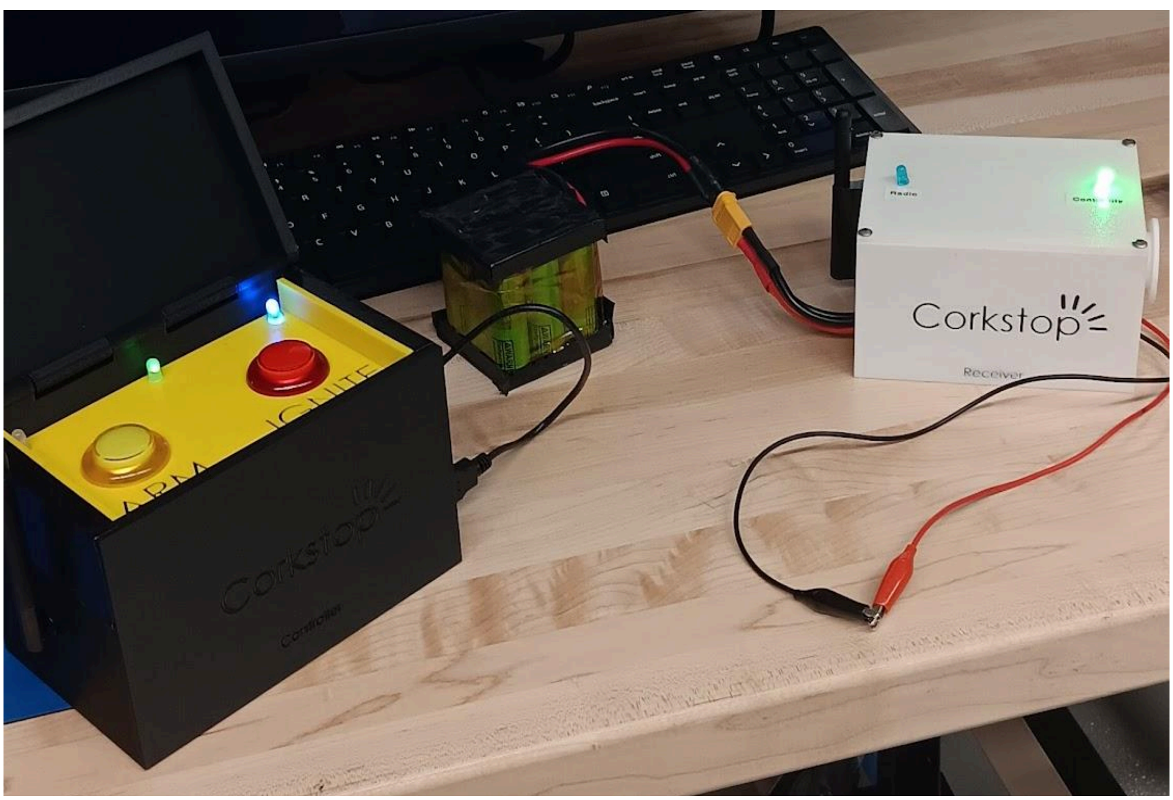

The only design difficulty was figuring out how to integrate the continuity sensor. Current sensors exist, but most of them are not built for these small amounts of currents (10s of milliamps). The LED taped to a photoresistor is certainly unique, but non-standard and potentially dangerous. If the LED burns out then the device will report no continuity even when there is (which is the most dangerous case). I tested the merits of the idea with an arduino and a breadboard three weeks before the project was due, and it seemed like it worked (in ideal environmental conditions that is). Certainly some more thought should have been put into it.

There were considerable manufacturing challenges, mostly due to a fairly large number of custom parts. I soldered XT60 connectors to the DC -> DC converter, as well as the resistor, current sensing LED, and alligator clips (with the heads cut off). The relay required custom wire with a molex crimp head, and that wire also had to be soldered in with the rest. I put some sleeves over the wire and heat shrunk them together, but this process was fairly difficult, especially at a three-way junction of relay, alligator clip, and LED which I had to redo twice (once when I put the LED on backwards, and again when the LED burned out under mysterious circumstances). I also needed more pins than the AVR-BLE could provide, so I ended up hijacking a few of the BLE pins (UART tx to BLE, as well as reset pin) by soldering wires onto the pads directly. That process was harder than I anticipated (and I totally destroyed the BLE in the process).

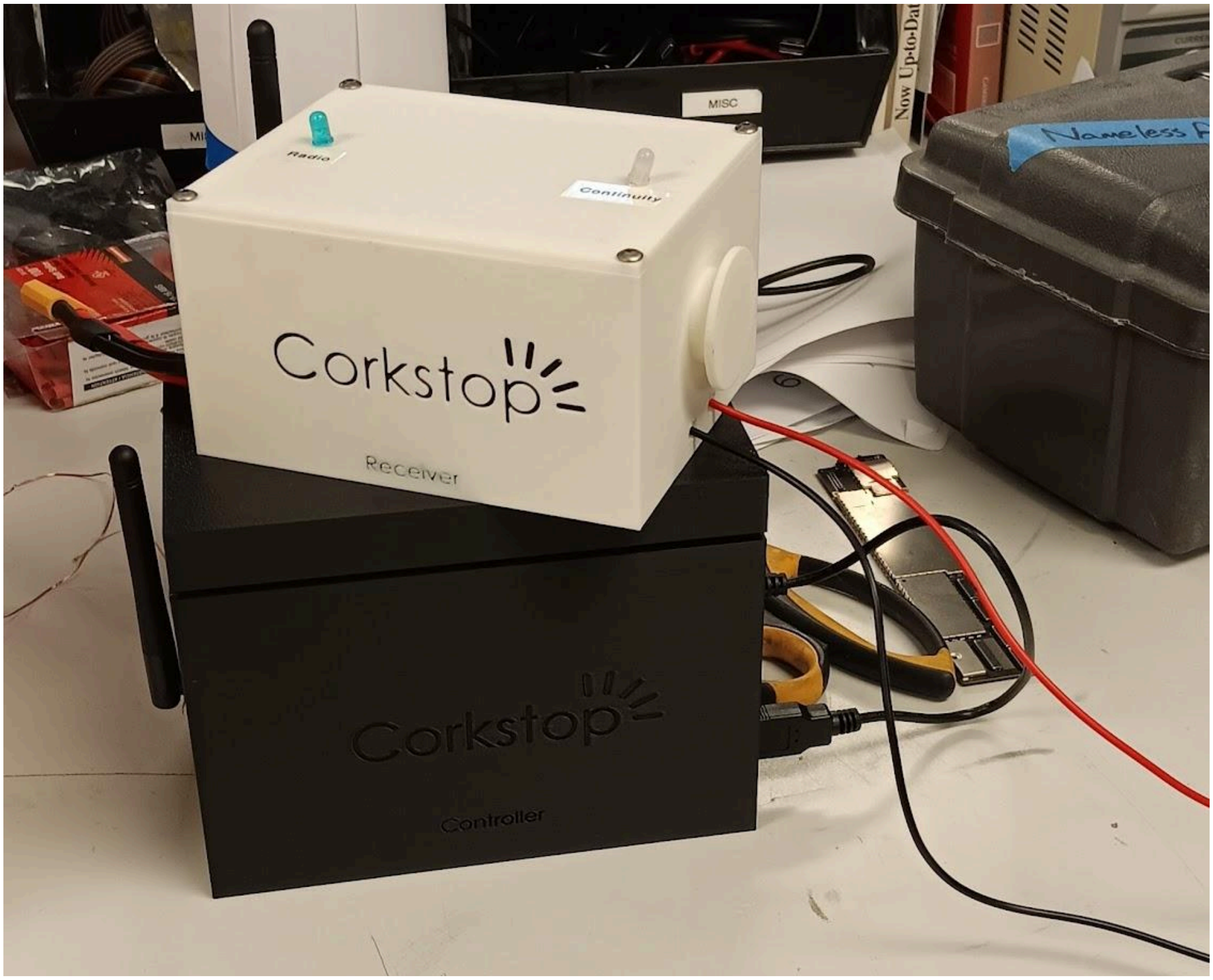
The other manufacturing challenge was CAD and 3D printing. Initial designs were finished in CAD about a week before the project was due, and I spent most of thanksgiving break printing them. Only one failed print, but some overlooked design quirks (buzzer hole was not big enough, screw taps did not work, caliper revealed that many of my holes were too small, luckily was caught before printing).

Finally, the assembly process was not difficult but was frustrating. I ended up stuffing all the wires in the box and hoping none of them disconnected. I also decided later to solder an extra antenna to the radio modules to increase range. This was very difficult to do after assembly had been completed.

# Final Product




In total, despite the rushed process, the device was working with all of its features on demo day. There were two tests prior to the due date, one with the electronics pre-assembly and another after the project had been initially completed. Both tests successfully ignited the ematch and every feature worked. Two more ematches were ignited on demo day. The current product has a 100% success rate so far.



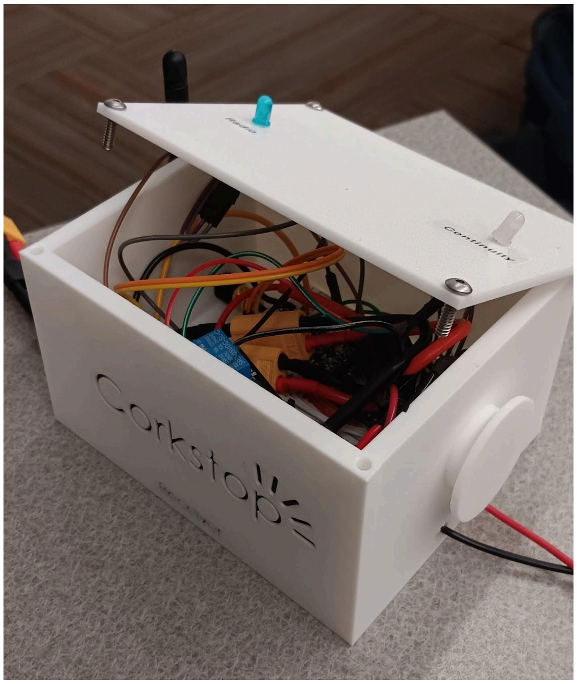

Every initial project goal was hit except for the range. The last range test gave poor results (~100ft), but I have added an extra antenna since then.

Overall, I am not totally satisfied by the quality and fragility of the project. I would’ve preferred it to be a lot more robust, and with more consideration for safety.

# Reflection

There are two main improvements that I would like to make to this device.

The first is radio and range. Upon testing the range of the device, it was only able to operate at around 100ft, which is not as far as the LoRa modules should be capable of. With more time and more knowledge of RF and antennas this could be improved.




The second is internal wiring. It is a mess inside the boxes and it is very likely that a wire can come unplugged just by shaking the devices minimally. Ideally a whole PCB could be designed, but a solder-in breadboard would also probably suffice, along with better cable management. These boxes also probably would not fair well in unideal weather conditions. The tape on the LEDs would fail in the New Mexico Deserts, none of the boxes are waterproof, and the radios need direct line of sight.