Vehicle to Vehicle Communication

Group 37

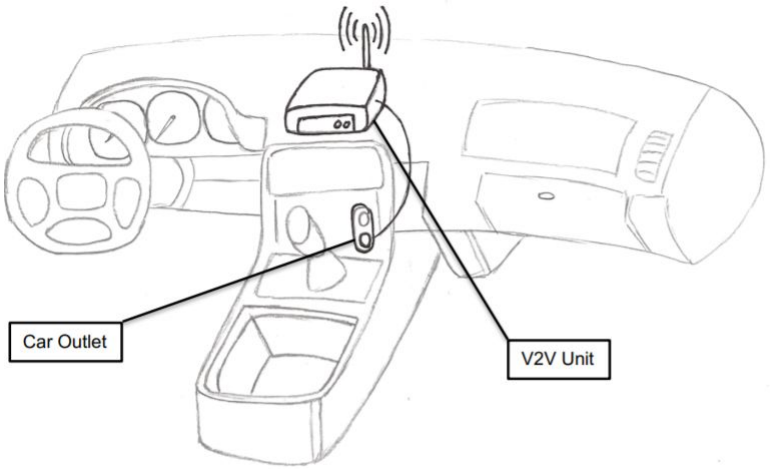
Alejandro Gonzalez, Harsh Harpalani, Tiger Chung

Problem Statement

- Smart vehicles rely on sensors to detect nearby objects and information around them
- Continuous reliability and certainty in interpretation of sensor data becomes imperative in making life or death decisions
- Limited scope of decision making since sensors have very short range

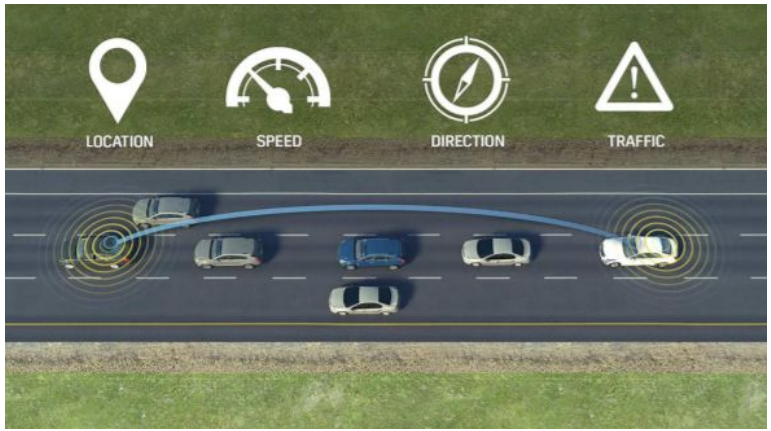
Our Solution: V2V

- Create a device that communicates with other identical devices
- Transmit/Receive relevant information such as GPS location, heading, speed, acceleration
- Device records data of nearby vehicles and feeds information into a smart system
- “Guesswork” taken out of data interpretation and vehicles will be capable of making much more complex decisions

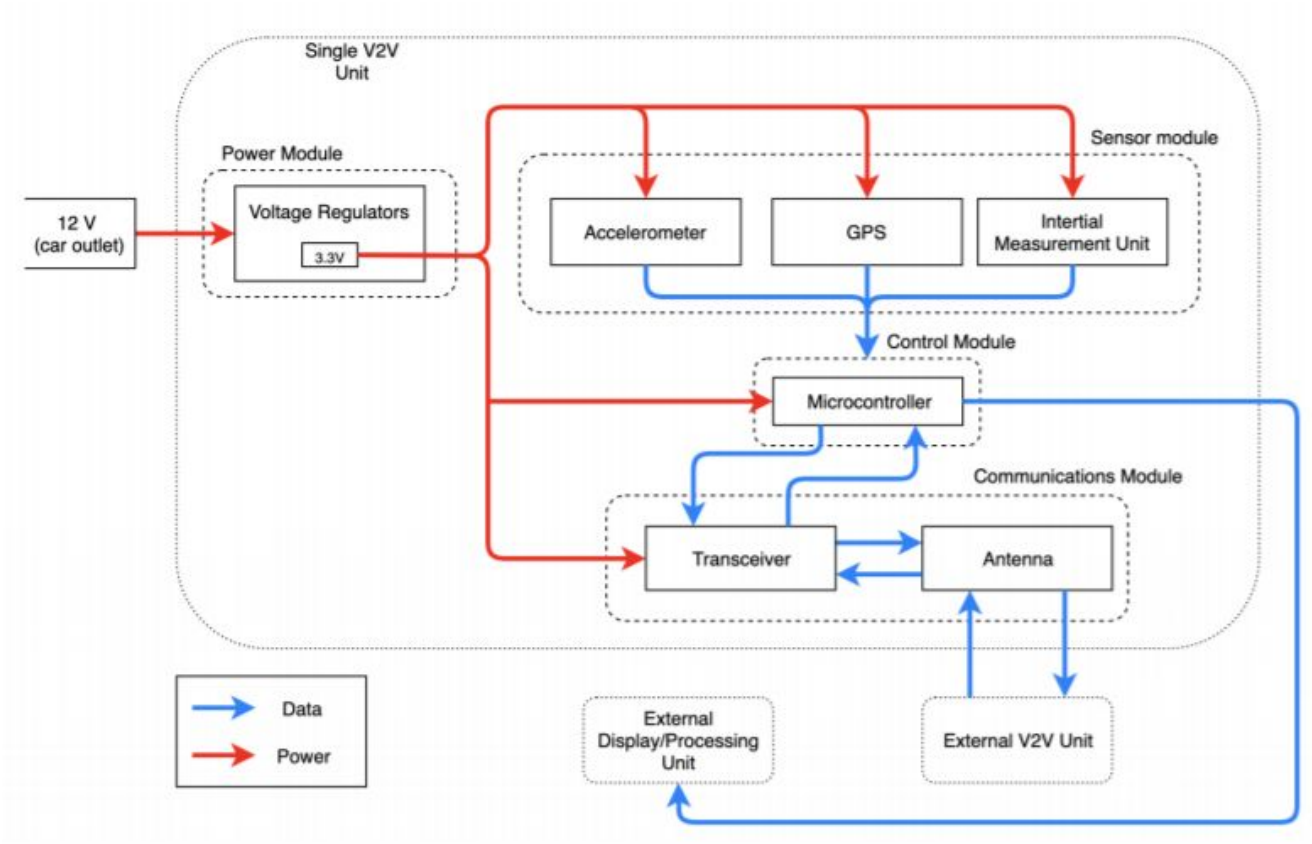


High Level Requirements

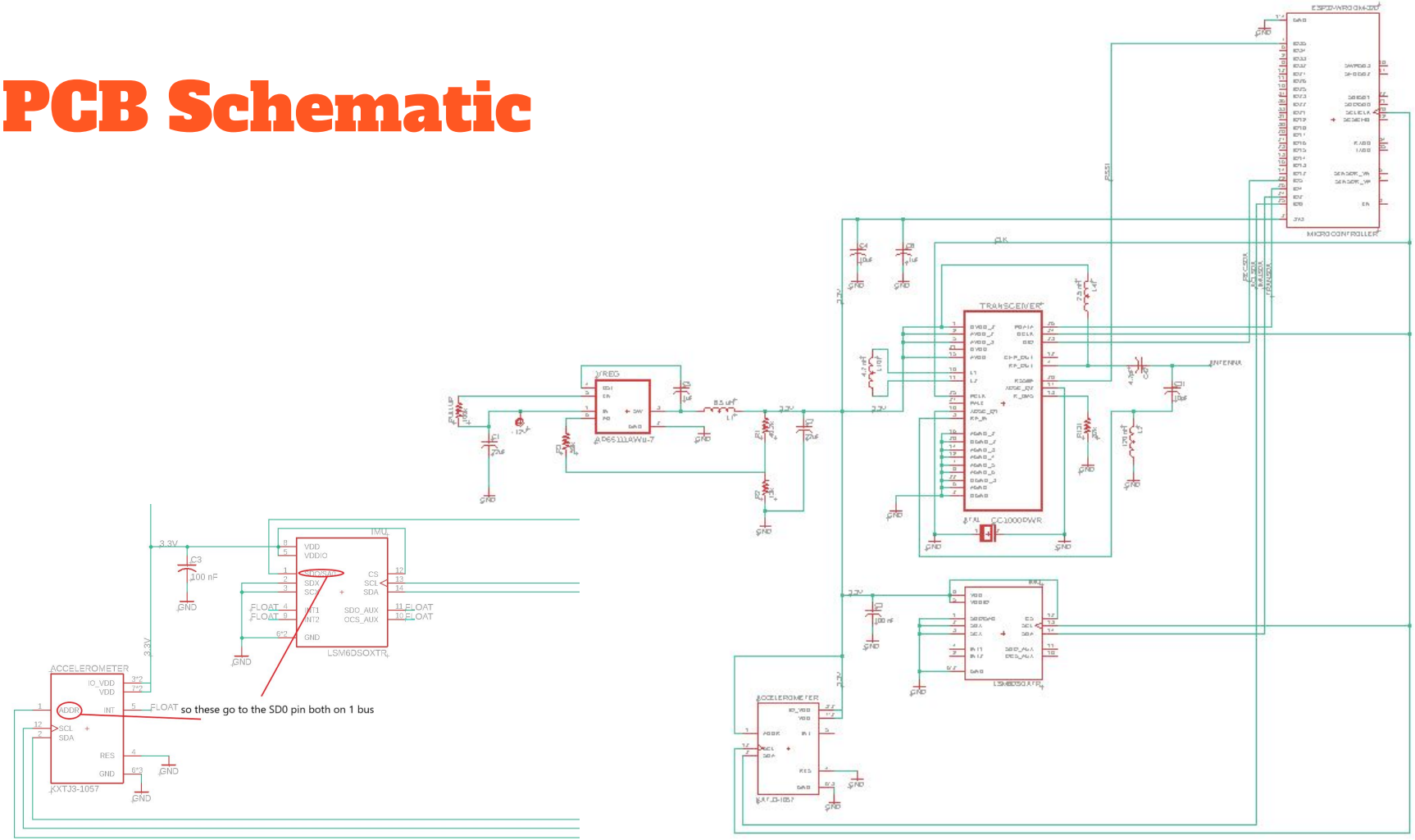
- Sensors must reliably collect and have microcontroller store sensor data at a rate of 10 measurements per second
- Device must be able to receive information from another device within 500 meters at 10kbps
- Device must provide an interface or API to export and display data externally



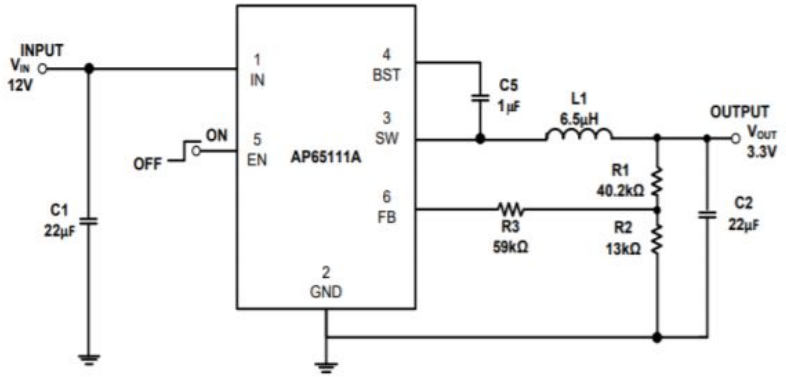
Block Diagram

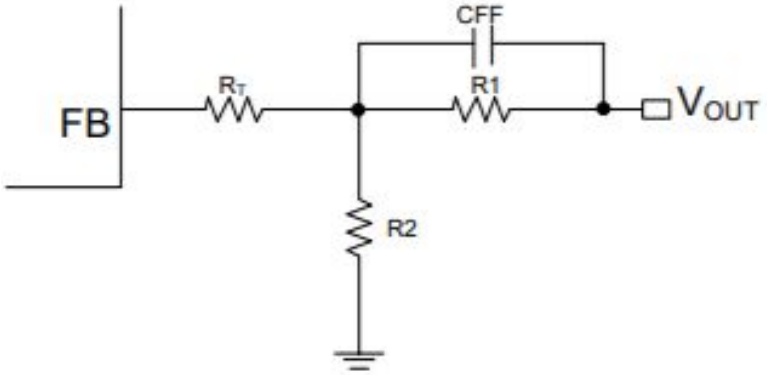


PCB Schematic



Power Module





- Power supply circuit is responsible for providing 3.3V to all of the components in the device
- Chose a buck circuit because of the efficiency the converter provides and the load that the power circuit would have to withstand

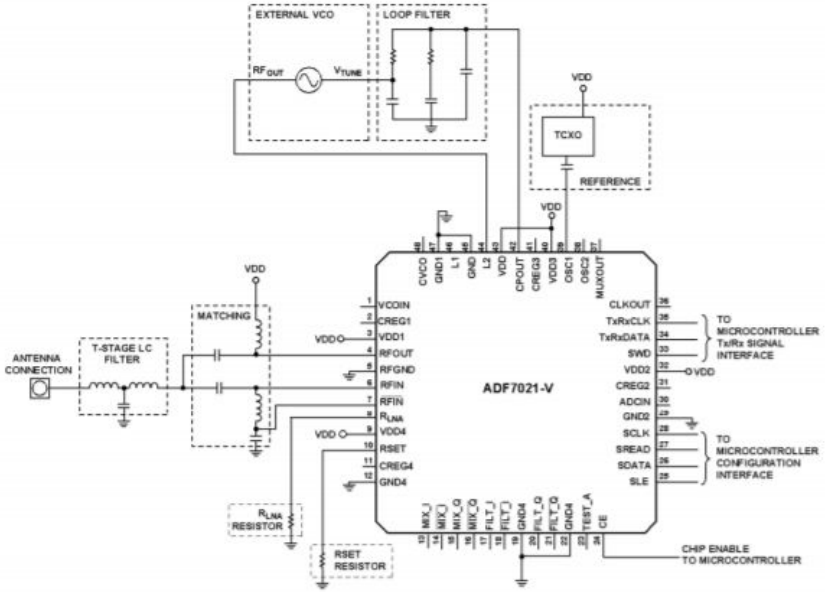
$$R_1 = R_2 \cdot \left(\frac{V_{OUT}}{0.8} - 1 \right)$$

Sensors Module

- Accelerometer
  - Provides acceleration to at least 2g and within ±10% accuracy
- Global Positioning System (GPS)
  - Provides location within 10 meters and speed within ±10% accuracy
- Inertial Measurement Unit (IMU)
  - Provides direction of heading within 10° degrees

Communications Module

- Responsible for broadcasting and receiving data from similar devices
- Initially wanted to use 5.9 GHZ
  - Allocated by FCC
- Decided to use 900 MHz
  - Balance between propagation and bandwidth
- Used CC1000 transceiver chip
  - Wide range of frequencies
  - Built in RF filters
  - Minimal external components



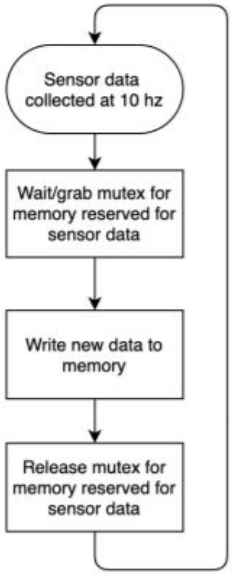
Controls Module

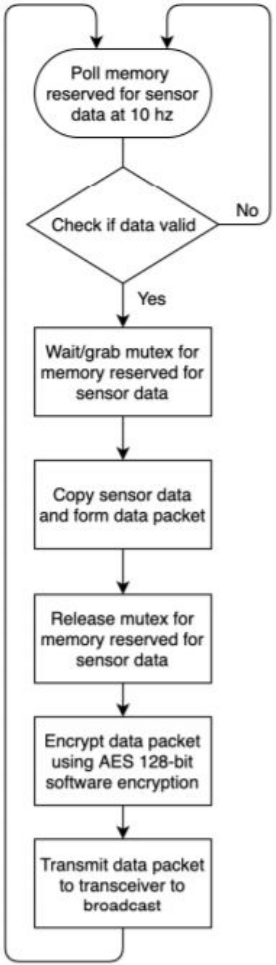
- Used ESP32 microcontroller for various features
  - Arduino IDE support
  - Low power consumption
  - Dual-core LX6 microprocessor, operating at 240 MHz (powerful processor)
  - Built in bluetooth and wifi capabilities (display)
  - Cryptographic accelerator (stretch goal)
- Responsible for interfacing between the sensors and communications module and the user

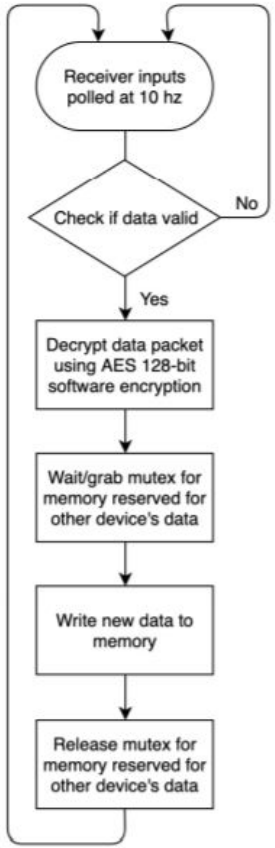
Software Flowcharts

Left to right:

- Collecting sensor data
- Transmitting sensor data
- Collecting received data

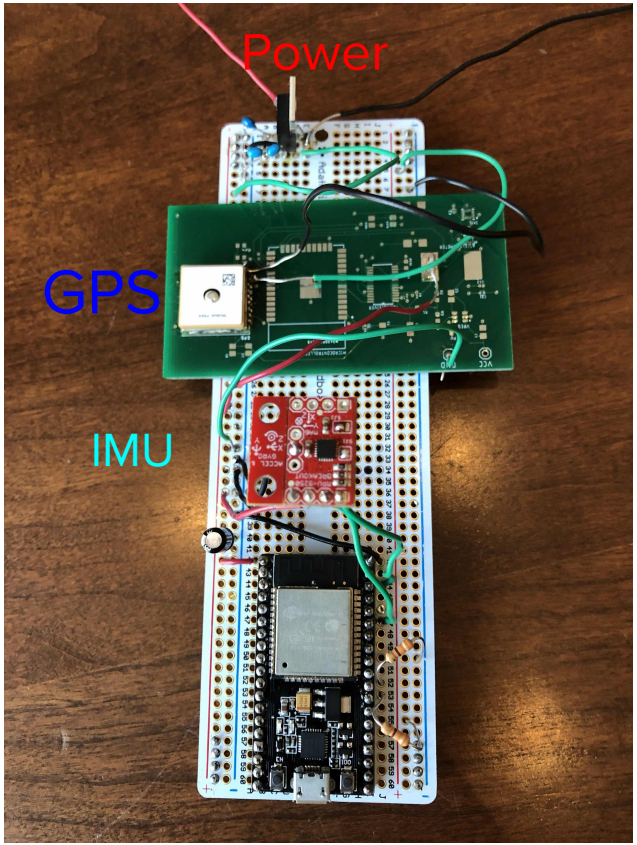






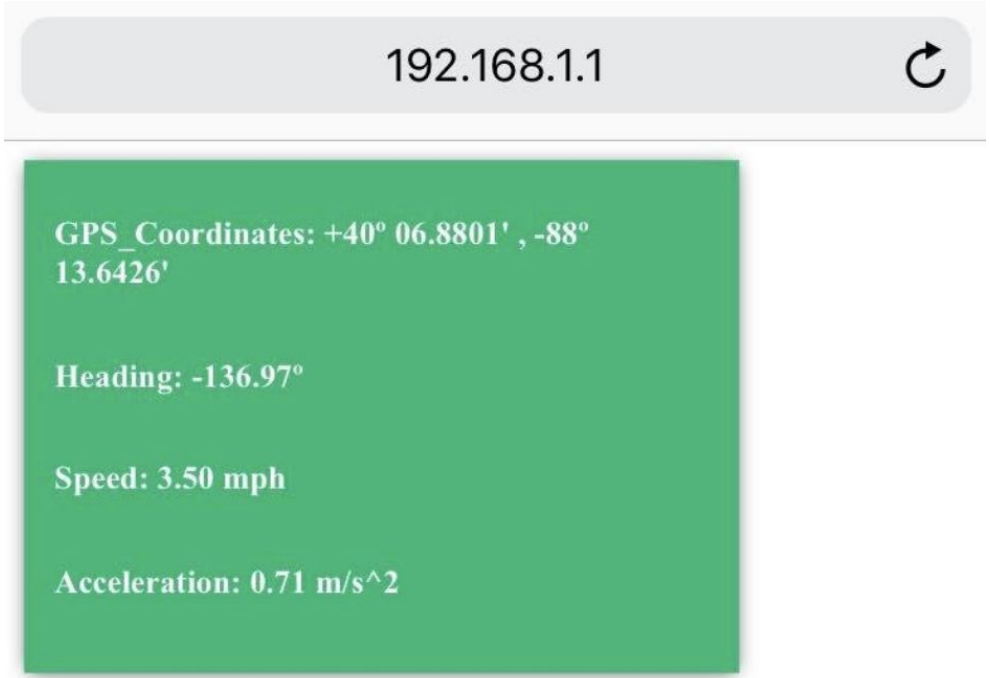
Our Progress

- Removed accelerometer, replaced IMU
  - New IMU has accelerometer and magnetometer
- Development board instead of standalone microcontroller
  - Benefit of programming with FTDI
- Limited time constraint
  - Could not finish communications module
  - Perfboard to connect our PCB with the rest of the circuit

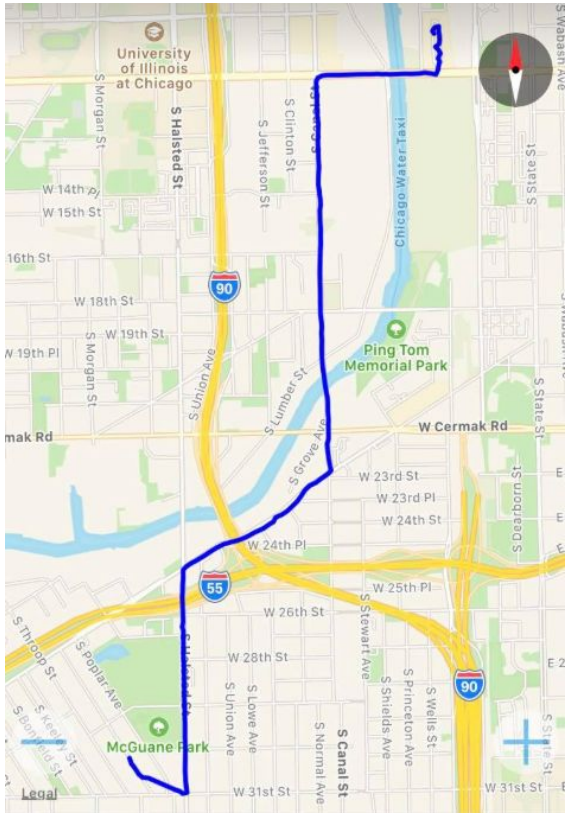


User Interface (Display API)

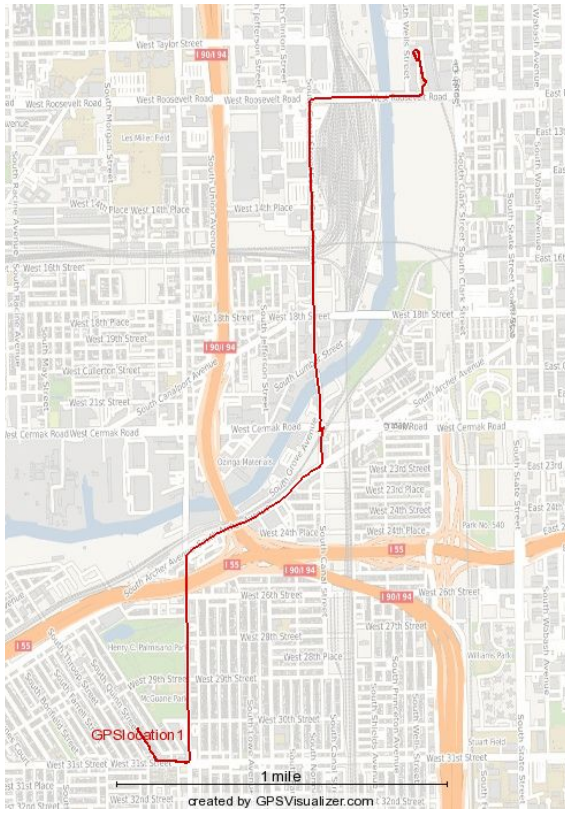
- ESP32 creates server and updates collected data on it
- Parses data and presents neatly on a webpage
- Accessible through the wifi network broadcasted from the microcontroller



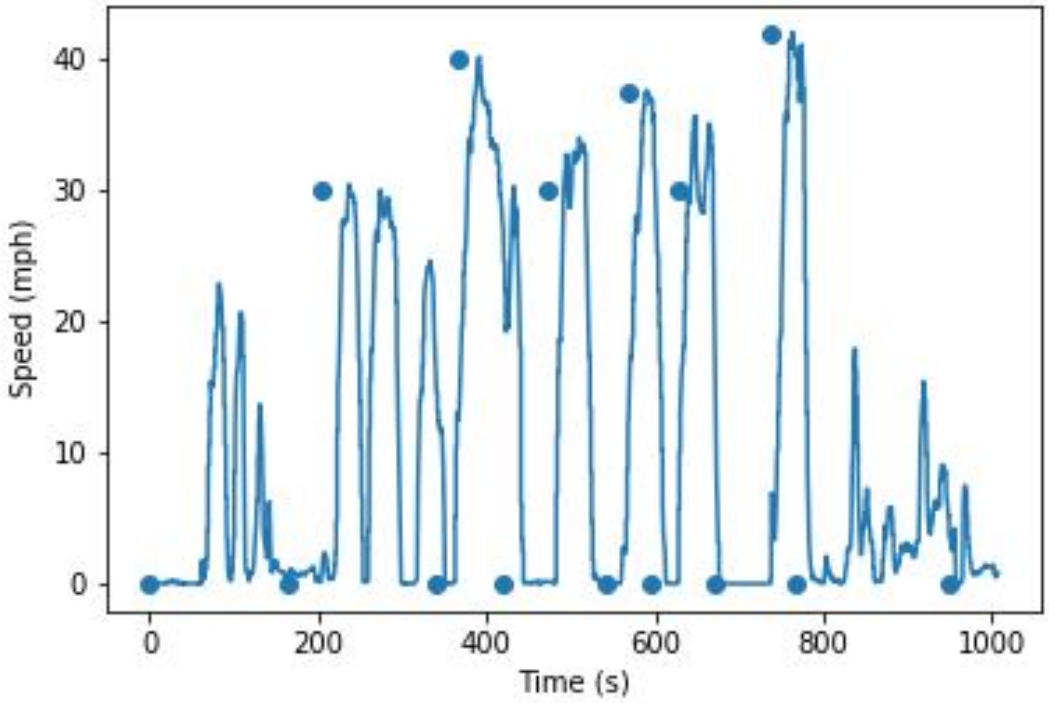
GPS Verification (Location)

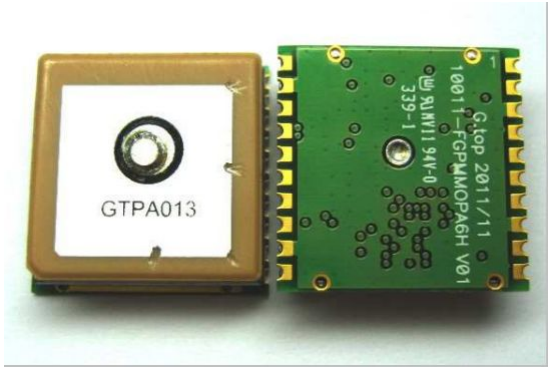


Left: GPS data from phone
Right: GPS data from our device



GPS Verification (Speed)

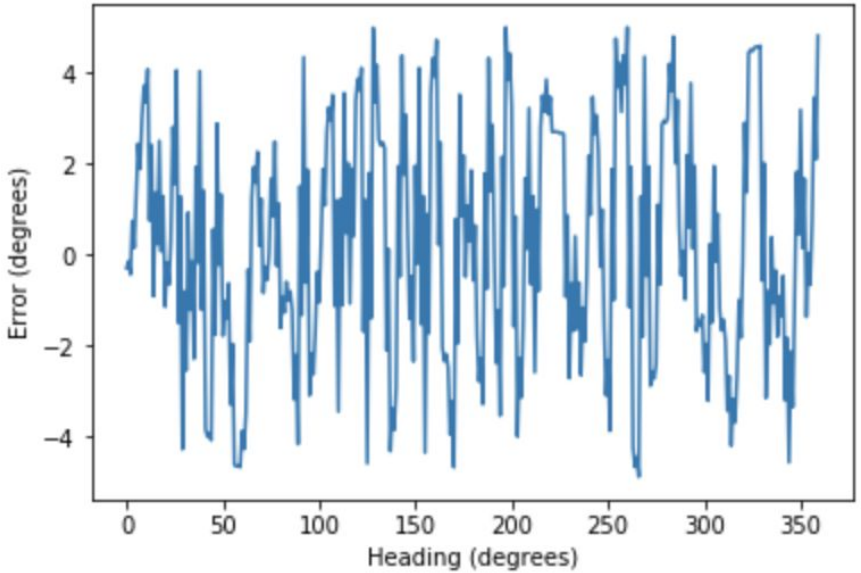




Left: GPS data collected from our device of the speed of our vehicle. The points are the speeds recorded from the speedometer.

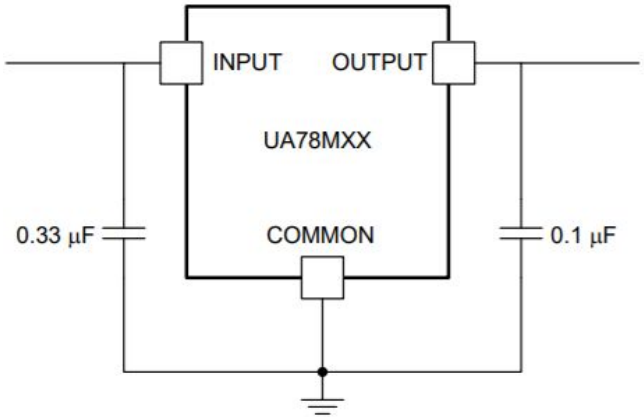
IMU Verification (Heading)

- Rotated phone and IMU about same axis
  - Compared heading readings with phone's magnetometer
- Error rate <5°



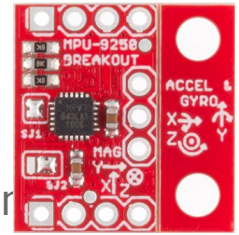
Challenges with Power Circuit

- Potential Causes
  - Small packages leading to poor soldering connections
  - Datasheet had wrong values



- Our Solution
  - Changing to a linear dropout regulator circuit

Challenges with Sensors



- GPS data was difficult to consistently receive due to interference from buildings
- Finding drivers and libraries for the IC's we chose
- Soldering the small chips onto our PCB
- Replaced the accelerometer and IMU due to limited interface
    - The new IMU was on a breakout board for ease of use to debug
    - The new IMU had the same specifications with a driver library available

Next Steps

- Finish integrating communications module
- Incorporate AES encryption
- Integration with an AI system that can recognize dangerous amounts of acceleration and speed or detect heavy amounts of traffic through many cars' locations
- Achieve a multi-nodal system where every vehicle retransmits information about all the vehicles it has data on

Thanks for Listening

Questions?