Bicycle Crash Detector

Team 68-Qihao Wang, Jiayi Wu, Ruofan Hu

ECE445 Project Proposal-Spring 2020
TA: Chi Zhang

Abstract

The purpose of a bicycle crash detector is to help the drivers of bicycle quick access to emergency resources. Once the detector identifies the crash, it will call 911 and report all the necessary information such as GPS data, name of the driver and the default message written by drivers. The crash detect system relies on the tilt angle between the ground and the vertical line when the bike is standing still. The tilt angle measured by the tilt sensor will be processed by the control unit and the control unit will activate all other modules including a GPS module, GSM module and alarm module. With all these modules, the system will be able to send the location of the crash and notify the nearby people to provide necessary help.

1 Second Project Motivation

1.1 Updated Problem Statement

Bicycles are one of the most popular vehicles in the world because of its low price and convenience. Moreover, recently, bicycle-sharing system has become popular in lots of cities. With more people using bicycles as their primary vehicles, people also start to pay attention to the safety issue around bicycles. Based on the data from the U.S department of transportation, 854 cyclists have died across the U.S and each year, around 55,000 cyclists are injured in the U.S. Moreover, the data also shows that the death of males cyclists with age larger or equal to 20 is increasing over years.

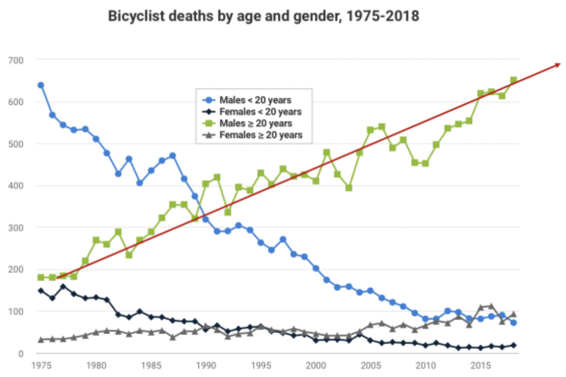
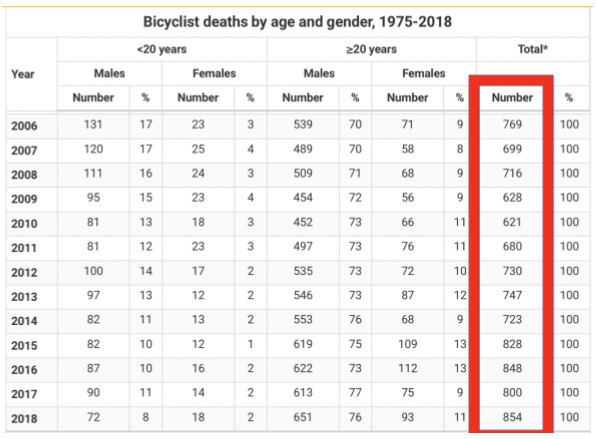


Figure 1.1 Trends



Bicyclist deaths by age and gender, 1975-2018

Year	<20 years				≥20 years				Total*	
	Males		Females		Males		Females			
	Number	%	Number	%	Number	%	Number	%	Number	%
2006	131	17	23	3	539	70	71	9	769	100
2007	120	17	25	4	489	70	58	8	699	100
2008	111	16	24	3	509	71	68	9	716	100
2009	95	15	23	4	454	72	56	9	628	100
2010	81	13	18	3	452	73	66	11	621	100
2011	81	12	23	3	497	73	76	11	680	100
2012	100	14	17	2	535	73	72	10	730	100
2013	97	13	12	2	546	73	87	12	747	100
2014	82	11	13	2	553	76	68	9	723	100
2015	82	10	12	1	619	75	109	13	828	100
2016	87	10	16	2	622	73	112	13	848	100
2017	90	11	14	2	613	77	75	9	800	100
2018	72	8	18	2	651	76	93	11	854	100

Figure 1.2 Deaths Data

The rising trend of death number, especially for the male with age over 20, is likely caused by riding with fast speed and lost control. And it is very common that cyclists lose consciousness after crashes or cannot move their hands. Under these cases, the cyclists will not be able to access the medical resources and cause their deaths. Therefore, it is necessary to design a safety insurance system to automatically provide help to cyclists when they are seriously injured.

1.2 Updated Solution

Our goal is to design a bike crash system with functionality of collision detection, automatic report crashes to police and medical department and alarm the nearby people. All modules of the system can be operated independent of cyclists so that it will cover the cases such as loss of consciousness and loss of hand functionality. Therefore, it will decrease the number of deaths caused by delay of medical help.

Our bicycle crash detector contains a solar power module, GPS & Tilt sensor module, GSM module, control module and alarm modules. The tilt sensor module will measure the tilt angle between the ground and the vertical line when the bike is standing still. If the angle is larger than 45 degree and the control unit will run the debounce code to check if the tilt angle is caused by "false cases" such as turning or vibration.

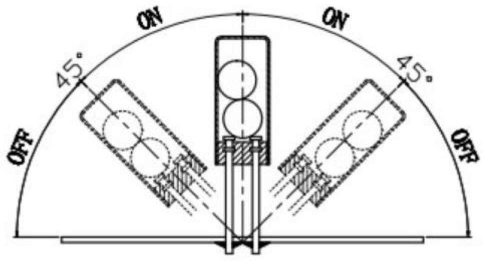


Figure 1.3 Tilt Sensor

Once the system identifies a crash, the control modules will activate all other modules. The GPS module will provide the location data. The GSM module will send the GPS data along with the default message to 911 and user selected contacts. The alarm module will start to flashlights and make an alarm ring sound to notify the nearby people to locate the crash. Lastly, our system will rely on solar power, which is less costly to the environment.

There are similar products on the markets. Most market products detect the crash through impact sensor. They pass the crash message through Bluetooth to the APP and then the message is transmitted to the police department and selected contact. And their system relies on the normal batteries with an approximate 6 months' life.

Compared with our solution, the solution in the current market has a fatal problem. The communication ability of these products completely relies on the user phone. However, the user phone could run out of power or far away from the user. Without the support from the phone, these products will lose their functionalities completely.

The solution provided by the previous group contains a battery module, GPS & acceleration sensor module, GSM module and control module. The previous group will read the data from the acceleration sensor. Then their control unit will use the logic in the following flow chart:

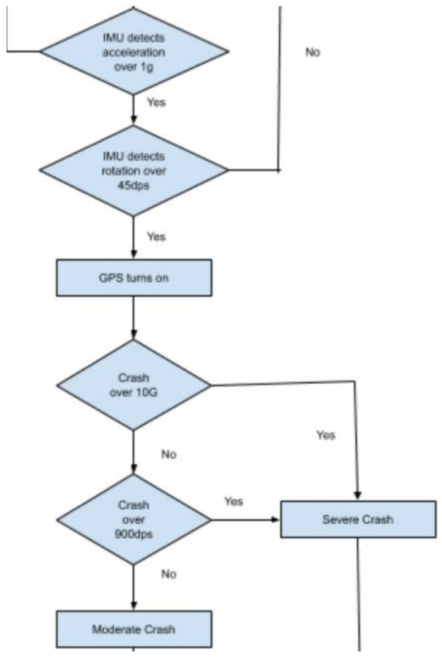


Figure 1.4 Flow Chart

Compared with the previous group solution, our design adapts the GSM and GPS module. However, we use a completely different sensor system to detect the crash. In our design, the tilt sensor only cares about the angle and the debounce code filters out the false cases by examining the duration of tilt angle. In the previous group's design, they mainly measure the angular velocity and the acceleration of the bike. Such design cannot filter out false cases such as brake or quick turning with a large angle. Further, the previous group does not offer an emergency stop button for false cases, so the false case could report a false crash accident to the police department.

In conclusion, both the solution provided by the previous team and current solution in the market use fewer components compared to our solution. Therefore, their solutions are less cost in money, energy and size. And the acceleration sensor can provide the data with high accuracy such as angle speed and the moving speed. However, their solutions will likely result in a high false rate in detecting or fatal functionality loss. In our solution, we can detect the crash with high accuracy, and we use solar power to cover the power consumption. On the other hand, our solution costs more money, energy and size since we have more components.

1.3 High-Level Requirements

- The system will be able to detect a crash and prevent false alarm from other controlled cases with accuracy higher than 90%
- The system will be able to send a message to emergency contact(s) with a GPS location along with the default message written by the user. The deviation of GPS location should lower than or equal 5m radius.
- The solar power system can store enough charge for the system to work for 14 hours with no sunlight.

1.4 Updated Block Diagram

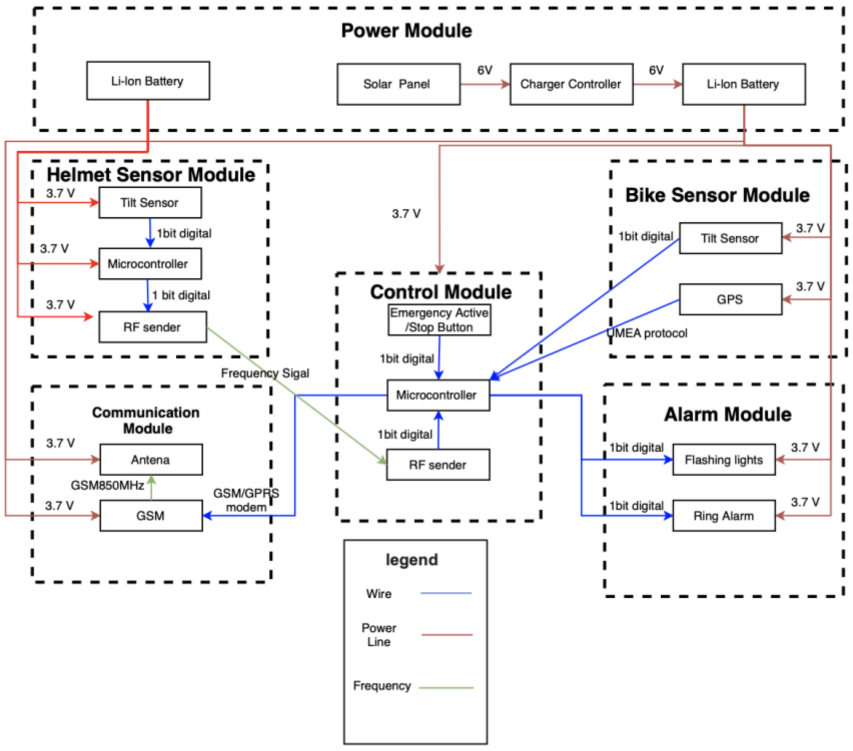


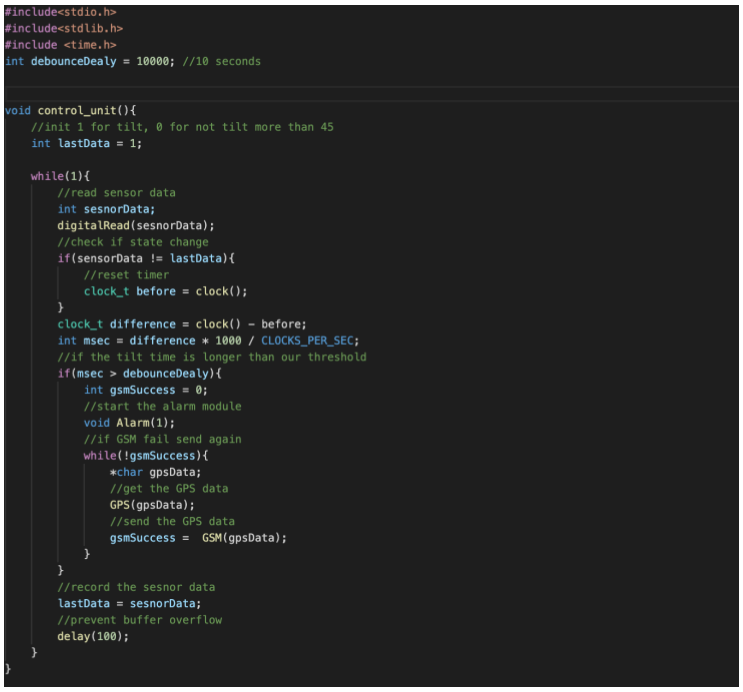
Figure 1.4 Flow Chart

2 Second Project Implementation

2.1 Implementation of software/firmware components

The Control unit main function will filter out the false case by measuring the duration of tilt angle. And when a crash is identified, the control unit will activate the Alarm, GPS and GSM modules. This is crucial for high level requirement 1, since the debounce code is closely related to accuracy.

Control unit main function:



```
#include<stdio.h>
#include<stdlib.h>
#include <time.h>
int debounceDealy = 10000; //10 seconds


void control_unit(){
    //init 1 for tilt, 0 for not tilt more than 45
    int lastData = 1;

    while(1){
        //read sensor data
        int sesnorData;
        digitalRead(sesnorData);
        //check if state change
        if(sensorData != lastData){
            //reset timer
            clock_t before = clock();
        }
        clock_t difference = clock() - before;
        int msec = difference * 1000 / CLOCKS_PER_SEC;
        //if the tilt time is longer than our threshold
        if(msec > debounceDealy){
            int gsmSuccess = 0;
            //start the alarm module
            void Alarm(1);
            //if GSM fail send again
            while(!gsmSuccess){
                *char gpsData;
                //get the GPS data
                GPS(gpsData);
                //send the GPS data
                gsmSuccess =  GSM(gpsData);
            }
        }
        //record the sesnor data
        lastData = sesnorData;
        //prevent buffer overflow
        delay(100);
    }
}
```

Figure 2.1 MainFunction

Details for each line is written in comments. In general, the main function is to keep reading the sensor data and filtering out the false cases by time difference. Once a crash is identified, it calls Alarm, GPS and GSM module to fulfill their duty. Notice: The Alarm, GPS, GSM module heavily depends on the hardware, we cannot write the code without the actual hardware.

We implement a second software component to show how we read the GPS data. This is important for high level requirement 2, since we need to read the GPS data correctly so that the system can provide meaningful GPS information to the place department.

Notice the following code is modify from

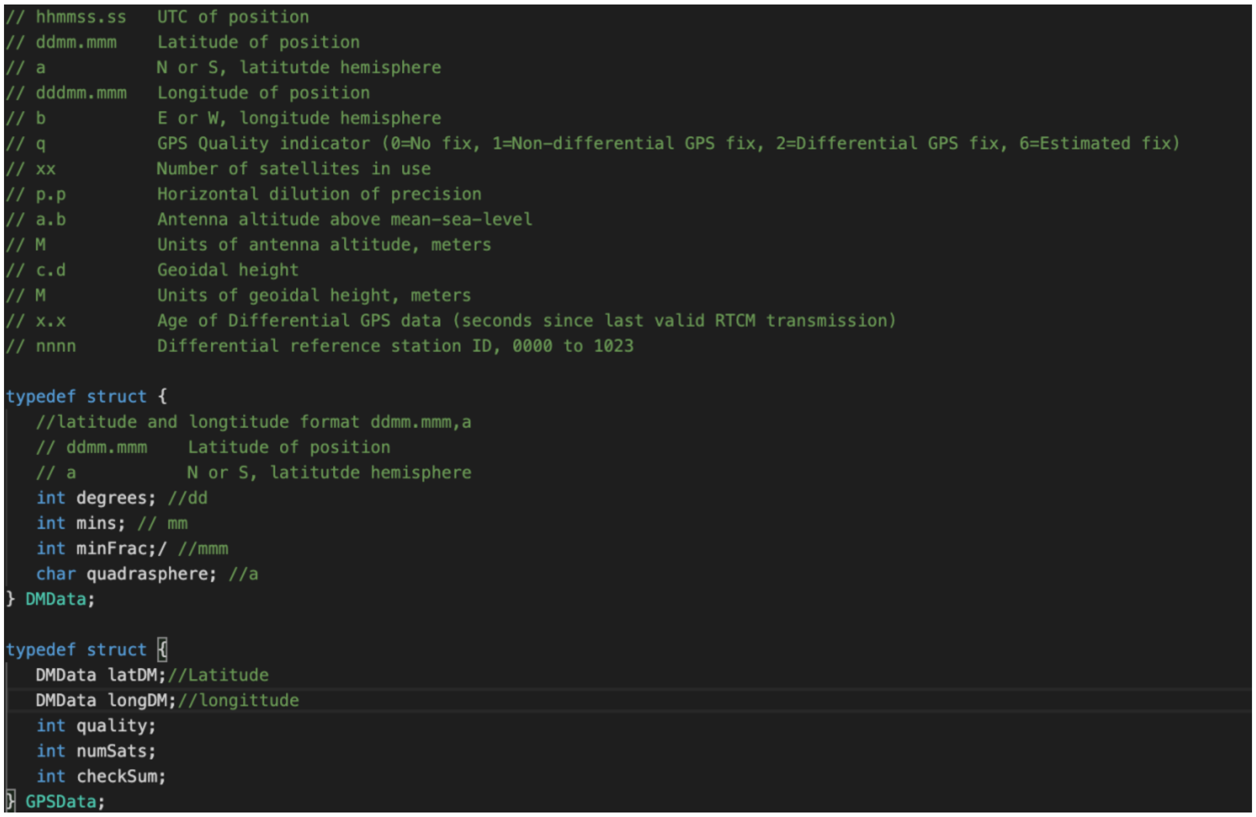
http://www.labbookpages.co.uk/electronics/gumstix/gps.html

For details about GPGGA look at:

https://learn.sparkfun.com/tutorials/gps-basics/message-formats

GPS data reading:

Gps.h contains two major structure we are going to use in this code



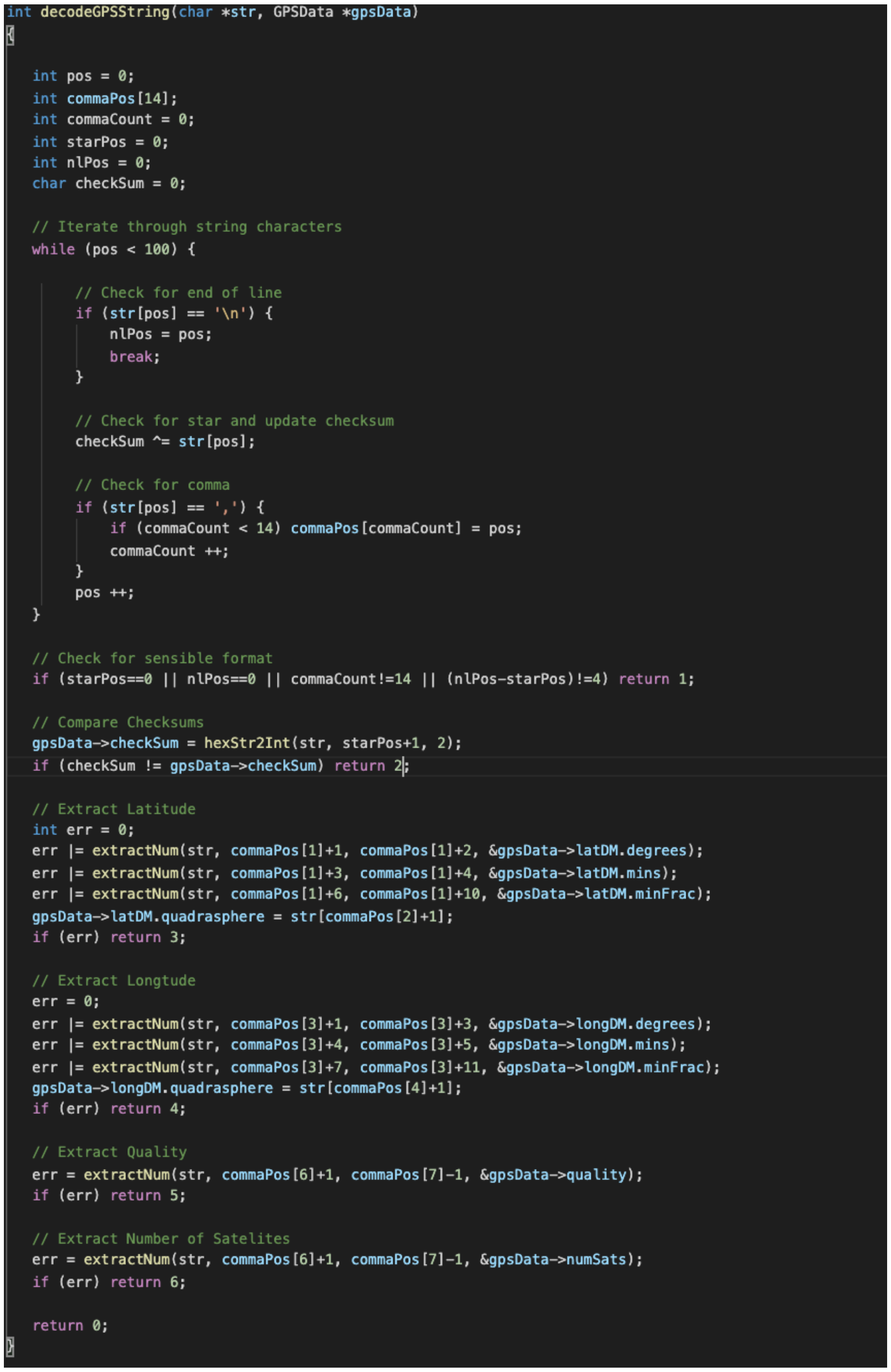
```
// hhmmss.ss   UTC of position
// ddmm.mmm    Latitude of position
// a           N or S, latitutde hemisphere
// dddmm.mmm   Longitude of position
// b           E or W, longitude hemisphere
// q           GPS Quality indicator (0=No fix, 1=Non-differential GPS fix, 2=Differential GPS fix, 6=Estimated fix)
// xx          Number of satellites in use
// p.p         Horizontal dilution of precision
// a.b         Antenna altitude above mean-sea-level
// M           Units of antenna altitude, meters
// c.d         Geoidal height
// M           Units of geoidal height, meters
// x.x         Age of Differential GPS data (seconds since last valid RTCM transmission)
// nnnn        Differential reference station ID, 0000 to 1023

typedef struct {
   //latitude and longtitude format ddmm.mmm,a
   // ddmm.mmm    Latitude of position
   // a           N or S, latitutde hemisphere
   int degrees; //dd
   int mins; // mm
   int minFrac;/ //mmm
   char quadrasphere; //a
} DMData;

typedef struct {
   DMData latDM;//Latitude
   DMData longDM;//longittude
   int quality;
   int numSats;
   int checkSum;
} GPSData;
```

Figure 2.2 DataStructure

Most of the details are in the comment’s lines.



```c
int decodeGPSString(char *str, GPSData *gpsData)
{

    int pos = 0;
    int commaPos[14];
    int commaCount = 0;
    int starPos = 0;
    int nlPos = 0;
    char checkSum = 0;

    // Iterate through string characters
    while (pos < 100) {

        // Check for end of line
        if (str[pos] == '\n') {
            nlPos = pos;
            break;
        }

        // Check for star and update checksum
        checkSum ^= str[pos];

        // Check for comma
        if (str[pos] == ',') {
            if (commaCount < 14) commaPos[commaCount] = pos;
            commaCount ++;
        }
        pos ++;
    }

    // Check for sensible format
    if (starPos==0 || nlPos==0 || commaCount!=14 || (nlPos-starPos)!=4) return 1;

    // Compare Checksums
    gpsData->checkSum = hexStr2Int(str, starPos+1, 2);
    if (checkSum != gpsData->checkSum) return 2;

    // Extract Latitude
    int err = 0;
    err |= extractNum(str, commaPos[1]+1, commaPos[1]+2, &gpsData->latDM.degrees);
    err |= extractNum(str, commaPos[1]+3, commaPos[1]+4, &gpsData->latDM.mins);
    err |= extractNum(str, commaPos[1]+6, commaPos[1]+10, &gpsData->latDM.minFrac);
    gpsData->latDM.quadrasphere = str[commaPos[2]+1];
    if (err) return 3;

    // Extract Longtude
    err = 0;
    err |= extractNum(str, commaPos[3]+1, commaPos[3]+3, &gpsData->longDM.degrees);
    err |= extractNum(str, commaPos[3]+4, commaPos[3]+5, &gpsData->longDM.mins);
    err |= extractNum(str, commaPos[3]+7, commaPos[3]+11, &gpsData->longDM.minFrac);
    gpsData->longDM.quadrasphere = str[commaPos[4]+1];
    if (err) return 4;

    // Extract Quality
    err = extractNum(str, commaPos[6]+1, commaPos[7]-1, &gpsData->quality);
    if (err) return 5;

    // Extract Number of Satelites
    err = extractNum(str, commaPos[6]+1, commaPos[7]-1, &gpsData->numSats);
    if (err) return 6;

    return 0;
}
```

Figure 2.3 Parse GPS data

Basically, the code is doing a text parsing format of GPGGA string. There are 6 types of error possible. 1. Format error 2.checksum error(means there is error bit) 3-6 GPS data contains error. hexStr2Int and extractHumber are two very simple functions. hexStr2Int will convert hex string to int and extractHumber just convert a string to digit one by one.

2.2 Software-based simulation

We could simulate some very basic cases. We can manually generate a series number 1 and 0 sensor data to test our control unit main function. Here is our result.

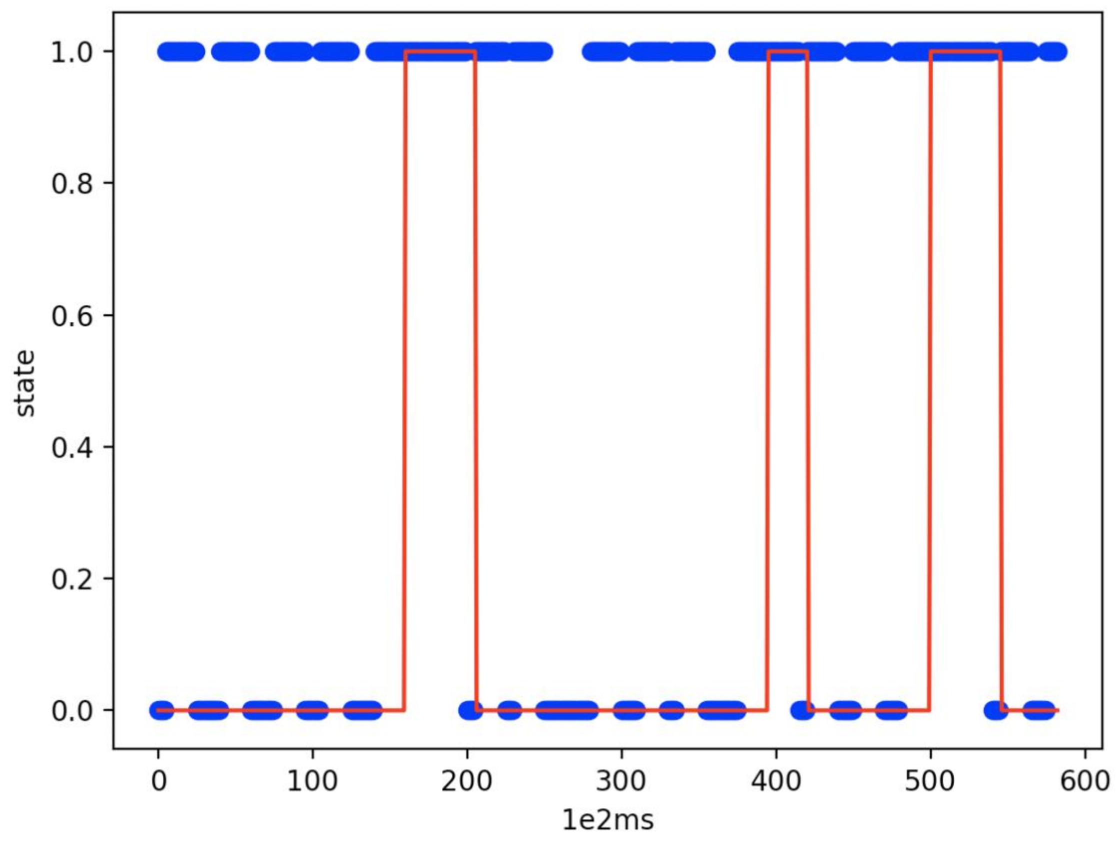
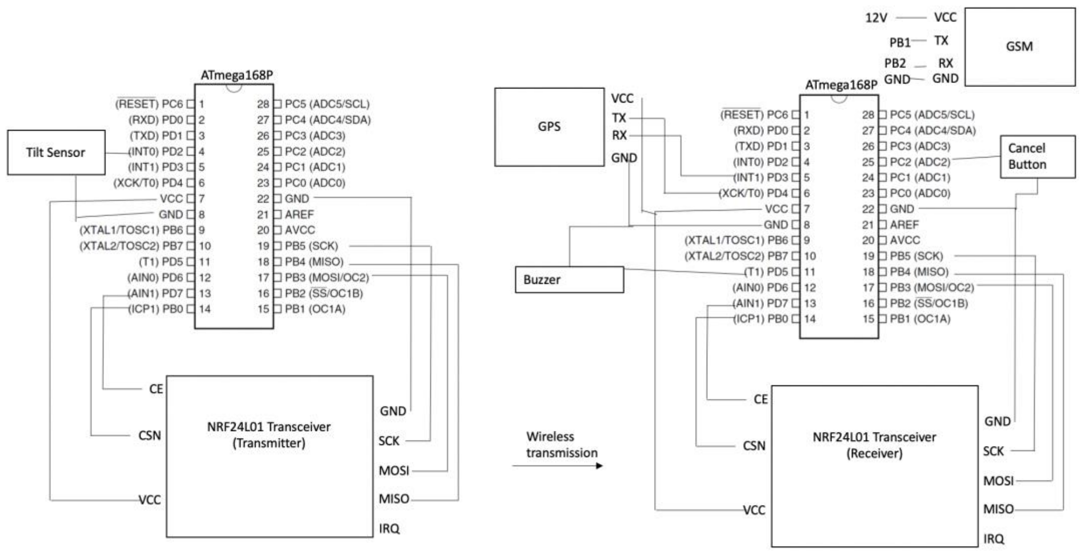


Figure 2.4 Software Simulation

The y-axis is the state of the sensor. And the x-axis has a unit of 100ms. The blue data point is random generated tilt sensor data, which 0 means less than 45 degrees. The red line represents the state of control, which high means a crash happened and low means normal. We simulate the code for 1min. In the experiment, we set the debounce delay to be 2s, so for tilt time longer than 2s, we will see the signal change for the red line means that it identifies a crash.

2.3 Schematic & Board Layout



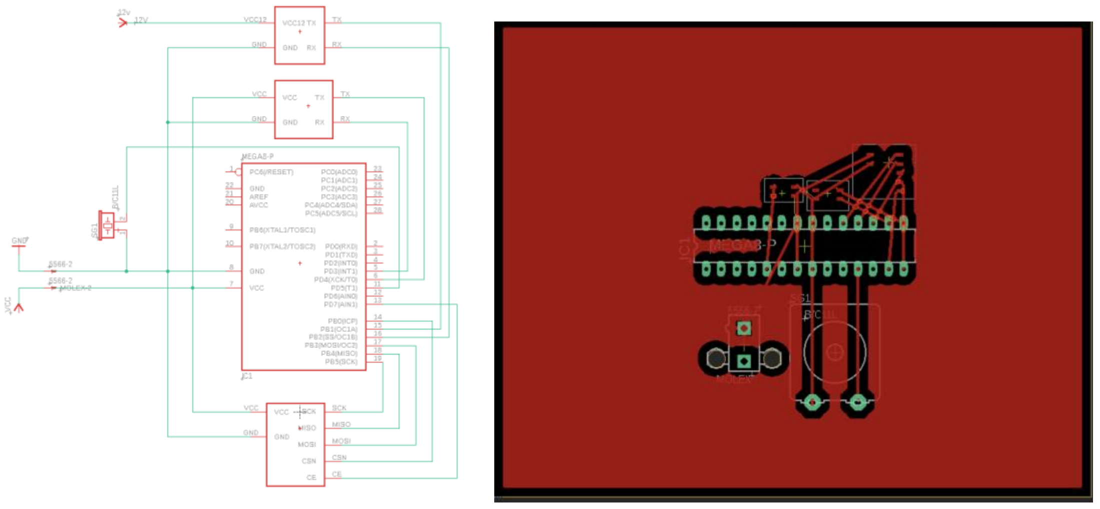


Figure 2.5 schematic & PCB view for receiver part

2.4 Mathematical analysis

The stability of the power system is very important for our system to function properly. We want to analyze our power consumption in sleep mode and activated mode. Sleep mode refers to the case which the driver is driving normally. Under sleep mode, only tilt sensors, GPS modules, and microcontrollers are consuming power. Activated mode refers to the case when a crash happened. Under activated mode, all modules are consuming power.

Sleep Mode:

GPS Module Current Consumption = 8 uA

ATmega88 Current Consumption = 1.8 mA

SIM Module Current Consumption = 6.0 uA

RF sender/Receiver Current Consumption = 18.0 mA

Tilt Sensors = 12 mA

Total Current Consumption = GPS Module + ATmega88 + SIM Module + RF Module

= 8 uA + 1.8 mA + 6.0 uA +18.0 mA +12 mA

= 31.814 mA

Activate Mode:

GPS Module Current Consumption = 16 mA

ATmega88 Current Consumption = 1.8 mA

SIM Module Current Consumption = 350 mA

Buzzer = 10 mA

LED = 20 mA

RF sender/Receiver Current Consumption = 18.0 mA

Tilt Sensors = 12 mA

Total Current Consumption = GPS Module + ATmega88 + SIM Module + RF Module

= 16 mA + 1.8 mA + 350 mA +18.0 mA +12 mA +10 mA + 20 mA

$$= 417.8\ mA$$

Now if we assume that the system will work 13 hours under Sleep Mode and 1 hour under Activate Mode. We have

$$Combined\ Model\ Current\ Power\ Consumption = Activate\ Mode * 1\ hour + Sleep\ Mode * 13\ hours$$

$$Combined\ Model\ Current\ Power\ Consumption = 417.8\ mA * 1\ hours + 31.814\ mA * 13\ hours$$
$$= 831.382\ mAH$$

Next thing, we want to fully charge our Li-ion batteries with 7 hours of sunlight.

$$battery\ charger\ current = Battery\ Capacity / Charging\ time$$

Formule 6.1

$$battery\ charger\ current = 1000mAH / 7H = 142.857142857\ mA$$

Therefore, in our project we choose a 180 mA solar panel and 1000mAH battery. In reality, the power will be lost during the transition of different components, so we choose our solar panel and battery larger than the result calculation. We believe that our power system should meet the high-level requirement in our project.

2.5 Implementation Conclusion

Our implementation shows that in theory, it is possible to detect the crash and filter out some false case. The simulator result shows that our code can detect the crash at least for some naive cases. And we also show the code for microcontrollers to parse the output from the GPS module. Lastly, tolerance calculation shows that our choice of solar panel and battery can support our system to run 14 hours. All these implementations show that we can at least accomplish GPS modules, solar power modules and tilt sensor modules. However, most of our data is generated manually, found on the data sheet and various online resources. Therefore, in reality, we may need to change some parameters or modify parts of code to meet high-level requirements.

3 Second Project Conclusions

3.1 Implementation Summary

In chapter 2, we implement the control unit code, GPS module reading code, a simple software experiment of our detect system, schematic and tolerance analysis about solar power. The control unit code is essential for our project, since it contains the debounce code, which is key to filter out false cases. The simple experiment shows that our control unit will be able to identify and filter out some simple cases. The GPS module reading code shows the control unit will be able to process the data from different modules. And the schematic and tolerance analysis provide evidence to support our design. In conclusion, our design can be accomplished but we still need the actual hardware implementation to verify the high-level requirements.

Jiayi Wu implements control unit code, GPS module reading code and a simple software experiment for crash detect system.
Qihao Wang implements schematic and board layout.
Ruofan Hu implements tolerance analysis.

3.2 Unknowns, uncertainties, testing needed.

The crash detect system requires a very short amount of response time and the response time of the system is dependent on the hardware behavior, which has lots of uncertainties. For example, the acquisition time of GPS data is related to the actual environment of the bike. If the bike is located in a normal campus environment such as around the main campus building, it will be relatively easy to identify in a short time. On the other hand, if the bike is located inside a building such as a garage, the acquisition time of GPS data may increase. Therefore, a series of response time tests in different environments is required in the design.

Moreover, it is possible that the system is damaged during the crash. Even though our design uses two sensors to ensure that the system will be able to gain data during a violent crash, the vibration caused by the crash may short the circuit or damage the functionality of the CPU. The stability of the system in different crashes is unknowns. Different crash tests are also required for our system. The test result may require additional gears or structure design to help stabilize the system during the crash.

3.3 Ethics and Safety

There are several safety and ethical issues involved in our system. The device itself is pretty harmless to users but may have negative impacts on the public and cause ethic issues.

The first potential risk is the lithium battery that our system uses to store the energy. If disposed improperly, lithium batteries can be very dangerous. Thus, we must prevent batteries from being exposed to dangerous conditions like overcharging by monitoring the temperature of the battery and warn our users of the potential hazards the batteries can do.

According to [5]IEEE code of ethics term No. 1, we must hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, and to disclose promptly factors that might endanger the public or the environment; The functionality of our system involved alarming and automatic calling to the police office. So, it is crucial for us to maintain the accuracy of our systems to avoid false alarms to the police or the public which may jeopardize the public welfare and efficiency. Moreover, maintenance and tracking are important to our system because our system is likely to be implanted on shared bicycles which means there's risk that it might get lost or stolen which will also affect the public welfare.

Moreover, the GPS module contains the real time position of users. It is our duty to protect the privacy of the users. Our system only connects to telephone networks through the GSM system rather than the internet network with WIFI modules. Such design lowers the risk for someone hijacking our system to locate the user for illegal purposes. If the users have deep concern about leaking their privacy, we can design an additional code module so that it will not activate the GPS module before crashing. Such design may increase the risk of loss of GPS data when a crash happens, but the system will have less privacy problems.

3.4 Project Improvements

First of all, it is very crucial to improve our crash detect sensor system by adding more sensors. Because the project is designed to compete within 13 weeks, our design only uses tilt sensors. However, it is possible to combine several different types of sensors so that we can build an advanced model to evaluate the crash and filter out false cases. In our current design, the crash detect system will only work with the campus environment, in which most roads are flat. With advanced sensor systems, our system can be applied to dangerous and complex environments such as mountains. For example, in bike extreme sports, players will maintain a large tilt angle for a very long time, so that they pass some dangerous short cuts with high speed. Under this case, an impact sensor will be a better choice. Therefore, in general, with more time, we could improve our project by utilizing more sensors to build a advanced crash detection system.

The second improvement will be stability of the system during a crash. Stability is currently the biggest concern in our project. There are too many unknowns for a crash. Our system could suffer serious damage during crash and loss of functionality. With more time, we want to perform a series of tests or design physical models for different types of crashes so that we can know the force, speed or angle during the crash. With these data, we can place our system in a better position or change its size or shape so that it will have a better chance to survive in the crash and offer help to the users.

The last improvement will be the GSM communication system. The communication system is only available with areas covered by the local communication company. However, based on our experience, the server from a local communication company is not always stable. And the failure of the communication system is fatal. Without the function of the communication system, the system cannot offer any meaningful help to the driver. Such problems can be solved by a satellite phone system. Again, such a system will likely be favored by extreme sports players. Therefore, with more time, we are willing to find a more reliable replacement solution for the GSM communication system.

4 Progress made on First Project

The main progress in our first project is our physical design and the PCB design. For the physical design, we talked to the machine shop multiple times. And we settled down the final design and ordered the necessary parts such as motors, battery. However, there is not enough time for machines to build it. And the same case for the PCB.

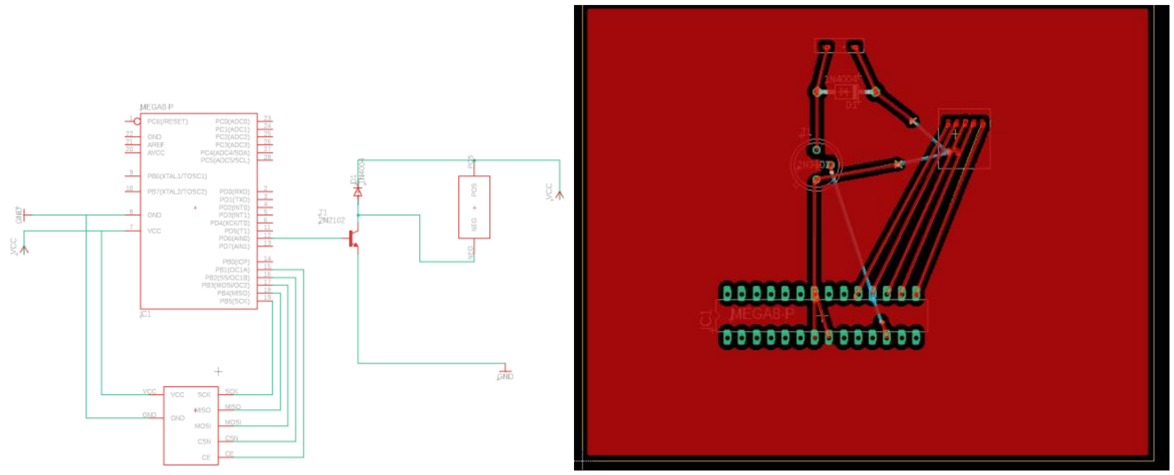


Figure 4.1 Schematic view for receiver part

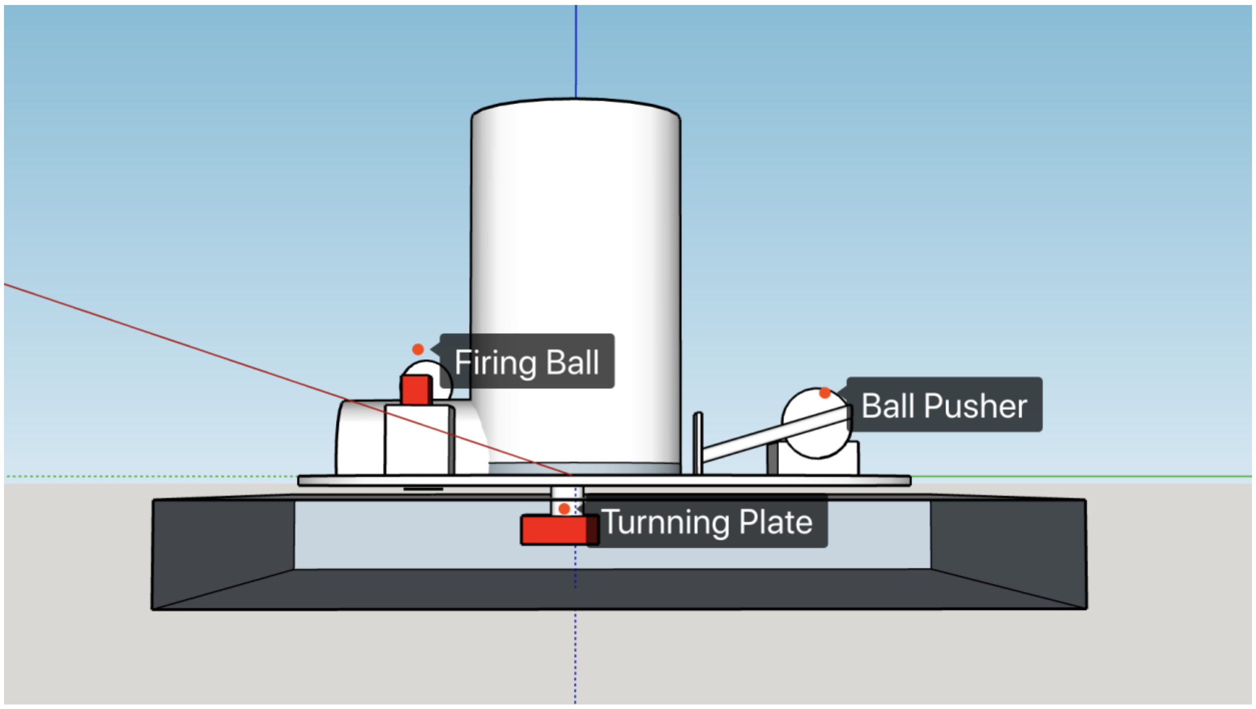


Figure 4.2 Right Side View of Physical Design

5 Citations and References

[1] National Center for Statistics and Analysis, "*Bicyclist and Pedestrian Safety - nhtsa.gov,"NHTSA*. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/documents/14046-pedestrian_bicyclist_safety_resources_030519_v2_tag.pdf. [Accessed: 18-Sep-2019].

[2] Fatality Facts 2018 [Online]. Available: https://www.iihs.org/topics/fatality-statistics/detail/bicyclists.

[3] CYCLIST INJURIES FROM AUTO ACCIDENTS[Online]. Available: https://bayareabicyclelaw.com/safety-laws/bike-stats/

[4] *Renewable Energy World, 'Solar-powered Internet Connectivity in Lascahobas, Haiti', 2012.* [Online]. Available: http://www.renewableenergyworld.com/ugc/articles/2012/01/solarpowered-internet-connectivity-in-lascahobas-haiti.html.

[5] *"IEEE Code of Ethics." IEEE*. Accessed April 3, 2020. https://www.ieee.org/about/corporate/governance/p7-8.html.

[6] SW-520d datasheet. Available: http://funduino.de/DL/SW-520D.pdf

[7] image of bike. Available: https://detroitbikes.com/products/b-type-1

[8] image of helmet. Available: https://www.explorethousand.com/products/skateboard-helmet?variant=31709206118447&utm_source=google&utm_medium=cpc&utm_campaign=TNT%7CThousand%7CSmart%20Shopping%7CHelmets&utm_content=Helmets&gclid=CjwKCAjwvZv0BRA8EiwAD9T2VcU2Oo0yrTFcqBN9-_i8QsH-fgnbvijqKVfekRJ9hm0fBN0DHqA5zhoCp44QAvD_BwE

Sound intensity level https://opentextbc.ca/physicstestbook2/chapter/sound-intensity-and-sound-level/

Li-lon Battery Charger with Thermal Regulation in SOP-8 Data Sheet:

https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf

RF receiver/sender Data Sheet:

https://www.sparkfun.com/datasheets/Components/nRF24L01_prelim_prod_spec_1_2.pdf

Previous Team’s Design Document:

https://courses.engr.illinois.edu/ece445/getfile.asp?id=16233