**Department of Electrical Engineering and Computer Science**

**University of Central Florida**

# Bike Dash

**Group # 11:**

Vincent Altavilla[1], Computer Engineering

Jose Davila[2], Computer Engineering

**Sponsor**:

*Duke Energy, Inc.

---

1. vince.altavilla@gmail.com
2. mannydavila@knights.ucf.edu

## Table of Contents

## Tables

# Figures

## Equations

## 1.0 Executive Summary

Bike Dash is a product idea that implements an easy to use bike monitoring system that provides the user with a more enjoyable experience by keeping them informed of their current biking conditions, progress, and performance. It is intended that Bike Dash will implement features such as GPS location, speed, distance traveled, current temperature, altitude, and a few other features as outlined in the basic system flowchart (Figure 1.0) seen below.

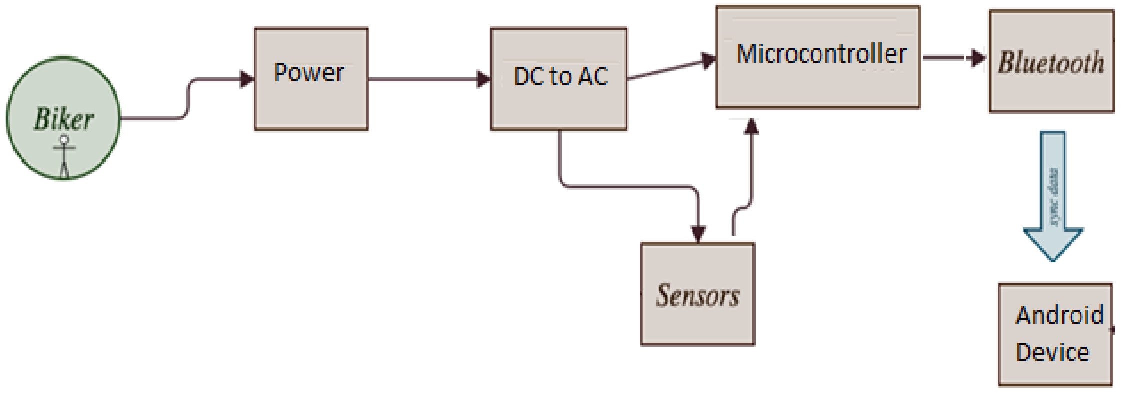


*Figure 1 - System Flowchart*

Bike Dash is intended to target bike riders of all types and is intended to be designed as light and economical as possible. All electronics will be powered by utilizing the rider's momentum through the use of a dynamo attached at the rear wheel of the bicycle. The intention to make Bike Dash light, yet durable is of utmost importance but because of the terrain traveled on a bicycle, and the possibility of inclement weather, we must ensure that the system is waterproof as well as shock resistant as well. Ideally, the device would be housed in durable plastic, possibly ABS, and sealed with silicone.

In order to move forward with the "green" effort in today's society, our device will utilize the power generated from the users pedaling by affixing a dynamo to the rear wheel and creating a voltage that will power all of our bicycle's components. To ease in the design process, a dynamo will be purchased as a COTS (Commercial off the Shelf) unit. It will be fitted on to our bike and passed through at which point we will outfit our system with power connections for our microcontroller and sensors.

During the ride (or after the ride is completed) the user will sync their data via Bluetooth to their Android enabled device. This will make the data viewable in a rich GUI layout and will also store the data on the phone. Storing the data on the phone will allow the user to retrieve previous data and compare to current data for route analysis. The user will also be able to see data graphically alongside the statistical data to provide comparisons of previous and current runs. The Android application will ultimately be the center for data collection and analysis. The implementation of and Android device as an LCD screen will allow the user to see real time data on the device mounted on the bike. The Android application will have a much richer interface than a simple LCD and will be able to archive much larger amounts of data than our microcontroller. It is for this reason we will utilize our Android device as the LCD screen.

## 1.1 Project Motivation

The motivation for this product stems from the idea of today's society moving forward towards a healthier, greener lifestyle. Many people today are overweight and lack the daily exercise that an average human being should receive. With the development of a low cost fitness tracker, hopefully enabling a more efficient and fun ride, more consumers will be encouraged to ride their bike into a healthier future. We also hope to appeal to more experienced riders that wish to travel on their bikes but do not set out with a specific route or riding plan. Advanced users that utilize our product would be able to track their location and save any routes that they would like to revisit. The route would be added to a database and the user would be able to recall their information from previous rides. This feature would, hopefully, appeal to riders of all skill levels.

Another motivation for our project was the availability of similar designs that seemed to provide only part of the scope of our intended project. The main difference in our design is that we intend to sync data through Bluetooth to an Android device that will act as the LCD screen (or output display) for the system. We intend on building upon current systems to also enhance the capability, durability, and portability of both the device and its data. We will also extend a few of the capabilities that were available in similar systems, which we will review in Section 3.8.

The last motivational factor in choosing this project was the ability to utilize previous knowledge in order to better plan and design our project. With two out of three members of the design team being computer engineering majors, choosing a project that would be able to offer a substantial amount of code as well as embedded applications was a must. With the combination of experience in programming, embedded systems, and previous experience with the PIC microcontroller we hope to streamline the process and achieve better results.

## 1.2 Technical Objectives and Goals

Though there are many things we wish to accomplish in our project, there stand out only a few key technical objectives and/or goals that retain high priority. The key objectives we will focus on are as follows:

- Durability – Must withstand inclement weather incl. high temps, rain, and snow
- Portability – Must be implemented economically and fashionably
- Power Consumption – Must be considered "low power" with minimal waste
- Economics - Must have balance of cost vs. performance

The latter is of least importance, but does still maintain a high priority as economics has always been very much a governing factor in engineering.

In regards to durability, our device will encounter many inclement weather conditions through its lifetime. From snow and sleet to rain and dry heat, our device must be able to cope with all (or at least most) temperature conditions. With this in mind, we will try to select the temperature sensor that has the broadest temperature range. Though we must consider extreme temperatures, there is a limit to what we must consider. For example, we do not intend for our product to be used in desert conditions or arctic situations. Since our device will be used for fitness, there is also a strong chance it will be subjected to dirt and mud meaning it must be both water and dust proof ideally, however, a resistant product would be acceptable. To consider a product to be waterproof, we must be able to submerge the product completely underwater for a period of time without and leakage. This is relatively difficult to do to the entirety of our system and could be a project of its own. With this in mind, we will (at a bare minimum) ensure that our product is water-resistant. To be water-resistant means that the device would be able to accept direct contact from water (possibly due to rain or a puddle) for a period of time without causing harm to the device. In designing our device to be water-resistant, we will in turn cause our device to be dust-proof as well with only minor modifications.

Another very important technical objective is to make our device as light and portable as possible. To do this we will use the lightest materials available for our application to ensure that we keep our total weight under 10 lbs. For our mounting system and housing, we will most likely make use of ABS plastic as it is both durable and light. At a weight of 10lbs the system (including dynamo) should not affect the functionality of the bike, nor impede the rider's movement. We will ensure that all wires are secured to the frame out of the rider's general range of motion and that the wires do not (ideally) pass over any joints or otherwise moving parts.

Additionally, we will look at power consumption and try to determine a desired upper bound for the overall power consumed. Ideally, we would like to keep our total power consumption to below 5 V. This power draw would be ideal as our microcontroller can accept, at most, 5.5 V input. As we move further into the project, more specifically the design section, we will be able to identify any power issues and coordinate a proper voltage. If there is additional power available, the ability of the user to plug in their mobile device and charge it would be highly desirable. To ease the integration of the mobile device charger, we would consider using a DC (car-style adapter) since it is 12 V vs. 120 V required from a wall plug. Ideally we would be able to charge the mobile device by connecting a charger directly to the PCB, however, there is a change we will connect the phone charging system to a branch that will occur before the PCB.

Lastly, we will investigate the cost objective and overall budget for our project. While we will try to keep our expenses and overall cost to a minimum, the total out the door cost is not our primary concern. A prime example would be when comparing microcontrollers, we find that microcontroller A is cheaper than microcontroller B, however, microcontroller B offers an increase in performance. If the cost difference is minimal we will, without a doubt, choose the microcontroller with the higher performance capabilities. However, if the cost difference is much more we would be apt to compare features in a manner that would outline cost vs. performance.

## 2.0 Project Management Plan

The project management style we will follow is an egoless approach in which each group member will share responsibility in every aspect of the project throughout its life cycle. As a group, we will hold each other accountable to achieve milestones by their deadline and to obey by the applicable standards set for the project in order to design a final product that contains high-level design. Contributions will be made by each group member in all aspect of the product life cycle which include design, testing and integration. Jose, a computer engineer, will mainly focus in software design of the Android application and designing software for the onboard components such as the Bluetooth module, the GPS receiver, etc. Vince, a computer engineer, will mainly focus in designing and integrating the microcontroller. Furthermore, Vince will focus on designing the power circuit that will power to the onboard components of the design. We plan to outsource after designing and integrating our components in order to get the final components soldered onto the main board. As a team we have a common goal of delivering a reliable, ultra-low power consumption product that performs flawlessly and enhances the rider's riding experience.

## 2.1 Milestones

### 2.1.1 Milestones Discussion and Table

As a group we set a guidelines of milestone that we will need to follow in order to achieve the optimal product design by the end of Senior Design II. Following the deadlines set by the professor, we choose dates that will allowed us applicable time to finish each assignment well before the deadline. We also decided to implement early and late start times in case we run into unforeseen problems during design and integration phase. This is shown in more detailed in the following section. Table 1 shows the milestones assignment for this project in both Senior Design I and II, highlighting the time estimate for each assignment and its proposed due date.

| Work Packages | Assignments | Time Estimates & Proposed Due Date |
|---|---|---|
| Senior Design I | Initial Design Proposal | September 16, 2013 (2 weeks) |
| | Select Microcontroller | September 21, 2013 (5 days) |
| | Complete System Schematic Completed | October 7, 2013 (16 days) |
| | Select All Sensors, LCD and Dynamo | October 24, 2013 (3 weeks) |
| | First 30 Pages of Final Document Completed | October 29, 2013 (5 weeks) |
| | Final System Schematic completed | November 5, 2013 (1 week) |
| | Revisions to Final Schematic Implemented | November 8, 2013 (3 days) |
| | Second 30 Pages of Final Document Completed | November 15, 2013 (1 week) |
| | Revision to Second 30 Pages Implemented | November 18, 2013 (3 days) |
| | Final Document Completed | December 3, 2013 (16 days) |
| | Implement Final Document Revisions | December 8, 2013 (5 days) |
| | Turn In Final Document | December 10, 2013 (2 days) |

*Table 1 - Milestones Chart (Senior Design One)*

| Work Packages | Assignments | Time Estimates & Proposed Due Date |
|---|---|---|
| Senior Design II | Review Final Document and Schematic | January 3, 2014 (3 weeks) |
| | Order Parts and Board | January 13, 2014 (10 days) |
| | Begin Coding of Microcontroller | January 27, 2014 (2 weeks) |
| | Implement Temperature Sensor | February 3, 2014 (1 week) |
| | Implement Accelerometer | February 10, 2014 (1 week) |
| | Code Review to Verify Sensor Integration | February 13, 2014 (3 days) |
| | PC program GUI completed/Mobile Application | February 20, 2014 (1 week) |
| | Revisions to GUI Implemented | February 24, 2014 (4 days) |
| | Implement Gyroscope | February 24, 2014 (4 days) |
| | Code Review to Verify Sensor Integration | March 3, 2014 (10 days) |
| | Power Generation System Installed on Bike | March 10, 2014 (1 week) |
| | Code Revisions Implemented | March 10, 2014 (1 week) |
| | Testing of Current Product (i.e. Sensors, Data Transfer, etc.) | March 17, 2014 (1 week) |
| | Testing Errors Fixed | March 24, 2014 (1 week) |
| | LCD integrated into device | March24, 2014 (1 week) |
| | Temperature and GPS Integration Completed | March 31, 2014 (1 week) |
| | Test and Review Product | April 1, 2014 (1 day) |
| | Implement Changes | April 8, 2014 (1 week) |
| | Test and Review Product | April 15, 2014 (1 week) |
| | Turn in Final Project | April 17, 2014 (1 day) |

*Table 2 - Milestones Chart (Senior Design Two)*

## 2.1.2 PERT Chart

Figure 2 maps out the milestones that we set for the group in order to fulfill the requirements for Senior Design I in well design Program Evaluation and Review Technique chart. Figure 3 maps out the milestone for Senior Design II. The arrows depict the critical path that must be taken in order to finish our product on time. The boxes are split up into seven sections, beginning from the top left corner, we map out the early start time, the time estimated duration, and the early end time. The middle shows the particular milestones assignment described and beginning from the bottom left corner, we map out the late start time, the total slack time, and then late end time. For a visual explanation refer to the legend in Figure 2.

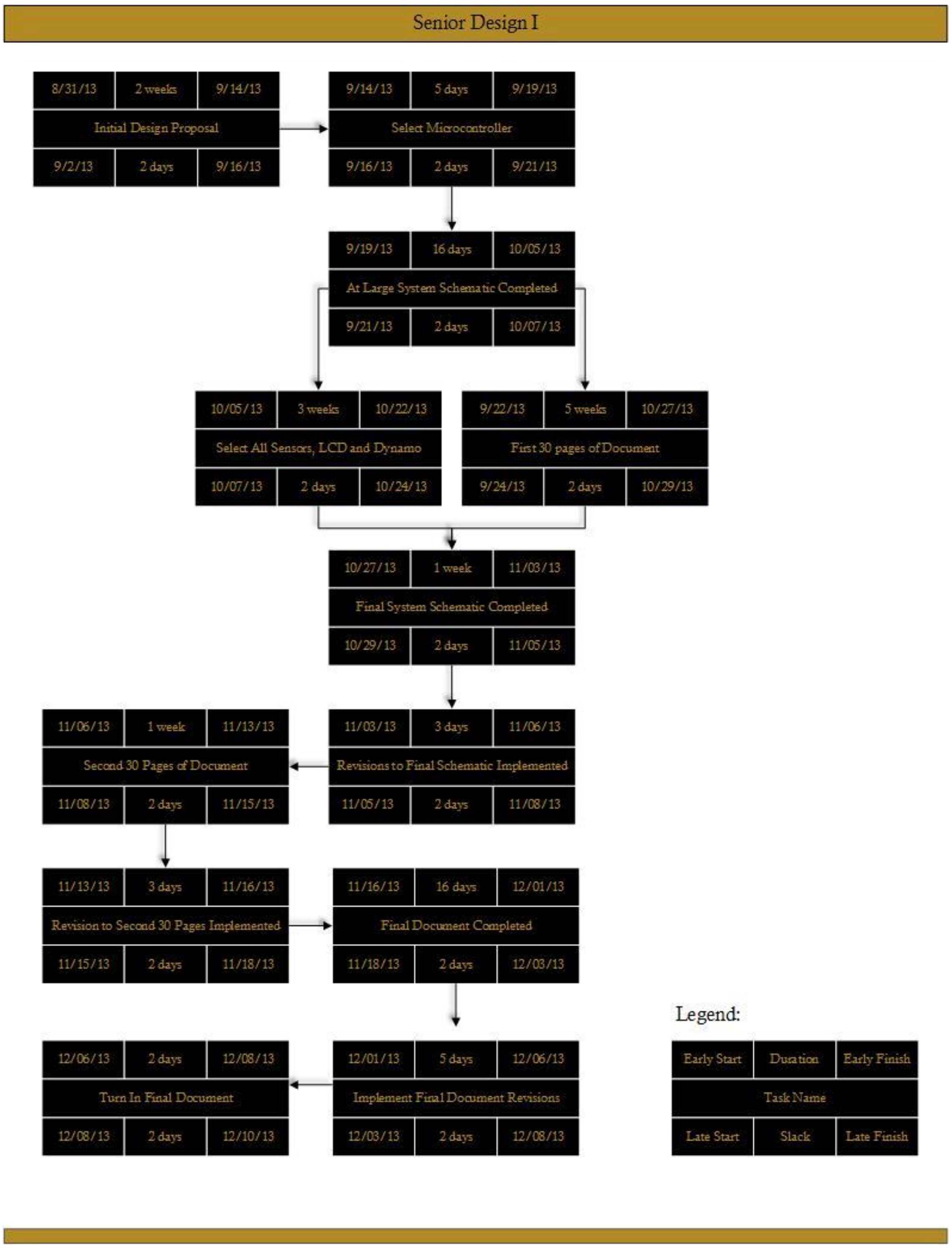

Senior Design I
8/31/13 | 2 weeks | 9/14/13
Initial Design Proposal
9/2/13 | 2 days | 9/16/13
9/14/13 | 5 days | 9/19/13
Select Microcontroller
9/16/13 | 2 days | 9/21/13
9/19/13 | 16 days | 10/05/13
At Large System Schematic Completed
9/21/13 | 2 days | 10/07/13
10/05/13 | 3 weeks | 10/22/13
Select All Sensors, LCD and Dynamo
10/07/13 | 2 days | 10/24/13
9/22/13 | 5 weeks | 10/27/13
First 30 pages of Document
9/24/13 | 2 days | 10/29/13
10/27/13 | 1 week | 11/03/13
Final System Schematic Completed
10/29/13 | 2 days | 11/05/13
11/03/13 | 3 days | 11/06/13
Revisions to Final Schematic Implemented
11/05/13 | 2 days | 11/08/13
11/06/13 | 1 week | 11/13/13
Second 30 Pages of Document
11/08/13 | 2 days | 11/15/13
11/13/13 | 3 days | 11/16/13
Revision to Second 30 Pages Implemented
11/15/13 | 2 days | 11/18/13
11/16/13 | 16 days | 12/01/13
Final Document Completed
11/18/13 | 2 days | 12/03/13
12/01/13 | 5 days | 12/06/13
Implement Final Document Revisions
12/03/13 | 2 days | 12/08/13
12/06/13 | 2 days | 12/08/13
Turn In Final Document
12/08/13 | 2 days | 12/10/13
Legend:
Early Start | Duration | Early Finish
Task Name
Late Start | Slack | Late Finish

*Figure 2 - PERT Chart for Senior Design One*

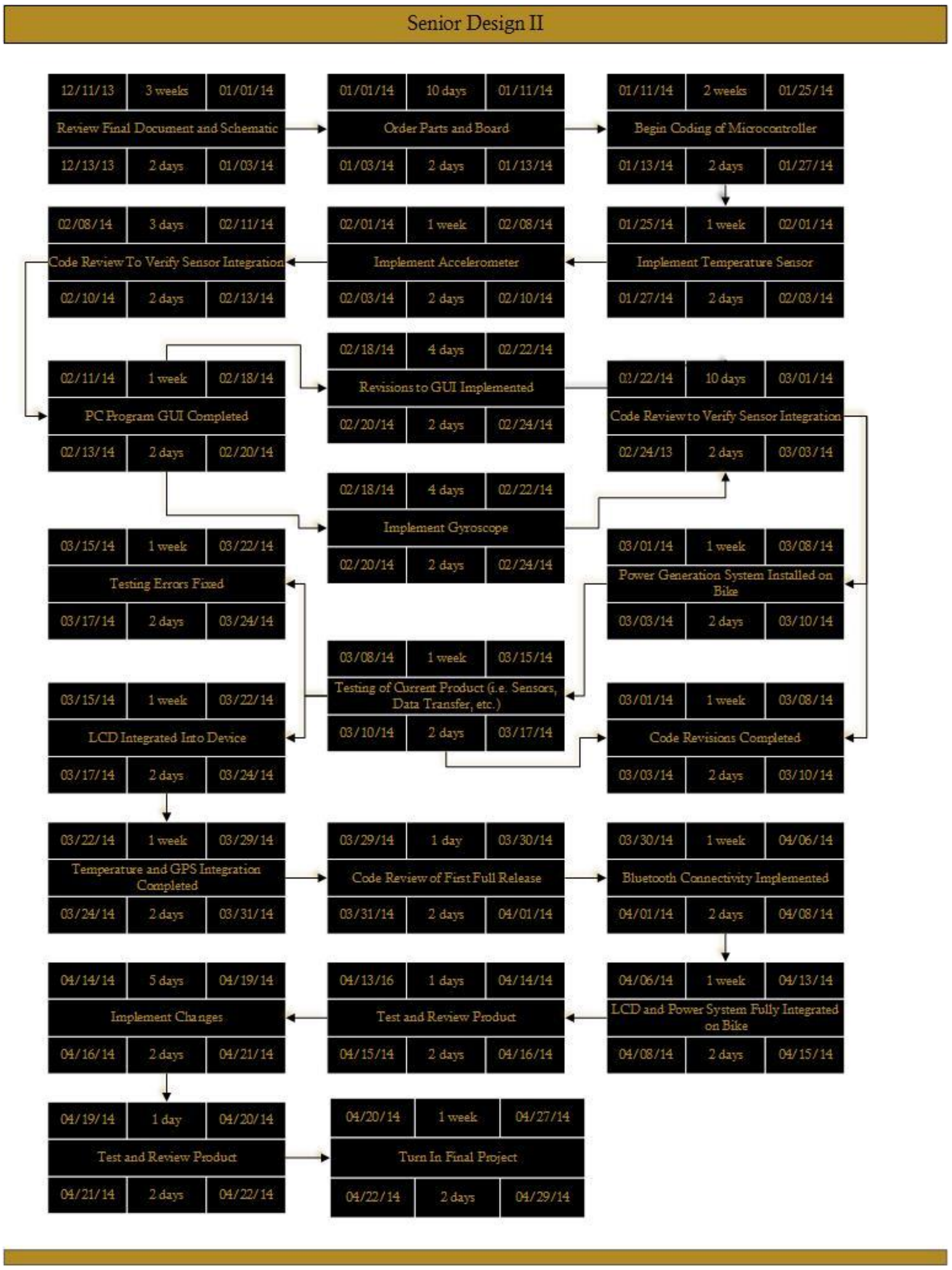


*Figure 3 - PERT Chart for Senior Design Two*

## 2.2 Team Organization

The team organization will be a critical part of this project, being organized will allow the group to move smoothly throughout the projects lifecycle. Creating a plan and staying organize throughout that plan will lead our group to a successful project at the end of Senior Design II. Using the tools described in the project management plan will simplify the entire project's lifecycle.

### 2.2.1 Group Dynamics

The design of the project will be distributed between two computer engineering majors. One of the group members will focus on choosing and implementing a microcontroller, a GPS receiver module, a Bluetooth module, an accelerometer and a temperature sensor. While the other focused on designing the Android application. The power system will be the responsibility of the two group members. Both will focus distributing the power created by the rider into a DC signal that is a regulated to a constant 3.3V in order to power the onboard components.

During the research phase of the project life cycle, the group will be communicating constantly through email, cellphones, following class time and through the use of the Dropbox created for the project. As a group we decided to meet three times a week after Senior Design lecture on Tuesdays and Thursday, then once again during the week on campus or at a previously agreed location. Dropbox provides a tool for us to keep research, images, datasheet and a to-do list to all be centrally stored in one place that can be accessed by all group members. Dropbox also provides 2 GB of memory, sufficient enough to allow the group to keep all previous versions of documents stored in the group's Dropbox.

Another tool that we found useful is Google+, which provides features that allow for online group discussions, merging of documents and even allow group member to simultaneously modify a document. Utilizing these tools will allow our individual group members to constantly stay in touch with each other with the ability to choose from different methods of communications.

### 2.2.2 Applicable Standards

Setting applicable standards when it comes to coding and documentation will simplified the process of merging document and java classes together to achieve on final design or document. The coding standard we will be using is the Oracle Code Conventions for the Java$^{TM}$ Programming language, which was revised on April 20, 1999. This style is well documented and easy to follow and can be found online (link on reference page [1]). Extensive use of comments and logical use of white space will be utilized during coding in order to make code simple and manageable. Specific variable names and consistent variable naming, for example (“thisIsAnExample”). For more information refer to the link provided in reference page located in Appendix D.

For the required documentation, all documents will be created using the following requirements that will aid in the “merging of documents” process. The documents will be created utilizing Microsoft word with Times New Roman font with a font size of 12. For the headings, we will utilize the same font but with a larger font size of 16 and the text will be bold. For the margins, all surroundings margins will be 1” around with the exception of the left margin. Which will be set to 1.5” to allow room for bindings, if needed. All documents pertaining to the project will be stored in the group Dropbox, while any modification made to documents will be documented in tabular form indicated the version number, the contributor, the date and comments and what revisions were made in that particular version.

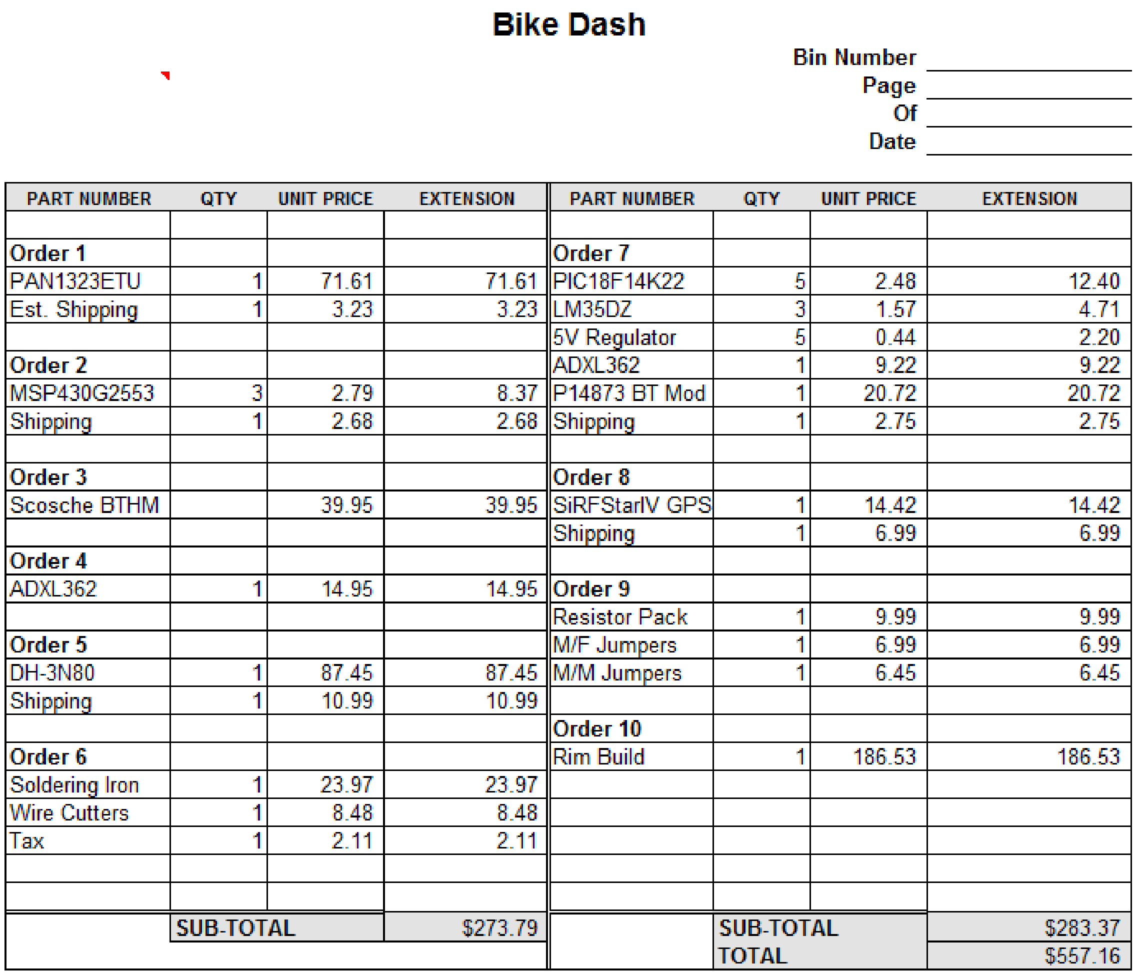
### 2.2.3 Tools and Computing Environment

After some research, the list below shows the tools and computing environments that will be utilized in the other to design and program the components of our project.

- **Operating System**: Windows 8 or Ubuntu 13.04
- **Programming languages**: Java
- **Compilers**: The Java Programming Language Compiler (javac), included in the Java Development Kit from Oracle Corporation, open-sourced since November 13, 2006.
- **IDE**: Android Developer Tools, Build: v22.0.5-757759. This product includes Eclipse Platform, JDT, CDT, EMF, GEF and WTP, all of which are Copyright © Eclipse contributors and others. Visit http://eclipse.org/
- Android Developer Tools are Copyright © The Android Open Source Project. Visit http://developer.android.com
- **Libraries**: Android 4.3 private libraries
- **Other**: Senior Design Lab
- **MCU IDE**: Energia release (0101E0012) – An IDE for the MSP430 Launchpad

## 2.3 Budget and Financing

The project proposal was submitted to Duke Energy, Inc. on September 16, 2013 in order to be considered as a candidate to receive financial support for the project. A response was received from Duke Energy, Inc. representatives, approving a budget of $570.35. Table 2.3 shows a list of actual expenses.



**Bike Dash**

Bin Number __________
Page __________
Of __________
Date __________

| PART NUMBER | QTY | UNIT PRICE | EXTENSION | PART NUMBER | QTY | UNIT PRICE | EXTENSION |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| **Order 1** | | | | **Order 7** | | | |
| PAN1323ETU | 1 | 71.61 | 71.61 | PIC18F14K22 | 5 | 2.48 | 12.40 |
| Est. Shipping | 1 | 3.23 | 3.23 | LM35DZ | 3 | 1.57 | 4.71 |
| | | | | 5V Regulator | 5 | 0.44 | 2.20 |
| **Order 2** | | | | ADXL362 | 1 | 9.22 | 9.22 |
| MSP430G2553 | 3 | 2.79 | 8.37 | P14873 BT Mod | 1 | 20.72 | 20.72 |
| Shipping | 1 | 2.68 | 2.68 | Shipping | 1 | 2.75 | 2.75 |
| | | | | | | | |
| **Order 3** | | | | **Order 8** | | | |
| Scosche BTHM | | 39.95 | 39.95 | SiRFStarIV GPS | 1 | 14.42 | 14.42 |
| | | | | Shipping | 1 | 6.99 | 6.99 |
| **Order 4** | | | | | | | |
| ADXL362 | 1 | 14.95 | 14.95 | **Order 9** | | | |
| | | | | Resistor Pack | 1 | 9.99 | 9.99 |
| **Order 5** | | | | M/F Jumpers | 1 | 6.99 | 6.99 |
| DH-3N80 | 1 | 87.45 | 87.45 | M/M Jumpers | 1 | 6.45 | 6.45 |
| Shipping | 1 | 10.99 | 10.99 | | | | |
| | | | | **Order 10** | | | |
| **Order 6** | | | | Rim Build | 1 | 186.53 | 186.53 |
| Soldering Iron | 1 | 23.97 | 23.97 | | | | |
| Wire Cutters | 1 | 8.48 | 8.48 | | | | |
| Tax | 1 | 2.11 | 2.11 | | | | |
| | | | | | | | |
| | | | | | | | |
| | **SUB-TOTAL** | | $273.79 | | **SUB-TOTAL** | | $283.37 |
| | | | | | **TOTAL** | | $557.16 |

*Figure 4 - Bike Dash Cost*

## 2.4 Integrity Testing

In order to verify the integrity of our components, we had to devise and adhere to a strict test plan that would assure us moving forward that all of our sensors and data acquisition devices operated as expected and according to the manufacturer's datasheet. The integrity testing procedures must be comprehensive and include testing all individual functions however, these tests are simply to ensure we are receiving accurate results. We eventually must test the system at large to ensure that not only is each component functioning independently, but that all components are functioning as expected and designed together as a whole. Integration testing and system testing will be accounted for in the Testing section of this report, Section 6.0.

The first step in organizing our integrity testing plan is to determine individual features, or feature sets, outlined by the manufacturer to test basic operational functionality. This will help to keep our testing more organized as well as allow for the implementation of specialized test scenarios by building on the tests created in this section. While devising test cases, it is also important to assign each test case a priority and impact. This well help to ensure that all major system functions are operable soonest. For our integrity testing we will consider all tests to be of high priority and high impact. In the following sections (2.4.1 through 2.4.5) we will outline our planned tests. For a complete set of test procedures (including system and integration test procedures) and expected results please refer to the latter section of this report title "Testing" (Section 6.0).

### 2.4.1 Microcontroller Testing

The first device we will subject to our integrity testing will be our microcontroller. During integrity testing for our microcontroller, we will test such standards as data rate, data integrity, environmental testing, as well as a few other select tests. In order for us to verify our MCU integrity, we must outline a set of key tests for our microcontroller to ensure that at a very basic level our microcontroller will operate as expected and as outlined by the manufacturer. The tests we have devised can be seen below in Table 4. For a comprehensive list of testing procedures (including system and integration testing) please see our Testing section, Section 6.0.

| | Microcontroller |
|---|---|
| **Test Objective** | Determine if input is received |
| **Test Description** | In this test, we will determine if the microcontroller is functioning as intended and receiving input from all of the sensors. We will test each sensor by calibrating and observing the input at the microcontroller pin and verifying it with the parsed data. |
| **Test Objective** | Stress test |
| | In this test, we will allow the system to run for an extended period of time to exceed normal operating times. We will observe temperature, speed, and reliability. |
| **Test Objective** | Environment testing |
| **Test Description** | In our environment testing segment, we will run our microprocessor in conditions outside of its "normal" operating range. We will run it for an extended duration in rain, cold, and heat as well as through dirt and possibly mud. |

*Table 3 - Microcontroller Integrity Test Procedures*

### 2.4.2 Temperature Sensor Testing

The next device that we will test will be our LM35 temperature sensor. With our temperature sensor, there are few claims we must investigate in order to ensure proper operation as this sensor only has one main function. In order to test this temperature sensor and ensure that it will act as desired during operation, we have devised a few simple tests as outlined below in Table 5. Comprehensive testing procedures can be found in the Testing section (Section 6.0).

| Temperature Sensor | |
|---|---|
| **Test Objective** | Temperature verification |
| **Test Description** | To verify that we are receiving an appropriate temperature readout, we will place our sensor in a known state (ex. A glass of ice water) and measure the temperature with an analog thermometer as well. We will also measure the voltage at this temperature to verify the correct output of the sensor as well. |
| **Test Objective** | Stress Test |
| **Test Description** | To test the integrity of our temperature sensor we will test various conditions to put abnormal stress on the sensor. We will begin by exposing the sensor to high heat and successively low temperatures for an extended period. We will also impose rapid temperature changes over an extended duration to further test the integrity of our sensor. |

*Table 4 - Temperature Sensor Integrity Test Procedures*

## 2.4.3 Accelerometer Testing

The third device that we will pass through integrity testing is our accelerometer. This device is responsible for measuring the acceleration in the coordinate plane system (x, y, and z) and must perform as stated by the manufacturer. To ensure the integrity of our device, we test the zeroing ability as well as the ability to acquire proper acceleration readings. The tests we will run to ensure the device operates as expected are outlined below in Table 6 For a comprehensive set of testing procedures please see the Testing section (Section 6.0) of this document.

| Accelerometer | |
|---|---|
| **Test Objective** | Zeroing |
| **Test Description** | In the zeroing integrity test we will determine whether or not the accelerometer is accurate reading 0 when not in motion. |
| **Test Objective** | Acquisition Test |
| **Test Description** | For our acquisition test we will record and validate free-fall movement experienced by our sensor. We will compare our recorded results with an expected acceleration of 9.81 $m/s^2$ |

*Table 5 - Accelerometer Integrity Test Procedures*

### 2.4.4 Bluetooth Module Testing

For our Bluetooth integrity testing, we will incorporate tests to verify data transmitted between devices. We will test the data rate and distance at which our Bluetooth module can operate effective and consistently. We will also test the susceptibility of our device to external interference. The integrity tests developed for this application can be seen below in Table 7. For a set of comprehensive test procedures, please see the Testing section located in Section 6.0.

| **Bluetooth Module** | |
|---|---|
| **Test Objective** | Tx and Rx |
| **Test Description** | For our transmitting and receiving test segment, we will simply test the error rate, speed, and reliability by transmitting and receiving small data packets. |
| **Test Objective** | Load testing |
| **Test Description** | In the load testing segment, we will test both the number of devices the Bluetooth module can handle as well as at what bandwidth the module will Tx/Rx while at full capacity or max load. |
| **Test Objective** | Interference testing |
| **Test Description** | In the interference testing section, we will test the ability of the Bluetooth module to resist interference and transmit our data packets successfully. |
| **Test Objective** | Distance Testing |
| **Test Description** | Though our system does not require an extended range to operate, we will still test the levels of signal strength at various distances. |

*Table 6 - Bluetooth Module Integrity Test Procedures*

### 2.4.5 Dynamo Testing

We will continue our integrity testing by next analyzing our dynamo. In order to verify our dynamo operates according to the manufacturer's specification, we will test both power output as well as stability of the system under high stress and under variable environmental conditions. The tests we will run on our dynamo to verify its integrity are outline below in Table 8. For a comprehensive set of testing procedures, please refer to the Testing section (Section 6.0).

| **Dynamo** | |
|---|---|
| **Test Objective** | Power Output |
| **Test Description** | In the power output test, we will test to ensure that the power generated by the dynamo is comparative to our expected results. We will consider both RMS and Peak performance. |
| **Test Objective** | Environmental Conditions Test |
| **Test Description** | In the environmental conditions test, we will run the dynamo in inclement weather conditions and under high load conditions to determine the integrity of the dynamo. |
| **Test Objective** | Stress Test |
| **Test Description** | For our dynamo stress test, we will place the system under a high load and measure both the output voltage as well as the stability of the signal. |

*Table 7 - Dynamo Integrity Test Procedures*

### 2.4.6 GPS Module Testing

For our last integrity test, we will be testing the GPS module. In order for our system to provide accurate results we must ensure that our positioning system is functioning as expected as it provides our system a large part of its functionality. In short, we need to make sure that our GPS will not only track accurately, but will track quickly and in high traffic areas where interference is expected. We will also test our devices ability to operate in sparsely occupied areas. The integrity tests we have developed for this device can be found below in Table 9. For a comprehensive set of tests, please see the Testing section of this document (Section 6.0).

| GPS Module | |
|---|---|
| **Test Objective** | Triangulation |
| **Test Description** | In this test, we will testing the acquisition of at least A3 satellite connection. We will test Time To Link (TTL) as well as the reliability and ability to consistently connect to at least three satellites. |
| **Test Objective** | Density Testing |
| **Test Description** | In our density test, we will be testing the ability of our device to operate in areas of dense vegetation, buildings, people, or other obstructions. We will test most likely during highly populated events. |
| **Test Objective** | Sparseness Testing |
| **Test Description** | In our test for sparseness, we will be testing to determine the integrity of our connection in areas of low population and where the probability of a connection is much lower. |

*Table 8 - GPS Module Integrity Test Procedures*

## 3.0 Research

## 3.1 Microcontroller

To begin our research, we must start with one of the most critical components in our system, the microcontroller (MCU). In this project, the microcontroller will receive data from all of the included sensors in real-time, manipulate the data, and then send the data as output data via Bluetooth to the user's mobile device. The MCU will be responsible for providing data both accurately and efficiently and must do so under stressful environmental conditions. The use of a real-time system design requires us to think carefully about the system at large and to make very precise decisions about our MCU. A basic block diagram of our system interfacing with the microcontroller can be seen below in Figure 5.

When considering prospective MCUs there are a few options that we must consider heavily. First and foremost is the throughput and data rate of our MCUs. This will dictate our MCU's ability to process data and will govern the speed of our system as a whole. Though it is intended to use an 8-bit MCU in our application, the use of a 16-bit MCU is not out of the question. The decision will be made, however, based on economics and necessity. While it is always good to have additional processing power, there is generally a financial and power cost associated with it. Also, the use of 16-bit addressing might be much more than what we need for our application. We will discuss this further when we compare microcontrollers in the latter part of this section.

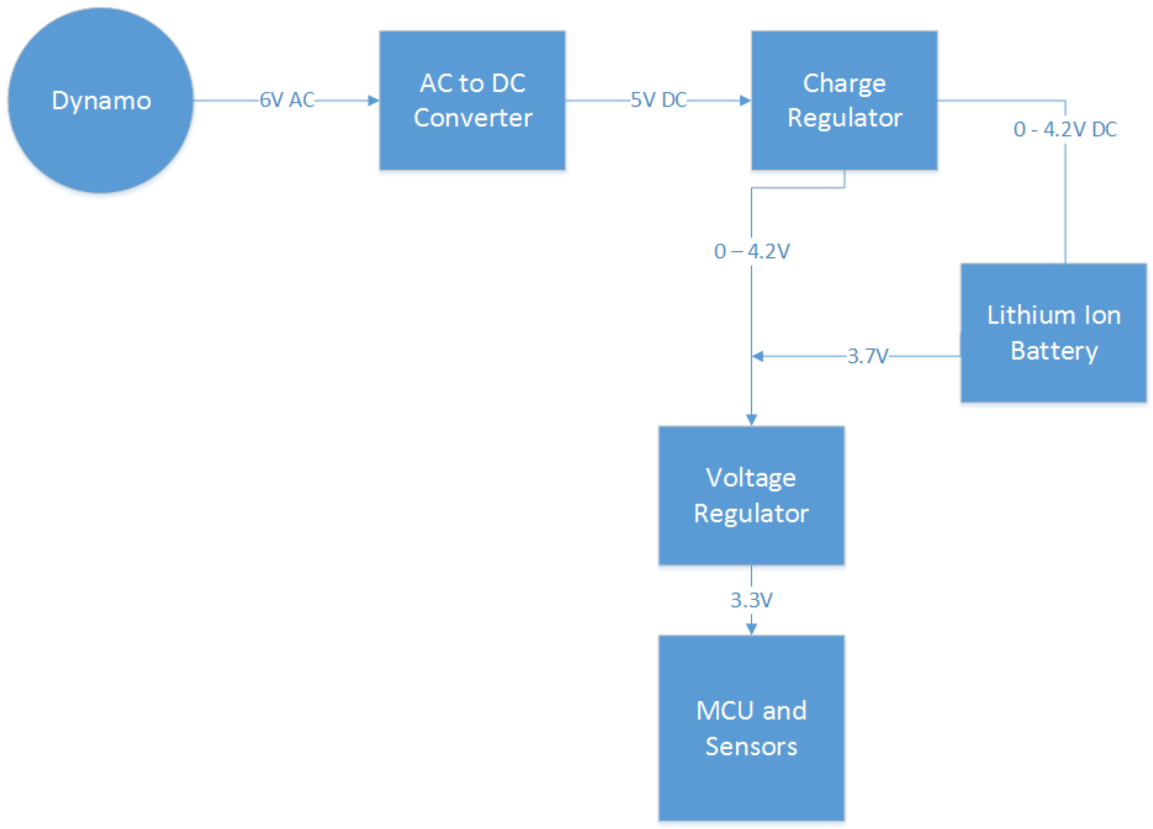


*Figure 5 - Basic System Block Diagram*

Another deciding factor when choosing our MCU is considering the number of needed GPIO (General Purpose Input/Output) pins. To determine the number of GPIO pins, we will consider the inclusion of up to 6 sensors for input as well as the need for ADC/DAC channels, PWM (Pulse Width Modulation), and the possibility of alternative communication channels. Lastly we will consider the memory constraints of our application by estimating code size and looking at RAM requirements. Ultimately our decision will be based heavily upon necessity and availability, but will include some decisions based on economics and personal preference.

### 3.1.1 Microcontroller Comparisons

For our microcontroller selection process, we will focus on two companies and compare similar MCUs. When we picked these two companies for prospective MCUs we tried to narrow down our pre-selection to competitors that would offer sizeable products at a reasonable cost. With the microcontroller being by far the most important part of our product it was imperative that we use only well-known and reputable companies. For this reason, we have only selected to comparative companies but feel that these two companies exemplify the ideal capabilities that the scope of our project needs to be both effective and successful. The two families of MCUs that we have chosen will be Atmel’s 8-bit AVR series and Microchip's 8-bit PIC series. The first MCU we will investigate is Atmel's ATmegaAVR 8-bit microcontroller, the ATmega48 (Figure 6).

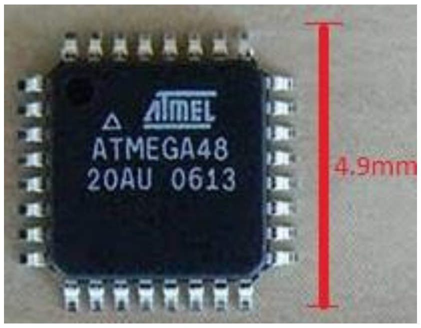


Figure 6 - ATMEGA48/20AU

The ATmega48 is a mid-range microcontroller offered under Atmel's 8-bit AVR product line. The megaAVR series aims to market an economical MCU that attempts to provide users with a low power solution, running at approximately 10 MIPS at 1.8v on the low end and can handle up to 20 MIPS with a 5.5v peak on the high end. We would intend to run the ATmega48 in the range of 2.7v to 5.5v due to our desire to achieve the optimal performance provided. Running out Atmega48 at 5.5 V would allow us to achieve the full 20 MIPS offered. Though we may not need the entirety of our bandwidth, it never hurts to have the ability to ramp up bandwidth rather than to replace an MCU and rewrite an architecture.

To compare, we look at Microchip's PIC18F series MCU as an alternative 8-bit option (Figure 3.2 – PIC18F14K22). The PIC18 MCU is Microchip's midrange MCU and attempts to offer an economical MCU comparable to the ATMEL AVR series microcontrollers. A key difference when comparing the two microcontrollers is the change in processing power between the two. The ATmega48 offers up to 20MIPS at a max voltage of 5.5v whereas the PIC18F is only able to offer up to 16MIPS with the same max voltage at 5.5v. Though this difference may not seem to be significant, every little bit can matter in a real-time system. The ability to process more instructions per second is always a valuable trait and usually means better processing of real-time data, but almost always comes with a power cost associated with it. Though the PIC microcontroller is only able to process at 16 MHz (1 MIPS per 1 MHz), if we employ a phase locked loop, or PLL, we are able to achieve a max processing power of up to 64 MHz from our device.

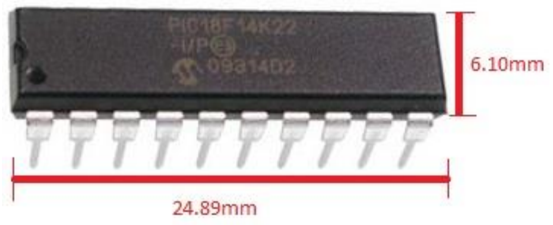


Figure 7 - PIC18F14K22

To continue our comparison we look at both the type, and amount of memory available on our selected MCUs. With memory, there are three types of memory that we are particularly interested in comparing. The three types of memory that we would like to compare are EEPROM, RAM and program memory. The EEPROM (or Electrically Erasable Programmable Read-Only Memory) memory is considered to be non-volatile meaning that data will remain on the device even after the power supply has been removed. This memory will be responsible for containing calibration data and tables for onboard devices that must access this data upon startup. The RAM (Random Access Memory) is considered to be a volatile memory type and will contain data being sent, received, or manipulated. The third memory type that we will investigate is the program memory. The program memory will contain the bulk of our data and will be responsible for housing the entirety of our MCU's system architecture. Though we do not intend for our code size to be very large, having extra storage space will allow for both improvements and a higher level of code modularity. While continuing to look at the PIC18F14K22, we see that the total Program Memory is 16 KB and also has a total of 256 bytes of Data EEPROM and 512 bytes of RAM. We compare this to the ATmega48 and notice that both the Data EEPROM and the RAM have the same memory capacity as the PIC18F microcontroller; the key difference, however, lies in the total capacity for the Program Memory. In the ATmega48 we have a total of 4KB of Flash memory whereas the PIC18F contains 16KB of memory. This is a significant difference in memory capacity and will heavily influence our decision.

Another important comparison to make when considering multiple microcontrollers is the availability of peripherals and GPIOs. We first look at the options offered by the ATmega48 microcontroller and the pins offered. We see that the ATmega48 offers a maximum of 23 input/output channels. The ATmega48 offers 8 channels of 10-bit ADC, 6 channels of PWM, and three timers (One 16-bit and Two 8-bit). Comparing the PIC18F14K22, we see that the maximum number of input/output pins is maxed at 25, with one pin reserved for input only. However, comparing further we see that the PIC18F offers 12 channels of ADC at 10-bits as well as four timers (3 16-bit and 1 8-bit). A minor drawback is the decrease in PWM channels at only 4 channels compared to 6 with the ATmega48.

### 3.1.2 Microcontroller Final Selection

There are many factors involved in choosing an MCU that led to our final selection of a microcontroller. While making our decision, we considered the following key aspects of our microcontrollers:

- Data Rate – Instructions per second and speed (in Hertz)
- Memory – Amount of RAM, EEPROM, and program memory
- Pin Outs – The number and type of pin outs available
- Size – The physical dimensions of the MCU
- Cost – The unit cost of the microcontroller and cost per MHz

After looking at all aspects of two closely matched microcontrollers, we have come to a final decision. The PIC18 has a higher capacity for program memory, meaning we will be able to house more instructions and more information for our sensors. Though we will do our best to ensure efficiency in our code, we can only estimate our code size and instruction set and thus going with an option that provides for a much higher memory capacity is in the projects best interest. Another deciding factor in our microcontroller selection process was the number and type of input/output ports available with our microcontroller. The ATmega48 MCU only contained 23 usable ports whereas the PIC18 contained up to 25 GPIO pins. The availability of more ADC pins on the PIC18 also gave further justification behind choosing this microcontroller. The pin outs and wiring diagram will be presented later in the design phase of this document. Lastly in our microcontroller comparison we have investigated both total cost and cost per MHz, as well as physical size. First we look at the overall cost of each of our microcontrollers. The ATmega48-20AU is set at a price of $2.79 per unit whereas the PIC18F14K22 is set at a price of $2.48 per unit. Though this cost difference seems negligible at one unit, the cost difference at 1000 units is already over $300.00. We do not intend on needing that many units, but the same engineering principles stand with one unit, or 1000 units. The effect of this cost difference is most evident when we look at cost per MHz on each unit. To do this, we simply divide the cost by the number of MHz available. For the ATmega48-20AU we can see that the cost per MHz is seen below in equation 3.1, which seems minimal until you compare the cost per MHz to the PIC18F1K22 which is seen in equation 3.2.

$$\frac{\$2.79}{20} \approx \$0.14/MHz$$

Equation 3.1 – Atmega48-20AU cost per MHz

$$\frac{\$2.48}{64} \approx \$0.04/MHz$$

Equation 3.2 – PIC18K1422F cost per MHz

From these two equations we can see that clearly the PIC microcontroller is more cost effective in terms of cost per megahertz. The next comparison we made was the physical size of both of our microcontrollers as well as the aesthetic structure of the MCU. When we look at the PIC18 microcontroller and compare it to the ATmega48 microcontroller, we can clearly see that the PIC18 has a greater surface area than the ATmega48. This, however, is not the only physical metric we use to determine suitability for our project. The next metric that we used was the pin layout of each of our devices. In the ATmega48 microcontroller (Figure 3.1), the pins are located on all four symmetric sides of the device. The PIC18 (Figure 3.2) has the pins located on the two long edges of the device. Though neither setup seems beneficial, we prefer the pin structure of the PIC18 over the ATmega48 simply because the PIC18 contains pins that are perpendicular to the surface making for easy install and removal during the prototyping stage of our design. The ATmega48 has its pins bent at the bottom, meaning we would have to physical modify, or deform the device in order for it to suit our initial prototyping application. For reference, see the Table below (Table 3.3) for a side by side comparison.

|  | ATmega48-20AU | PIC18F14K22 |
|---|---|---|
| Instructions Per Second | 20 MIPS | 16 MIPS |
| Memory (kilobytes) | 4 KB | 16 KB |
| Speed (in Hertz) | 32 kHz (Low Power) / 20 MHz (Max) | 31 kHz (Low Power) / 16 MHz (Max) / 64 MHz (PLL) |
| Pins | up to 23 | up to 25 |
| RAM (bytes) | 512 bytes | 512 bytes |
| EEPROM (bytes) | 256 bytes | 256 bytes |
| Power Usage (volts) | 1.8V (Low Power) / 5.5V (max) | 1.8V (Low Power) / 5.5V (max) |
| Size | 4.90 mm x 4.90 mm (LxW) | 24.89 mm x 6.10 mm (LxW) |
| Cost | $2.79 | $2.48 |
| Cost/MHz | ~ $0.14 | ~$0.04 |
| Safe Operating Temperature | -40ºC to approx. 85 ºC | -40ºC to approx. 85 ºC |

*Table 9 - Comparison of Prospective Microprocessors*

UPDATE:

For our final design we have decided to use the MSP430G2553 as our preferred microcontroller. This is due to both the experience with the MSP430 of the Bike Dash team, but also the availability of resources. Characteristics of the MSP430G2553 can be seen below in Table 10.

| MSP430G2553 |  |
|---|---|
| Operating Voltage | 3.3V |
| Speed | 1 MHz |
| RAM | 512 bytes |
| Memory | 16 KB |
| Pins | 20 |

*Table 10 - MSP430G2553 Characteristics*

## 3.2 GPS Module

To continue our research, we move on to another important component in our system, the Global Positioning System (GPS) module. In this project, the GPS[6] module will calculate the rider's precise three-dimensional location in order to determine the distance travelled during the rider's bike ride while also allowing the rider to utilize the product for navigation purposes by providing accurate timing information in real-time. Our GPS module will receive signals from the Global Navigation Satellite System (GNSS), currently funded and operated by the US Department of Defense (DoD) that contains information of the satellites position and the time that signals were transmitted. That information will then be inserted into the navigation equations embedded into the GPS module to determine the precise time and the velocity (direction and speed) and an extremely accurate three-dimensional location consisting of the precise altitude, longitude, and latitude of the rider. All the information will then be passed to the microprocessor through the UART serial communication.

In order to achieve accuracy the GPS module must be visible by at least 4 or more satellites, as the visibility of satellites increases the accuracy error will decrease. In order to achieve a reasonable "Time-to-First-Fix" our GPS module must have a large number of channels with the ability to perform simultaneously tracking and calculations in order to achieve a quick first "lock". Since we are utilizing the rider's own momentum to provide power to the complete system, power consumption along with accuracy are one of the most important features when it comes to deciding on a GPS receiver module. Our research will be focused on comparing three prospective GPS receiver module that focused on accuracy, a reasonable number of channels and most importantly power consumption. The three GPS modules we will be considering and comparing will be the A2235-H a GPS receiver module from Maestro Wireless Solutions, Ltd., the GMS-HPR GPS receiver from GlobalTop Technology, Inc. and the Venus638FLPx-L.

To further analyze the selection process of our three top prospective GPS receiver module we will look at features such as update rate, antenna type and economic cost. For our project size will be considered but will not contribute to final selection as all prospective receiver are of similar size. In terms of the update rate of the GPS receiver module, an update rate of 1Hz may be sufficient enough for our system but since some riders and bikes can reach speeds of up to 25-40 mph, an update of around 5-10 Hz is more desirable. We will then compare the type of antenna each receiver module possesses. Each antenna is finely tuned to pick up the GPS LI frequency of 1.57542 GHz, the frequency assigned by the DoD for civilian use. Certain type of antennas increase accuracy in heavily forested areas and urban cities with tall building, which is something that must considered in order to provide an efficient and accurate product for rider's that would like to use our product in cities like New York or an off-road biking experience. Finally we will also take into consideration the economic cost of our GPS receiver module in order to produce a final product that's accurate, power efficient and affordable.

## 3.2.1 GPS Module Comparisons

The first GPS receiver module we will investigate is the A2235-H, Maestro's newest active GPS receiver module that features a stack-up antenna SiRFstarIV integrated solution. The A2235-H is a versatile GPS receiver that combines a SMT-based integrated GPS high patch antenna to achieve high levels of sensitivity and position accuracy with the high performance of a SiRFStarIV GPS engine and its features. Maestro's wireless SiRFstarIV GPS receiver modules provides engineers with a product that offers optimal "Time-to-First-Fix", ultra-low power consumption and ease of integration to allow minimal assembly cost. While the on-board patch antenna allows the GPS receiver to perform at high levels of accuracy and sensitivity in extreme conditions and it's suitably assembled above the component decreasing its size. Figure 8 presents the dimensions of the A2235-H GPS module.

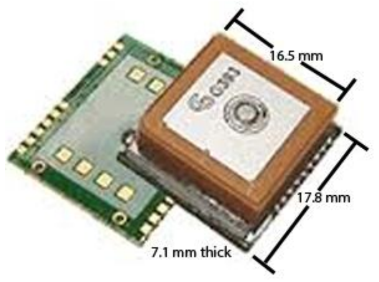


*Figure 8 - A2235-H*

The A2235-H contains 48 channels working at the L1 frequency of 1.575 MHz that allow the module to achieve a "hot start" of less than 1 second and a cold start of just under 35 seconds. The A2235-H performs at high levels with respect to sensitivity by employing its Jammer remover technology which removes up to 8 in-band jammer to achieve levels of -163 dBm while being utilized for tracking purposes and -160 dBm while being utilized for navigation purposes. It also performs well when it comes to accuracy with the use of SiRFStarIV technology it can pin point location down to less than 2.5 m circular error probability (CEP). Achieving high performance while maintaining low power consumption with a supply voltage of 3.0 to 3.6 V and an average current draw of 40 mA while searching and 29mA while tracking.

After covering the performance of our first prospective GPS receiver module we investigated the ability to perform in different environments, its dimensions and the ability to communicate accurately with the chosen microprocessor. We found that the A2235-H is compatible with any microprocessor that utilizes UART serial communication with a baud rate between 1,200 to 115.e k and clock rate of up to 6.8 MHz and it also possesses the ability to withstand varying temperatures while being small in dimensions and lightweight. Other technical details that could be useful in the selection process of our GPS receiver module are highlighted in Table 3.8, including performance details, communication, power, environment and dimensions.

Next we investigate the GMS-HPR, Global Top Technology, Inc. latest GPS receiver module that also features an integrated patch-on-top antenna much like the A2235-H. Along with being compact in size it also shares the features of achieving optimal TTFF, performing at a high level with respect to sensitivity and accuracy while focusing on ultra-low power consumption. The dimensions of the GMS-HPR are presented in Figure 9.

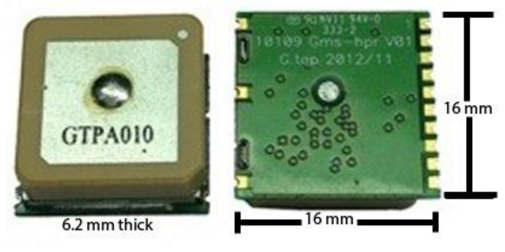


Figure 9 - GMS-HPR

The GMS-HPR GPS receiver module integrates MediaTek's MT3339 Chipset containing 66 channels that could possibly track up to 66 GPS satellites at one time.  This amount of channels allow the module to achieve a "hot start" of less than 1 second and a "cold start" of just under 35 seconds.  The GMS-HPR features a 12 multi-tone active interference canceller along with the MT3339 Chipset to achieve a high sensitivity of -165 dBm.  It also performs well when it comes to accuracy achieving a position accuracy of less than 2.5 m CEP.  The GMS-HPR also fits into the ultra-low power consumption category since it performs at a power supply of 3.0 – 4.3 V and only consuming 20 mA of the current while tracking.

Then we investigated its dimensions, its ability to perform in different environments and the ability to communicate with our prospective microprocessor.  We found that the GMS-HPR GPS receiver module performs well in varying temperatures conditions and utilizes the on-patch antenna technology in order to achieve optimal size.  The GMS-HPR receiver is able to communicate with our microprocessor as it employs UART (TTL) serial communication with a baud rate between 4,800 and 115.2k bps.  Other technical details that could be useful in the selection process of our GPS receiver module are highlighted in Table 3.8, including performance details, communication, power, and environment.

Finally we moved on the Venus638FLPx-L, from SkyTraq Technology, Inc.  A high performance, low cost GPS receiver module targeting the mobile market.  The lightweight receiver it's as small dime in dimensions while packing the capability to achieve optimal TTFF performance and exceptional accuracy for its size.  The dimensions of the Venus638FLPx-L are presented in Figure 10.

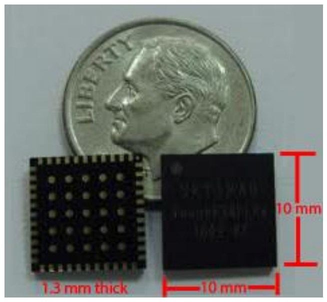


*Figure 10 - Venus 638FLPx-L*

The Venus638FLPx-L GPS receiver module has the same capabilities as the two previous receiver described when it comes to accuracy and sensitivity. Achieving an accuracy of 2.5 m CEP and a sensitivity of -165dm while tracking. The Venus638FLPx-L achieves even lower power consumption than the A2235-H and the GMS-HPR, consuming between 18mA – 29 mA and only need a supply voltage between 2.8 V – 3.6 V to work properly. After some research, the main downfall of this particular GPS receiver module is the absence of a built-in or attached antenna for increased accuracy in extremely low reception areas. Other technical details that could be useful in the selection process of our GPS receiver module are highlighted in Table 11, including performance details, communication, power, environment and dimensions.

### 3.2.2 GPS Module Final Selection

There are many factors involved in choosing a GPS receiver module that led to our final selection. After looking at three key aspects of three closely matched GPS receivers, we have come to a final decision. The GMS-HPR has a total of 66 channels with an update rate range of 1 to 10 Hz that can simultaneously analyze signals received from the satellites that make up the GNSS, which allows the GPS receiver to achieve the typical standard in the industry for TTFF. Another deciding factor is the fact that the GMS-HPR has an on-patch antenna built-in which allows it to outperform the Venus638FLPx-L in extremely low reception conditions. The built-in antenna will save our team cost and time by not having to purchase and integrate an external antenna needed to achieve same level of accuracy performance with the Venus 638FLPx-L.

That then brings us to two potential GPS receiver modules, the main factor that set this two candidates apart were the difference in power consumption. The A2235-H is inferior to the GMS-HPR when it comes to power consumption, consuming a current of 40 mA while searching and consuming a current of 29 mA while tracking. The GMS-HPR also has a higher update rate needed to achieve accuracy during high speed cycling, with the ability to be modified from 1 to 10 Hz. Those factors mentioned along with the fact that the GMPS-HPR GPS receiver module is smaller in dimensions and petite, allow us to overlook the $6.00 cost difference when compared to the A2235-H. Below is a table that supports the decisions made in the final selection of our GPS receiver module.

| | A2235-H | GMS-HPR | Venus638FLPx-L |
|---|---|---|---|
| **Performance** | | | |
| **Channels** | 48 | 66 | 65 |
| **Sensitivity (Tracking)** | -163 dBm | -165 dBm | -165 dBm |
| **Sensitivity (Navigation)** | -160 dBm | -162 dBm | -162 dBm |
| **Position Accuracy** | < 2.5m CEP | < 2.5 m CEP | < 2.5 m CEP |
| **TTFF Hot Start** | < 1 s | < 1 s | < 1 s |
| **TTFF Cold Start** | < 35 s | < 33 s | < 29 s |
| **Update Rate** | 1 – 6 Hz | 1 – 10 Hz | 1 – 20 Hz |
| **Communication** | | | |
| **Baud Rate Range** | 1,200 – 115.2 k | 4,800 – 115.2k | 4,800 – 115.2k |
| **Protocols** | NMEA, SiRF | NMEA 0183 MTK Command | NMEA 0183 SkyTraq Binary |
| **Ports** | Tx (input) Rx (output) | Tx (input) Rx (output) | Tx (input) Rx (output) |
| **Power** | | | |
| **Supply voltage** | 3.0 – 3.6 V DC | 3.0 – 4.3 V | 2.8 – 3.6 V |
| **Current Draw-acquisition** | 40 mA | 25 mA | 29 mA |
| **Current Draw-tracking** | 29 mA | 20 mA | 11 – 18 mA |
| **Power Consumption-acquisition** | 100 mW | 82 mW | 88 mW |
| **Power Consumption-tracking** | 86 mW | 66 mW | 60 mW |
| **Trickle Power Mode(1 Hz)** | 4.1 mA | 3.7 mA | 2.5 mA |
| **Environment** | | | |
| **Operating Temperature** | -40ºC to +85ºC | -40ºC to +85ºC | -40ºC to +85ºC |
| **Humidity** | Non-condensing | Non-condensing | Non-condensing |
| **Dimensions** | | | |
| **L x W x H (mm$^3$)** | 17.8 x 16.5 x 7.1 | 16 x 16 x 6.2 | 10 x 10 x 1.3 |
| **Weight** | 4.0 g / 0.14 oz. | 6 g | 1.8 g |
| **Cost** | $14.42 | $20 | $39.95 + antenna cost |

*Table 11 -Technical Details of Prospective GPS Modules*

UPDATE:

For our final project, USGlobalSat's EM-506 GPS unit was used. This unit was used for its ease of integration as well as the autonomous nature of the embedded software. This GPS unit provides similar accuracy as the other GPS modules, and offers an onboard antenna as well. This GPS unit also offers a 6-pin jumper that allows for quick disconnects. Characteristics of the EM-506 can be seen in the table below.

| EM-506 | |
|---|---|
| Supply Voltage | 3.3V |
| Channels | 48 |
| Time to First Fix (TTFF) | < 35s (Cold); < 1s (Hot) |
| Update Rate | 1 Hz |
| Position Accuracy | < 2.5m |
| Velocity Accuracy | < 0.01 m/s (Speed);<br>< 0.01 degrees(Heading) |

*Table 12 - EM-506 Characteristics*

## 3.3 Bluetooth Module

Another feature that's necessary for our device is the Bluetooth module which will give our device the ability to utilize Bluetooth technology in order to communicate the information calculated by the microcontroller to the Android device. The data that is sent from the GPS receiver module, the temperature sensor, and the accelerometer will be processed by our microprocessor. From there, the microprocessor will send the data to the Bluetooth module via the UART serial communication. The Bluetooth module will then convert the serial port data to Bluetooth and exchange the data using short wavelength radio transmissions in the range of 2400-2483.5 MHz, providing our system with the capability of communicating with our Android phone over a short range. The Bluetooth module adds the capability of sending data to the Android phone without utilizing any cabling, like a USB. In order to implement a Bluetooth module into our system we will consider several products that will ensure that Bluetooth integration into our system is quick, simple and a cost-effective process. We will be considering Bluetooth modules which implement the EDR (Enhance Data Rate) technology, the Bluetooth High Speed technology and the Bluetooth low energy technology released in 2011 via the release of Bluetooth Specification v4.0. This type of Bluetooth modules are known as Bluetooth "Smart Ready" devices. The Bluetooth's low energy technology key feature is its low power consumption, one of the main needs in our design, possessing a maximum power consumption of 15mA and an average power consumption of about 1 µA. Just as important is the ability for our Bluetooth module to communicate with our microcontroller, requiring a module that utilizes Bluetooth SPP (Serial Port Protocol) that contains the ability to communicate with UART interface with a programmable baud rate. In terms of range, we will focus on class 2 modules because they are the most common and readily available. Class 2 radios have a range of up to 33 feet. There are thousands of different Bluetooth module available in the market today, we will use the requirements just mentioned along with the comparison of size and cost to determine which Bluetooth module will suit our system optimally.

### 3.3.1 Bluetooth Module Comparison

Utilizing the features mention in the previous section we decided to focus on three different Bluetooth modules from three separate companies. The Bluetooth modules we decided to compare are the CC2564-PAN1326, a Dual-Mode Bluetooth® v4.0 Smart Ready module from Texas Instrument Inc. and Panasonic, then the HC-05, a Bluetooth SPP module from Wavesen Co. and finally the TiWi-uB2, a Bluetooth module with BLE[14] (Bluetooth Low Energy) Smart Ready technology from LS Research. To begin we examined the CC2564-PAN1326, a dual-mode BT module that has the ability to be switched between acting as a master module and a slave module and was designed by Panasonic utilizing TI's Bluetooth chip. This products aims to lower the consumption of power while achieving a fast connection in order to transfer the data. Typically, Bluetooth RF performance is measured by Tx power, Rx sensitivity and RF blocking. The CC2564-PAN1326 achieves optimal RF performance in those areas, supporting extended range Tx power of 10 dBm typical output, a RF sensitivity of up to -93 dBm and its blocking feature increases sensitivity by 11 dBm. This module utilizes BLE 4.0 technology allowing the module to enter sleep mode when not in used only being activated for event activities in our system, increasing the power consumption of our proposed system. This technology allows the module to possess the capability of powering a device for up to 10 years of operation with a single coin cell, such as a CR2032. While in sleep mode, the CC2564-PAN1326 consumes less than 40 µA of power, and maintaining an overall power consumption of around 40mA. The supply voltage needed in order for the module to operate also is low and is compatible with the voltage supply of our microprocessor which will operate at around 3.3V, ideally. This module also supports dual mode which allows for BT or BLE technology and ANT, which will enable us to mix and match products and brands with the assurance that integration will be made easier. The CC2564-PAN1326 incorporates a UART interface that is used as wireless UART communication with a maximum baud rate of 4 Mbps. Another characteristic that is relevant to final selection of BT module are the dimensions, in order to minimize size and weight of our final product. Figure 3.9 presents the dimensions of the CC2564-PAN1326 Bluetooth module.

Next, we examined HC-05 a Serial Port Bluetooth module that is qualified as a Dual-Mode Bluetooth V2.0 + EDR. This module boost an integrated antenna that allows it to achieve high levels of sensitivity, up to -80 dBm. The HC-05 also incorporates an UART interface with programmable baud rates between 9600 and 460.8 k. As a dual-mode module it will provides us the ability to switch the module between master and slave mode, depending on the needs of our system. A big drawback of this module compared to the CC2564-PAN1326 is that it only implements BT v2.0 technology and lacks the BLE technology which allows the module to enter sleep mode while not in use. While it achieves superior power consumption while active at 25mA versus the 40mA the CC2564-PAN1326 module consumes, it never enters sleep mode therefore no matter whether is processing communication or not, the current remains high at 8mA in comparison to miniscule 40 µA current consumption of TI's BT module. The dimensions of the HC-05 Bluetooth module are displayed in Figure 3.10. Even though this module is able to achieve higher levels of sensitivity, power consumption takes priority in our design in order to deliver a final product that thrives in achieving low levels of power consumption.

Then we examined the TiWi-uB2, a module that also has the CC2564 Bluetooth chip from Texas Instruments, Inc. that fully supports Bluetooth 2.1 + EDR and Bluetooth Low Energy 4.0 technologies. The TiWi-uB2 also integrates a UART interface that is used a wireless UART communication with a maximum baud rate of 4 Mbps. While utilizing an external antenna in order to achieve an Rx receive sensitivity of up to -94 dBm. The implementation of BLE 4.0 technology allows the TiWi-uB2 to achieve best in class in power consumption, improving on the consumption of the CC2564-PAN1326 by a few µA while in sleep mode. The operating supply voltage range of this module is between 2.2 V and 4.8 V. Even though we will need to implement an external antenna to our design, this module's dimensions contribute to our goal of achieving a small final product while maintaining good performance and ultra-low power consumption. Figure 3.11 displays the dimensions of the TiWi-uB2 Bluetooth module.

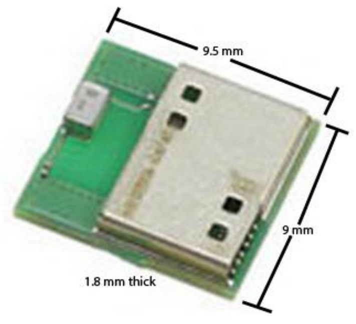


Figure 11 - CC2564-PAN1326

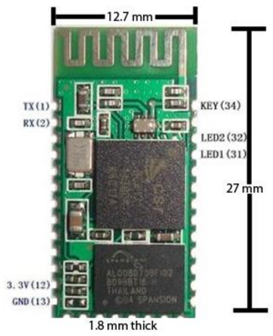


Figure 12 - HC-05

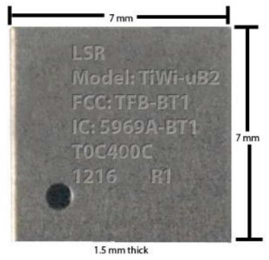


Figure 13 - TiWi-uB2

## 3.3.2 Bluetooth Module Final Selection

Through our research we learned that there are many different module options for different purposes. Therefore we eliminated some of the possibilities by focusing on Bluetooth modules that fall in the class 2 specifications or class 1.5 specifications, which allow for a maximum permitted power of about 4 dBm and 10 dBm, respectively with a range of up to 10 m (33 ft.). We found that to be sufficient for our design as the rider and thus the Android phone will never be 33 feet away from the box that will be mounted onto the bike containing our components including our Bluetooth module. Furthermore, obtaining certification for a Bluetooth system can be costly and very time consuming, with a cost up to $40,000. We decided to focus on Bluetooth modules that were already certified by at least the FCC (Federal Communications Commission). While certification by the IC (Industry Canada) and CE (Europe) were taken as an added bonus in order to expand our product to the international market. There are many other factors involved in choosing a Bluetooth module that led to our final selection of a BT module. While making our decision, we considered the following key aspects of our microcontroller:

- Data Rate – Instructions per second and speed (in Hertz)
- Power Consumption – Amount of power consumed by the BT module in sleep mode (if applicable) and active mode
- Antenna type – External or On-patch
- Size – The dimensions of the Bluetooth module
- Cost – The unit cost of the Bluetooth module

After looking at all the specifications of the three prospective Bluetooth modules, we came to a final decision of utilizing the CC2564-PAN1326 for our design. When compared to the HC-05 module we quickly realized how superior the CC2564-PAN1326 module was. The chosen Bluetooth modules achieves higher data rate of up to 4 Mbps, while maintaining relative performance in RF receive sensitivity at up to -93 dBm. In regards to power consumption, the CC2564-PAN1326 triumphs over the HC-05 by utilizing BLE 4.0 technology which allows the BT module to enter sleep mode, only consuming 40 µA of the current while in sleep mode and 40 mA while pairing (active mode). In comparison the HC-05 does not utilize BLE 4.0 technology and never enters sleep mode therefore consuming 25 mA while pairing (active mode) and 8 mA the remaining of the time. The CC2564-PAN1326 also contains all three certification giving us the capability to expand our product internationally, if desired, while the HC-05 only contains the FCC certification for the USA. Next we compared the chosen module to the TiWi-uB2 a very similar module that also utilizes a built in CC2564 Bluetooth chip from Texas Instruments, Inc. We found that the two modules contain similar specifications for communication, data rate and receive sensitivity due to both utilizing the same Bluetooth chip. As Bluetooth Smart Ready devices they both have the ability to support dual mode, classic Bluetooth technology (BT 2.1 + EDR) and BLE v4.0 technology, allowing the modules to improve power consumption while maintaining the same range. Dual-mode modules will allow us to create a product that can connect to existing or older mobile phones via Bluetooth and to newer mobile phones that support Bluetooth Low Energy. The two did have a slight difference in the areas of power consumption, operating supply voltage and current. The TiWi-uB2 performs slightly better in the area of power consumption, while the CC2564-PAN1326 requires slightly less supply voltage and current in order to operate correctly. Supply voltage and power consumption are an important aspect of our project as we are trying to achieve a product that is powered by the rider’s pedaling movement and momentum through the use of a dynamo. The difference between the two in power required and consumed is meniscal, allowing us the ability to choose either BT module for our project. The size and cost difference between these two module is also was also neglected, as the difference is also minimal. The final deciding factor when choosing a BT module was the antenna type, the CC2564-PAN1326 utilizes an attached antenna while the TiWi-uB2 utilizes an external antenna. After some considerations an attached antenna is ideal for our project opposed to an external antenna. Our design will have to withstand nature as our project is built for outdoor use, therefore an external antenna will make it harder for us to ensure that our prototype is waterproof and shock resistant. The CC2564-PAN1326 already has an attached antenna allowing us to reduce overall size of the system, reduce the cost associated with integrating the Bluetooth module into our design by removing antenna cost and enables us to more efficiently secure our design from damage.

Table 13 can be used for reference and supports the decisions made in the final selection of our BT module.

| | CC2564-PAN1326 | TiWi-uB2 | HC-05 |
|---|---|---|---|
| **General** | | | |
| **Bluetooth Standard** | BT 2.1 + EDR, BLE 4.0 | Built in CC2564 Bluetooth, supports BT 2.1 + EDR, BLE 4.0 | BT 2.0 + EDR |
| **Certification** | FCC, IC, CE | FCC, IC, CE | FCC |
| **Antenna on module** | Yes | No | Yes |
| **Profiles** | SPP, A2DP, AVRCP | SPP, A2DP, AVRCP | SPP |
| **Pin Count** | 24 | 33 | 34 |
| **Performance** | | | |
| **RF-Receive Sensitivity (Rx)** | Up to -93 dBm | Up to -94 dBm | Up to -80 dBm |
| **Current Consumption (Sleep Mode)** | 40 µA | 33 µa | No Sleep Mode 8 mA |
| **Current Consumption (Active Mode)** | 40 mA | 38.8 mA | 25 mA |
| **Communication** | | | |
| **UART rate** | Up to 4 Mbps | Up to 4 Mbps | Up to 3 Mbps |
| **Max Data Rate** | 2.178 Mbps | -- | 3 Mbps |
| **Host Interface** | HCI UART | HCI UART, PCM | UART |
| **Power** | | | |
| **Supply Voltage Range** | 1.7 – 4.8 V | 2.2 – 4.8 V | 1.8 – 3.6 V |
| **Output Power (Tx)** | 10 dBm | 10 dBm | 4 dBm |
| **Tx Current** | 33 mA | 47.1 mA | 40 mA |
| **Rx Current** | 11 mA | 13 mA | -- |
| **Environment** | | | |
| **Operating Temperature** | -40ºC to +85ºC | -30ºC to +85ºC | -- |
| **Dimensions** | | | |
| **L x W x H (mm$^3$)** | 9 x 9.5 x 1.8 | 7 x 7 x 1.5 | 12.7 x 27 x 1.8 |
| **Weight** | 0.25 g | 0.18 g | 0.9g |
| **Cost** | \$11.08 | \$8.99 | \$9.47 |

*Table 13 - Technical Details of Prospective Bluetooth Modules*

UPDATE:

For our final design, Roving Network's RN-42 Bluetooth module was used. The decision to use this module was made because we the RN-42 offers a more accessible surface mount area and also offered a cheaper breakout board set. Characteristics for the RN-42 can be seen in the table below.

| **RN-42 Bluetooth** | |
|---|---|
| Operating Voltage | 3.3v |
| UART Data Rate | 1 Hz |
| Antenna | On-board |

*Table 14 - RN-42 Characteristics*

## 3.4 Temperature Sensor

One issue with a lot of exercise applications on the market today is the inability to account for outside ambient temperature when calculating the number of calories a user has burned. The ability to do this will be made available through the use and integration of a temperature sensor. The temperature sensor will give the user the ability to sense and record ambient temperature while he/she is on a ride. The data acquired will also be saved on to the Android device for future retrieval and data analysis. In order to acquire the ambient temperature, we must employ the use of a temperature sensor capable of measuring temperatures experienced in nature. When selecting the temperature sensor, however, there is one thing we must consider. We must consider the possibility of extraneous temperatures as the device should be able to be used in cold climates (such as Canada, North Dakota, etc.) as well as tropical and southern climates (Florida, Georgia, etc.). With this in mind, we must ensure that our temperature sensor will encompass a range of temperatures to suit these conditions.

Though temperature acquisition is the main function of our sensor, there are a few other key operating specifications we must keep in mind when making our selection. First and foremost is the operating voltage of the sensor. Though we expect the voltage in and the voltage out to be minimal, a lower operating voltage is beneficial for our application. We will also investigate the voltage range for the temperature sensors to determine which has a more desirable range. In most cases it is the truth that the broader the temperature range of a sensor the better; this will hold true for our application as well. Last, but not least, we will consider both the accuracy of our compared temperature sensors as well as the scale on which they operate. Though the end success of our project does not depend solely on the accuracy of the temperature, in order to provide the end-user with the best results we must ensure that we are getting the most accurate data available. After we conduct a thorough comparison of both of our sensors, we will be able to make our decision. This decision will be discussed later on in this section (3.4.2). Though we will not discuss cost and economics of our sensors (the low cost of temperature sensors enables us to skip over this discussion), we will take the two into consideration when making our final selection.

## 3.4.1 Temperature Sensor Comparisons

After doing extensive research, we have found three viable temperature sensors that seem to suit our application. The three temperature sensors we have found are Texas Instrument's LM35, Analog Device's TMP35, and Honeywell's TD5A. All three temperature sensors offer similar feature sets, but each have their pros and cons. The first device we will review Texas Instrument's LM35 temperature sensor.

The LM35 is a well-known temperature sensor marketed by Texas Instruments as a "precision centigrade temperature sensor". The sensor offers low power consumption capabilities and operates at an absolute maximum voltage of 35 V. Though this measure tells us the ability of our device to withstand variable voltage inputs, it does not give us any valuable insight into our sensors performance capabilities. For this reason, we have provided a voltage vs. temperature chart (seen in Figure 14). This chart shows us the typical voltage of the sensor at continuous temperature levels. Looking at Figure 14, we can see that for a temperature of 20 ºC, our approximate supply voltage is roughly 3.3V. The operating range of this sensor is wide and ranges from -55ºC to 150ºC with a typical accuracy of ± 0.5ºC. This is quite a broad range of temperatures and far exceeds the requirements of the sensor for our application and expected conditions. The accuracy range is acceptable as well as our application will not suffer catastrophic damages due to any incorrect temperature readings.

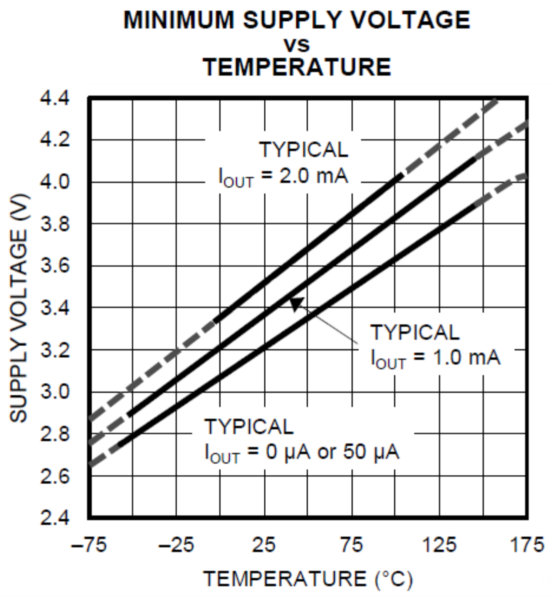


*Figure 14 - LM35 Characteristics*

The next temperature sensor that we will review is Analog Devices' TMP35. The TMP35 is also a low power temperature sensor and is said to have an absolute maximum operating voltage of up to 7 V. Again, this is not a comprehensive metric for comparability so we must refer to our voltage to temperature chart, which can be seen in Figure 15. Looking at this chart we can see that at just slightly under room temperature (approximately 20ºC) the TMP35 requires a supply voltage of roughly 2.1V. This sensor has an operating temperature range of -40ºC to 125 ºC and has a typical accuracy of ±1 ºC. This temperature range also exceeds the design requirements of our system however, the TMP35 has a smaller temperature range of which the implications will be discussed in Section 3.4.2.

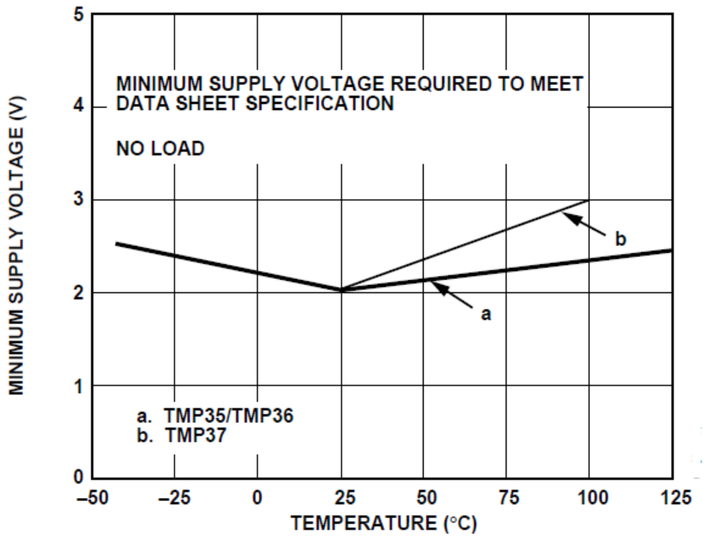


*Figure 15 - TMP35 Characteristics*

The last temperature sensor that we will review is Honeywell's TD5A temperature sensor (Figure 3.12). This temperature sensor, unlike the two previous sensors, is not considered to be a low power device due to its supply voltage nearing 10 V. The sensor has a temperature range of -40 ºC to 150 ºC, which is fairly comparable to the other two sensors, even offering a broader range than the TMP35. Though the temperature range comes close in comparison to the other two sensors, we cannot look past the fact that this sensor uses a supply current drain that surpasses the other two sensors. This device also outputs a current of 1 mA where the other sensors are able to offer an output current of < 60 µA. This device has a comparable accuracy rating at ±0.7 ºC

## 3.4.2 Temperature Sensor Final Selection

After carefully reviewing and comparing the LM35, TMP35, and TD5A we have decided to use the LM35 as our temperature sensor of choice in moving forward with this project. In the end, the LM35 had a current drain that was higher by 10 µA when comparing to the TMP35 but remained far ahead of the TD5A which drew a current of close to 1 mA. The temperature range of the TD5A and the TMP35 were fairly comparable with the LM35 having a greater operating range by 15 ºC. Though this may seem like a fairly wide gap, the temperature difference falls on the negative end of our Celsius scale referencing temperatures our device is never likely to reach. This basically means that we can consider the temperature range difference between the TD5A and the LM35 to be negligible. Another key deciding factor for temperature sensor selection was the typical accuracy experienced within the normal operating range. Looking at the TMP35 we see that it is able to keep a fair amount of accuracy at ±1 ºC, but was outdone yet again by the LM35, which was able to achieve accuracy of ±0.5 ºC. Comparatively speaking, we can see that the TD5A provides for the best accuracy of all of our devices at ±0.4 ºC. Though the TD5A offers better accuracy and a lower cost than the other two sensors, these two metrics do not dictate the best sensor. Instead, we must look at the results as a whole and make our decision accordingly. After reviewing all three sensors and their respective specifications, we have decided to go with Texas Instrument's LM35 temperature sensor. Hands down the LM35 offered the broadest operating range, and though it was the most expensive of the three it also had the lowest current drain and the widest storage temperature range. Reference data can be seen in the table below (Table 15).

| | **Texas Instruments LM35** | **Analog Devices TMP35** | **Honeywell TD5A** |
|---|---|---|---|
| **Temperature Range** | -55 ºC to 150 ºC | 10 ºC to 125 ºC | -40 ºC to 150 ºC |
| **Accuracy** | ±0.5 ºC | ±1 ºC | ±0.4 ºC |
| **Supply Drain** | < 60 µA | < 50 µA | 1 mA |
| **Max Voltage Input** | 35 V | 7 V | 10 V |
| **Cost** | $4.76 | $1.45 | $3.03 |
| **Storage Temperature** | -60 ºC to 180 ºC | -65 ºC to 160 ºC | -55 ºC to 170 ºC |
| **Operating Voltage at Near Room Temperature (≈ $20°C$)** | 3.3V | 2.1V | - |

*Table 15 - Temperature Sensor Comparisons*

## 3.5 Accelerometer

In bike dash one of the concepts we added to the system is Accelerometer, which measures the force of acceleration of the biker whether caused by gravity or by movement. This feature will help the rider keep track the speed he is going by. We used it as an input of the system to help us analyze the way the device is moving. There are so many accelerometers to choose from, after the research we did, we narrowed down to couple more energy sufficient for our project. The purpose of this component is to measure the acceleration of whatever object the sensor is attached to, or measure the tilt of the object.

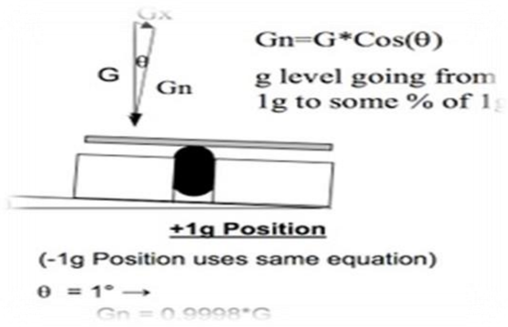

Gx
G
θ
Gn
Gn=G*Cos(θ)
g level going from 1g to some % of 1g
+1g Position
(-1g Position uses same equation)
θ = 1° →
Gn = 0.9998*G

Figure 16 - Measure of Tilt

So in this project we considered some features that will help us choose the right one before buying it. First of all, we must choose between an accelerometer with analog outputs or digital outputs. This will be determined by the hardware that we are interfacing the accelerometer with. Second, the number of axes, in this project two is enough, however if we want to attempt 3D positioning, we will need a 3 axis accelerometer. Also sensitivity, the more the better quality. Finally, the bandwidth, which is going to help us have a reliable acceleration reading.

### 3.5.1 Accelerometer Comparisons

When trying to choose the best accelerometer to fit our project, we considered some characteristics that we are looking for, such as range, interface, and number of axes measured, power usage which is very important, bandwidth, and sensitivity. Thus, the two accelerometers with chose are ADXL362 and ADXL335. Both are super power, small and thin. The ADXL362 micro-power digital output accelerometer can work for our project, because it has the 3-axes accelerometer controls the high side switch by monitoring the acceleration in 3 axes and closes or open the switch depending on the presence or absence of motion.

| ADXL362 | |
|---|---|
| **I/O voltage range** | 1.6V – 3.5V |
| **Power consumption** | 1.8 µA @ 100 Hz<br>3 µA @400 Hz<br>270nA @ wake-up mode<br>10nA @ standby current |
| **Package** | 3 mm x 3.25 mm x 1.06 mm |
| **Axes** | ±2g, ±4g, ±8g |
| **Temperature range** | −40°C to +85°C |
| **Bandwidth** | HALF_BW = 0<br>HALF_BW = 1 |
| **Output Data Rate** | Between 12.5 Hz and 400 Hz |

*Table 16 - Features of ADXL362 Accelerometer*

*Figure 17 - ADXL362 Accelerometer*

The ADXL362 is the lowest power MEMS accelerometer on the market consuming about less 2 micro-amps current and an output data rate up to 200Hz ODR with a 2V voltage supply, and it only consume 300 nA at the wake up mode. In addition to its outstanding power consumption level of the device, the ADXL362 system lever power savings is very intelligent, continuously operational at the motion-activated switch. In the presence of motion the system power is connected, thus it consumes normal current. In the other hand, when there is an absence of motion, the system consumes 0 power. Another feature that makes this component more powerful to use in our project, is that it has an enhanced activity detection function that distinguishes between the different kinds of motion. This can be used to prevent the system from turning on unnecessarily, which mean when the system is off is off, and when it is on it's on. Another additional features that the ADXL362 have, such as built-in micro power temperature sensor, ability to synchronize sample times, and two lower noise mode that allow the user to reduce the noise by a factor of 3. With a 2V power supply the competitors have been able to reduce the current between 10 and 20 micro-amps at 100Hz. And between 35 and 80 micro-amps at 400Hz the competition requires 500nA in standby mode. The figure below shows the current consumption versus output data rate.

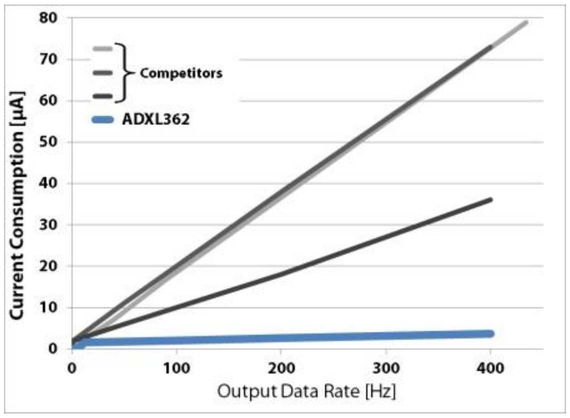


Figure 18 - Power Comparison Chart

In the other hand the ADXL335 that is very similar to ADXL362, is one of the best quality MEMS devices, which makes it perfect choice for our system. The table below tells you all feature this accelerometer has.

| ADXL335 | |
|---|---|
| I/O voltage range | 1.8V – 3.6V |
| Power consumption | 350 µA |
| Package | 4 mm x 4 mm x 1.45 mm |
| Axes | ±3g |
| Temperature range | −40°C to +85°C |
| Bandwidth | Between 1Hz and 500Hz |
| Output Data Rate | 0.5 Hz |

Table 17 - Features of ADXL335 Accelerometer

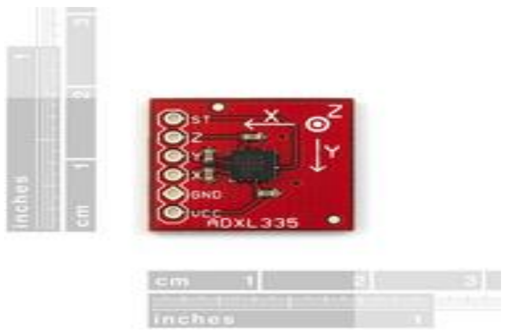


Figure 19 - ADXL335 Accelerometer

### 3.5.2 Accelerometer Final Selection

Our project will be demonstrating the process after the accelerometer activates the on switch from the existence of micro gravity. It should also be able to handle vibration and shock. Our accelerometer should have some type of sensor or mechanism that must output a zero flag once it detects micro gravity. Additionally, it should be low powered consumption because it will be running on battery. We will be researching what accelerometers are and what the different types are. Then we will determine what type we need. Thereafter, we will narrow our selection down to three to choose from. After we have our three main accelerometers, we will compare and see which one will be the best choice. There are many factors in choosing the right accelerometer that led to our final selection of accelerometer. After looking at the most important aspects of the two closely, we have come to final decision. The ADXL362 Accelerometer has an ultralow power consumption, very small that will fit perfectly in our power circuit board; also it is easy to communicate with the ADXL362 over SPI. The figure below shows how we are going to link the ADXL362 to the circuit board.

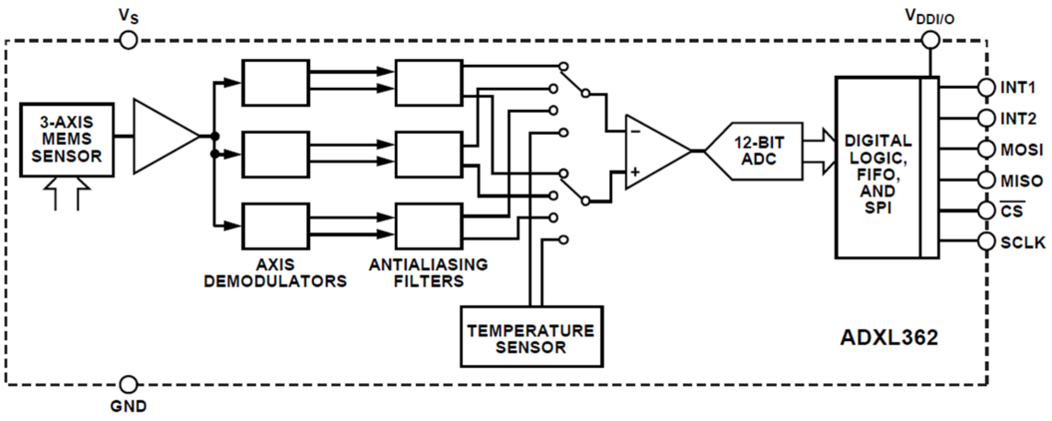


*Figure 20 - ADXL362 Accelerometer Block Diagram*

The figure below shows the ADXL362 eagle software layout and how it going to be connected when we wire it to the power circuit board along with the other components of the our project design.

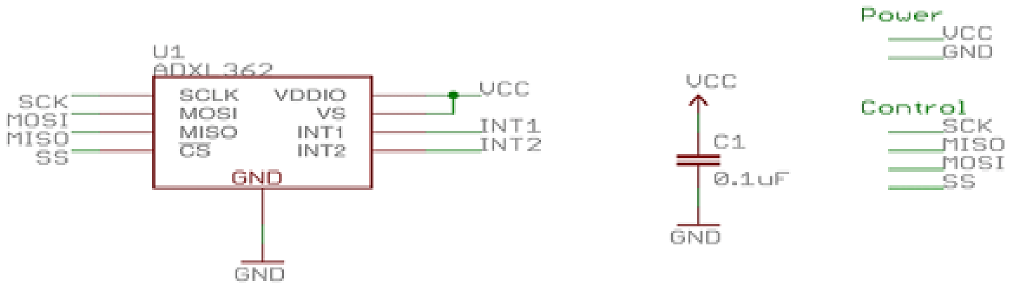


*Figure 21 - ADXL362 PCB Layout*

<u>UPDATE:</u>

For our final design, the ADXL335 will be used. The main reason for this change is that with the ADXL335 we are able to collect measurements via an ADC channel whereas with the ADXL362 we had to collect measurements by using an SPI connection. This change benefits us by allowing more CPU power to be used for other operations as we will not need to use an external clock.

## 3.6 Dynamo

To continue our design we decided to add the most important component of the system which is the dynamo, that is made up of two permanent magnets with a coil inside their poles, so when the coil spins the magnetic field produce an electric field which drives the charge carriers through the wire forming a current. In our project this dynamo is going to have a transformer that consists of two wire coils. The first coil is the primary and the other one is the secondary coil. The purpose of coils that wound onto a core made of iron is when alternating the voltage is applied to the primary coil, the magnetic field that present in it, changes in accordance with the rotation, inducing a voltage to the secondary coil. Our goal by using the dynamo is provide electricity for the light at night and to generate power to the power circuit board. The figure below shows how the dynamo produces electricity.

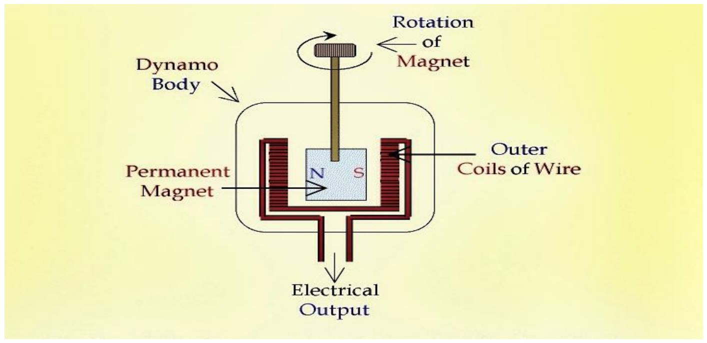


*Figure 22 - Bicycle Dynamo Mechanism*

To further analyze the selection of this component, we will consider some features such as the generator efficiency, weigh, axle-type, drag of dynamo hub, drilling and economic cost. For our project we will focus on the dynamo that has less weigh and more energy efficient.

### 3.6.1 Dynamo Comparisons

After consideration we narrowed our selection to couple dynamos that will be a good fit for the design system. The dynamos we chose are the Joule 3 dynamo hub, and Shimano Alfine DH-3N80 dynamo hub. Both are very similar to an extent. For our project we had some ideas of that dynamo going to help the rider in certain circumstances. For instance, when it comes to riding a bike at night or during the day, a high quality dynamo wheel can mean freedom from constantly replacing your batteries. After the research, we found that the older dynamo hubs had a lot of drag and weighed slightly less than a solid block of metal. In the other hand the new technology of dynamo hubs are much lighter, and do not act like a resistance while pedaling. The first dynamo we chose is the Joule 3 is one of the lightest, most efficient bicycle dynamo hubs on the market. The table below shows the feature of the Joule 3 dynamo hub.

| | Joule 3 Dynamo |
|---|---|
| Generator Efficiency | 73% |
| Weight | 355g |
| Hole | 32 |
| Output | 6V, 2.4W |

*Table 18- Features of Joule 3 Dynamo*

The other dynamo we chose is Shimano Alfine DH-3N80, which is super-efficient for our design, and it is virtually drag free 3 Watt power supply, with a quick release axle for standard dropouts. One of the reasons we picked this dynamo is that we can use it with a rim brakes at the center lock disc brake, and its weather protected. For our bike dash project, we focused on the power consumption. Since we are using a small dynamo to generate electricity, the less energy is lost throughout the circuit the better. So we decided to implement energy efficient components whether possible. The dynamo in our design will be mounted in the rear wheel away from the rider, distributing the power to device component.

| | |
|---|---|
| **Shifting** | Both hubs can shift under load. It shifts easier and smoother under high load and uphill |
| **Gearshift Lever** | More gears can be shifted in one movement |
| **Gear Ratio** | One higher and one lower gear |
| **Break System** | Offer a disc brake and rim brake option |
| **Weight** | 490 g |
| **Wheel Mounting** | Uses the "easy click connector" |
| **Power Supply** | 3.0 Watts |

*Table 19 - Shimano Alfine DH-3N80 Features*

### 3.6.2 Dynamo Final Selection

One of our goals first when we started thinking about this part was to design an electric generator that satisfy our needs. This involving researching types of magnets, DC vs AC generator design. Because the dynamo needed to be small, durable, lightweight, and more efficient when it comes to dynamic environment, we decided to purchase one on the market. To get the onboard devices power, the dynamo is going to take the mechanical energy applied by the rider into the bicycle and converting it into electricity. Thus the dynamo will feed the battery that we going to use as well and convert it into chemical energy to be stored. From this stored energy our devices such as the LCD and GPS will be powered by the electrical energy converted from the batteries.

After we compared both dynamos, there are some factors that will us choosing the best-fit dynamo for our project. After looking closely to the main key aspects of the two dynamo hubs, we have finalized our decision. The Shimano Alfine DH-3N80 dynamo hub will be used for our design, because its cost is cheaper than the Joule 3, also its new technology especially for sport bikes, and it is more energy efficient.

|  | DH-3N80 | Joule 3 |
|---|---|---|
| **Efficiency** | 70% | 73% |
| **Weigh** | 490gd | 355g |
| **Spoke Hole** | 32 | 32 |
| **Over Lock-nut Dimension** | 100mm | 100mm |
| **Price** | $139.95 | $149.99 |

*Table 20 - Dynamo Comparison*

## 3.7 Heart Rate Sensor

To continue our research we examined heart rate sensor and the benefits it will bring once integrated into our system. A heart rate monitor system consists of two parts a transmitter and a receiver, which will be our Android phone. As the heart beats, an electrical signal is transmitted through the heart muscle, that electrical activity will be detected through the skin. The transmitter will be placed on the skin around the chest area in order to pick up the electric signal. The transmitter will then send that electromagnetic signal containing the heart rate data to the Android phone which will be used to display that information to the user. A key benefit of heart rate monitoring is that it helps you maintain the optimal heart rate target zone during your workout in order to allow you to reach your specific fitness goal. The ability to exercise in the right heart rate zone helps the user optimize their performance during the working. Users that are aiming for a fat-burning goal require 40 to 80 minutes in one zone while an aerobic fitness goal requires 10 to 40 minutes in another. The target zone is a percentage based on your maximum heart rate according to age. The following table (Table 21) displays the target heart rate zone according to age, this information was obtained from the American Heart Association.

|  | **Heart Rate Target Zone 50-85%** | **Average Maximum Heart Rate 100%** |
|---|---|---|
| **Age** | **Beats per minute** | **Beats per minute** |
| **20** | 100-170 | 200 |
| **25** | 98-166 | 195 |
| **30** | 95-162 | 190 |
| **35** | 93-157 | 185 |
| **40** | 90-153 | 180 |
| **45** | 88-149 | 175 |
| **50** | 85-145 | 170 |
| **55** | 83-140 | 165 |
| **60** | 80-136 | 160 |
| **65** | 78-132 | 155 |
| **70** | 75-128 | 150 |

*Table 21 - Target Heart Rate Zones*

## 3.7.1 Heart Rate Sensor Comparisons

Today there are many different technologies available that allow monitoring of the heart rate while exercising.  Some of those technologies include finger sensors, hand grips with built in heart rate sensors, wrist watches, wrist sensors and chest strap transmitters.  After some research, we decided to compare a chest strap heart rate transmitter and an armband heart rate monitor both have built in Bluetooth capabilities in order to collect and transmit the heart rate data that’s obtain.  Utilizing one of those heart rate transmitter will allows us to simplify integration, while making it comfortable for the rider to integrate during their working as is something that could be easily wore on the rider.  Which eliminates the need to use cables and will not affect the rider’s ability to ride the bicycle.  Chest strap and armband transmitter models also offer continuous heart rate information without needing to stop during exercise to measure the rider’s heart rate.   During our research we will compare two similar products the H7 heart rate sensor from Polar and the RHYTHM Bluetooth armband heart rate monitor from Scosche.  The Polar H7 heart rate sensor is chest strap transmitter compatible with Bluetooth smart ready devices that has the ability to operate with just a rechargeable CR 2025 cell battery.  The Polar H7 boast a battery life of around 200 hours will in use and has an operating temperature range that fits our needs of -10 degrees Celsius to +50 degrees Celsius.  Meanwhile the RHYTM Bluetooth armband is also compatible with Bluetooth smart devices while boasting a battery life of around 6 hours.  The RHYTHM has an operating temperature range of 0 degrees Celsius to +45 degrees Celsius.

### 3.7.2 Heart Rate Sensor Final Selection

While we simplified the selection process of the heart rate sensor by limiting our proposed heart rate sensor to armband or chest strap technology. The accuracy of both sensors are both within +/- 3% range and both have the ability to communicate via Bluetooth using a data rate of 1 kHz. While the Polar H7 heart rate sensor has a large distinction in battery life, the high cost of the transmitter led us to select the RHYTM Bluetooth armband heart rate monitor from Scosche. This decision was made in order to stay within the project's budget and provide a cheaper product to the user. Table 22 displays some of the specifications for the heart rate sensors in order to support our final decision.

| | Scoche RHYTHM | Polar H7 |
|---|---|---|
| Heart Sensor Style | Armband | Chest Strap |
| Bluetooth Enable | Yes | Yes |
| Bluetooth Certification | FCC | FCC, IC, CE |
| Battery Life | 6 hours | 200 hours |
| Operating Temperature | 0 ºC to +45 ºC | -10 ºC to +50 ºC |
| Dimensions (inches$^3$) | | 8 x 1.5 x 5 |
| Weight (oz.) | | 1.6 |
| Cost | $ 35.00 | $ 69.99 |

*Table 22 - Specifications of Heart Rate Sensors*

## 3.8 Existing Similar Projects and Products

### 3.8.1 Stationary Bikes

Stationary bikes or "exercise bikes" are mostly used for exercise in order to increase general fitness or for training purposes for professionals. It's also often used for physical therapy due to its low impact, safe nature that allows the user to achieve an effective cardiovascular exercise. Most exercise bikes today incorporate and ergometer, a device used for measuring the work a person exerts while exercising on the bicycle. Even though stationary bikes are meant for indoor use we found that they are very similar to when our product is trying to achieve. They provide the user with its heart rate, speed, calories burned and time elapse during the workout. One stationary bike that would be used for research purposes is the Augie's Quest Lifecycle upright exercise bike. A self-powered exercise bike that utilizes Polar Telemery technology for heart rate monitoring. In order for the heart rate monitoring services to work the rider must utilize a chest strap transmitter. This machine has a power requirement of an AC power line of 115 Volts and 15 amp. The machine also has the ability to display information to the rider through the use of an LCD display,

### 3.8.2 Bike Buddy

To continue our research we examined a Senior Design project from fall 2009 performed by a group of Electrical and Computer engineering students here at the University of Central Florida. Their project named the “bike buddy” aimed to enhance the experience of the rider while riding his bicycle. Inspired by the high cost of energy, the group utilized an electric AC generator to harness power from pedaling. Their AC generator then powered the onboard components through a system of lithium-ion batteries. Their design implements a temperature sensor, a microcontroller, a graphical LCD, and a GPS receiver in order to provide the rider with information relevant to their riding experience. That data was then displayed through the use of a LCD monitor providing the rider the ability to visually see their velocity and direction, the time of day, its geographical location as longitude and latitude, the ambient temperature, the charge state of the lithium-ion batteries and the power being generated by rider’s pedaling motion. The design also provides the ride with a USB plug-in for various devices such as iPods, cell phones or digital cameras for charging purposes. Their project goals was to efficiently convert power generated by the rider through the use of an electric AC generator in order to charge lithium ion batteries via the electric AC generator, while maintain full operational capabilities of the onboard peripheral devices. Their project was also sponsored by Progress Energy as most of their system components aim to achieve ultra-low power consumption.

The group designed their prototype utilizing an Atmel ATmega128L microcontroller, which is part of the ATmega microcontroller family of one of the prospective microcontroller evaluated for our project. In their system the microcontroller had an operating voltage of 3.3 volts and requiring a power consumption of about 62.7 mW. To obtain the riders velocity, direction, time and geographical location data the group integrated the LS20031 GPS unit into their design, a unit that required an operating voltage of 3.3 V and a power consumption of 231 mW. The design also implement a temperature sensor, the DS1625, in order to measure the ambient temperature. The temperature sensor chosen for their project required an operating supply voltage between 4.5 V and 5.5 V. Then the data gathered from the sensor and the GPS are displayed on a 128 x 64 pixel KS0108 LCD screen that was mounted on the bike’s handle bars and required an operating voltage of 6 V and a power consumption of 1320 mW. The USB female port required an operating supply voltage of 5V and a power requirement of 2500 mW. As a whole the total system design required a total power of 4.12 W. In order to provide the required power to the system their team created a system that is entirely self-sufficient. By attaching a small electrical generator to the rear bike they were able use the rider’s pedaling motion to create an AC voltage that is then convert into a well filtered, regulated DC voltage. That DC voltage is then used to charge one of the two rechargeable 7.4 V lithium ion batteries they implemented into their design and to power the switching circuit that enables their system to decide which battery needs to be charge. Meanwhile the other 7.4 V lithium ion battery is used to power the onboard devices of the system. A top-down block diagram of Bike Buddy can be seen below in Figure 23.

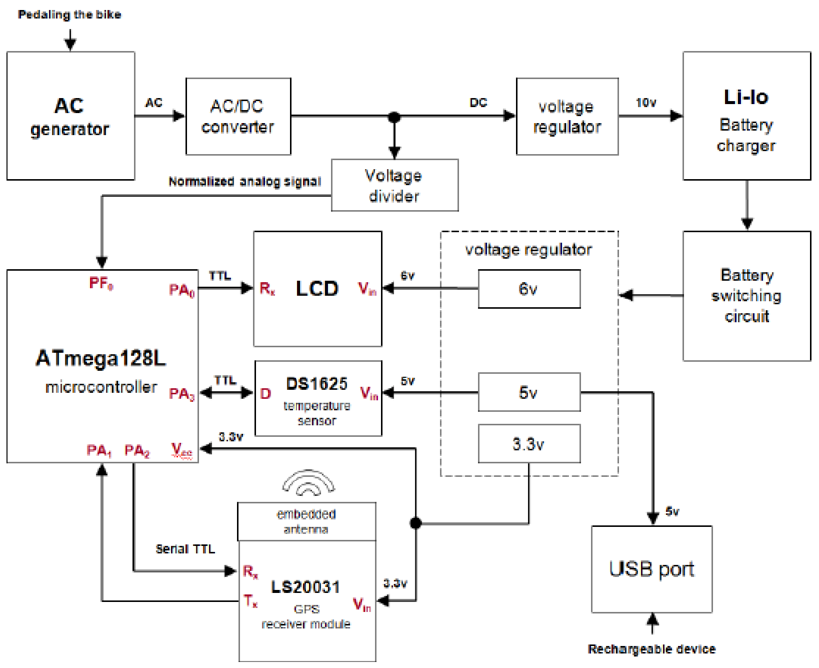


*Figure 23 - Top-down Diagram for Bike Buddy*

This project is similar to the system we are attempting to design. As a group we will aim to achieve even greater improvements in the area of power consumption while adding more advanced features such as providing our product compatibility with Android phones. We will be using improved components that will allow us to achieve higher accuracy while maintaining ultra-low power consumption. We plan to implement a Bluetooth module in order to send data to the Android phone. The data received by the phone will then be used to create a convenient Android application that will be used to display the wanted information to the rider. Making our product self-sufficient with the ability to provide the rider with a rich GUI display on their Android phone.

## 4.0 Design and Integration

In the design and integration phase of the project, we focused on integration of all of our physical components as well as their integration with our software component. We also focused on designing components that were not to be purchased as a COTS system. Some of these devices are the PCB, wiring, housing, etc. This will be further outlined as we continue this section. It was intended that the software portion, both the Android application and the MCU OS, be completely original and are included in the design section for this reason. As we moved forward with the integration phase, we began by integrating our physical components of the system so that we were able to better understand constraints imposed on the software. Though a substantial effort had been made to ensure that we selected parts that could accommodate the entire scope of our project, we ran into issues that caused the system to not run as expected. In order to best run our system, we had to change the microcontroller, GPS unit, battery, Bluetooth, and accelerometer. This was a major overhaul to the original design selected in the Research section of our paper. The changes to the design will be outlined in the following sections.

There are some things that we had to consider heavily in the design phase of our project. First and foremost, we had to consider the durability of the system. As we mentioned previously, our system must be very durable as the device will be subjected to adverse environmental conditions. The most prominent danger to our system is water damage meaning that we had to carefully consider the design of our enclosure. We may not have needed a waterproof system, but needed at least a water resistant system. Another key design aspect of our system was the need to build a system that was very light and compact. This design aspect will be discussed throughout the design phase as it is important across the entire scope of the project.

## 4.1 Hardware

To begin our design phase, we started with the physical layer of our project, which includes the PCB, the battery pack, the wiring, and the enclosure. When we were designing the hardware for our system, there were a couple things that we had to keep in mind. First and foremost was the ability to integrate each component with the others (and as a system at large) seamlessly and as efficiently as possible. One of the main concerns with the integration of the hardware was managing the wiring of the system as there are many opportunities for the wiring to become disorganized and tangled in other system components. This, however, will be discussed later on in our integration section (Section 4.3). The next thing that we had to keep in mind when we designed our physical layer was the cost. While we attempted to use items considered to be inexpensive, we did not sacrifice quality of the project. To ensure that our product met all of our quality expectations, each component endured a series of tests as outlined in the testing section of this report (Section 8).

Beginning with hardware, everything from the suitable battery to best the microcontroller to use for calculating the amount of power generated was thoroughly investigated to be certain that all possibilities are evenly weighted out. It was only after this process that decisions were made on the types of the components that we used, their configuration, the different options and details about each option for all main hardware components of the system.

### 4.1.1 Power System

The first design segment of our system was creating a power system that would be able to accept a 6V AC input and output a steady 3.3V to our Main System. To do this, we used a Dynamo outputting 6V AC, passed it through a charge regulator which also charges a battery, then passed that voltage output into a Low Dropout voltage regulator giving our Main circuit the proper voltage. A schematic of the charging system can be seen in the figure below, Figure 24.

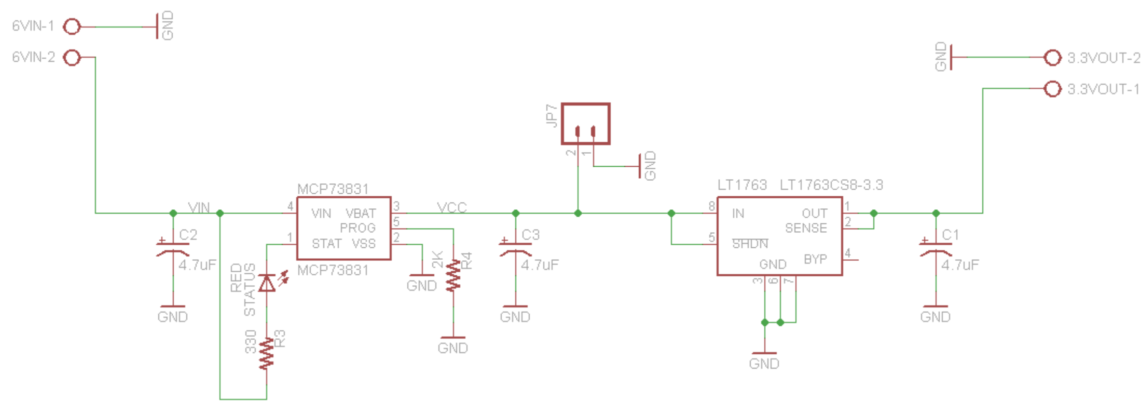


*Figure 24 - Charging System Schematic*

The first step in designing our power system was mounting the dynamo on the bike. To do this, a custom wheel build was required for two reasons. The first was that the new dynamo hub was a 32-spoke hub and the original wheel was a 36-spoke wheel. To accommodate our dynamo, we purchased an MTB Hybrid wheel (meant for both road and trail). The other reason for a custom wheel build was the difference in size between the old hub and the new hub. The new hub was significantly bigger, meaning that a custom spoke length and pattern would be required. With this being a big job, the rim build was outsourced to Dave's World Cycle in Altamonte Springs, FL. The next step in our charging system was converting the AC voltage output by the dynamo into a DC voltage that was usable by the charging circuit. Due to issues with time restrictions and team member involvement, a COTS AC/DC converter specific to the dynamo was purchased. This device takes the 6V AC voltage output by the dynamo and converts it in to a 5V DC source for the charging circuit. The device used was the BioLogic Reecharge adapter for hub dynamos. The converter plugged directly into the lead-outs of the dynamo and included a standard dc jack for input into our charging circuit. The dynamo and AC/DC converter can be seen in the image below.

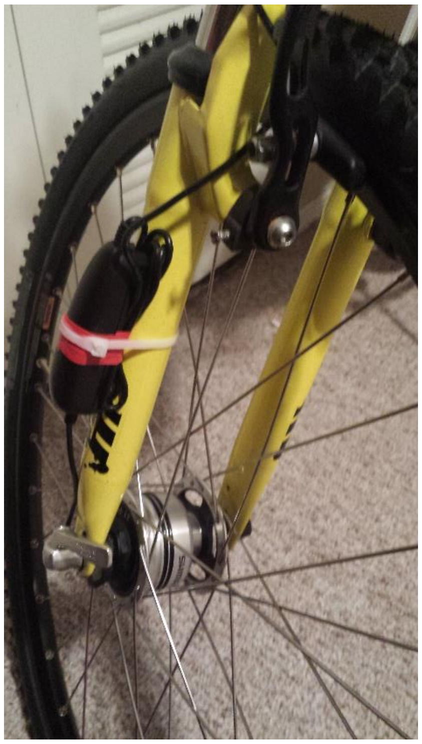


Figure 25 - Hub Dynamo and AC/DC Converter

The charging system is comprised of 3 key components that work together to output a steady 3.3V voltage source for our Main board. The first component seen by the AC/DC converter’s 5V output is the MCP73831. The MCP73831 is a charge regulator that allows a voltage input of up to 6V and will produce a 4.2V charging voltage for a LiIon/LiPo battery. As the dynamo is turned, a 3.7V LiIon battery is charged at a maximum voltage of 4.2V. This battery will discharge within a range of 3.5V – 4.2V. After the voltage passes through these two components (the charge regulator and the battery) it is passed in to a LDO (Low Drop Out) voltage regulator. This voltage regulator accepts a voltage input up to 20V and produces a 3.3V output (measure to be approximately 3.28V). The voltage regulator that was implemented in our design was Linear Technologies LT1763. After the voltage has passed through the voltage regulator, it should be considered a safe voltage for our Main system.

### 4.1.2 Main System

The second design segment of our system was creating the Main system. The Main system board contains all of our sensors (GPS, Temperature, and Accelerometer) as well as our Bluetooth module and our microcontroller. The purpose of the Main board is to use the MSP430 microcontroller to collect data from the external sensors and pass the data to the Bluetooth module to be transmitted to the Android device. A schematic of the Main board can be seen in the image below, Figure 26.

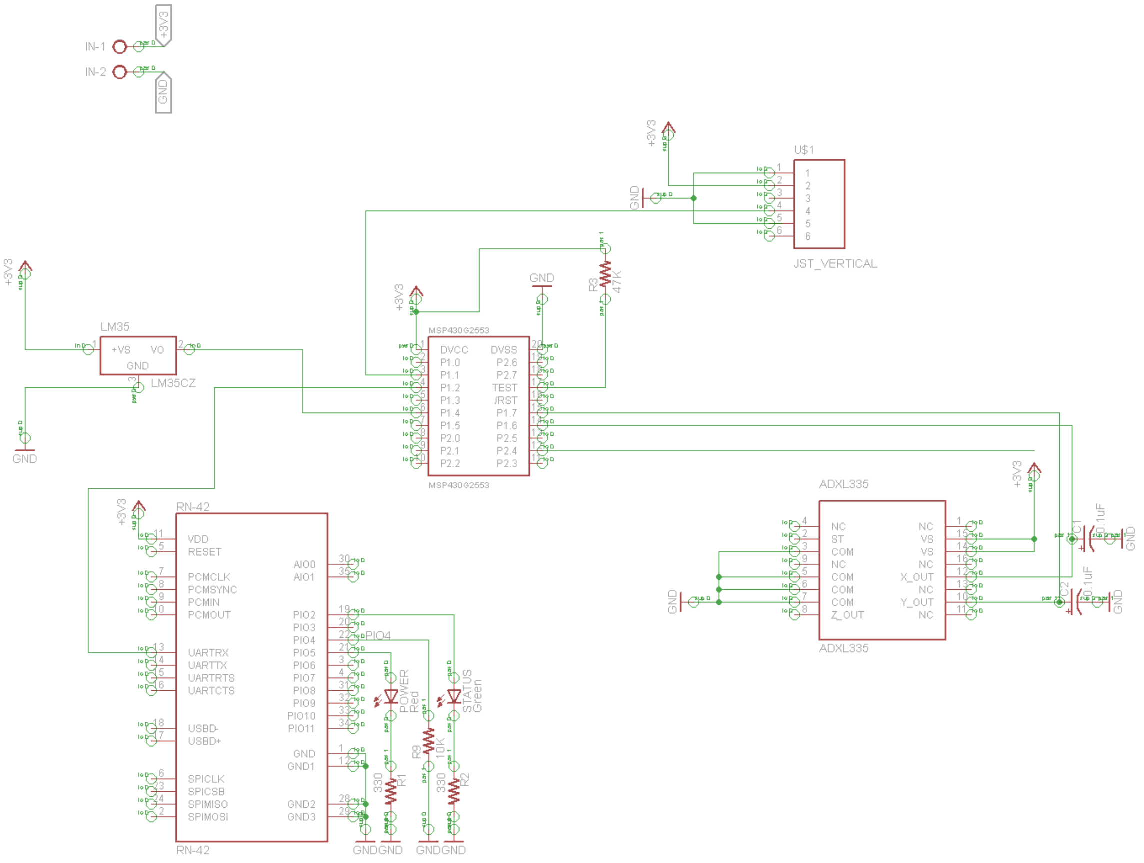


*Figure 26 - Bike Dash Main System Schematic*

The GPS unit used in this project is USGlobalSat's EM-506, a 48 channel GPS module that includes a built-in antenna. In the Bike Dash project, the GPS unit is used to acquire GPS coordinates and relay GPRMC sentences to our microcontroller. To receive the proper NMEA sentences, the TX pin of the GPS unit is connected to the RX pin of the MSP430, creating a UART connection and allowing data to flow asynchronously. Once data is received by the microcontroller, it is relayed out to the Android device through a Bluetooth transmitter for further parsing. The sentences passed to the Android device are used to calculate the users speed, heading, and current location. GPS coordinates are also used to build the map view of the application.

The accelerometer used in Bike Dash is Analog Device's ADXL335. This device provide a three dimensional measurement of acceleration, though we will only use two for our application, and operates at the necessary 3.3V. The ADXL335 is read by connecting two ADC channels on the MSP430 (one for the X-axis and one for the Y-axis). The accelerometer provides results in terms of a voltage measured on the channel in mV and we use the voltage differences to calculate the force and angle. These measurements are used to determine if the user has been involved in a collision and will ultimately alert the appropriate parties of such an incident.

The temperature sensor we are using in the Bike Dash system is Texas Instrument's LM35. The LM35 is a precision centigrade temperature sensor and has an output that is linearly proportional to the Centigrade temperature, as seen in Figure 1. The sensor also offers low power consumption capabilities and operates at an absolute maximum voltage of 35 V. To measure the ambient temperature, we use an ADC channel of the MSP430 to capture the voltage at the channel after applying a ~3.3V input voltage. This voltage reading is then transmitted to the Android device via a Bluetooth link where it is converted into degrees Celsius and Fahrenheit.

The microcontroller used in the Bike Dash project is the MSP430G2553. This microcontroller is used to collect data from all of our sensors and relay the information to the Android device for both processing and display. The MCU will be responsible for providing data both accurately and efficiently and must do so under stressful environmental conditions as this device will be subject to moderate environmental conditions. The MSP430 reads in data using a combination of ADC (Analog to Digital Conversion) channels and its RX and TX UART channels. Data from the GPS unit is collected at the receive buffer and forwarded through to the transmit buffer. As data is received at the transmit buffer, it is passed to the Bluetooth module for transmission to the Android device.

In order to establish communication between the Android device and the onboard components we decided to use the RN42 Bluetooth SMD Module from Roving Networks. This Bluetooth module is a powerful and easy to use module that boasts a small footprint. For this project the Bluetooth module will be used to replace serial cables in order to transfer data from the MSP430 to the Android device which utilizes its built-in Bluetooth module to be paired with the RN42. This Bluetooth module also contains an embedded Bluetooth stack which allowed for an easy integration into our system. The data to be passed from the microcontroller will be place in the RX buffer of the Bluetooth module which will then put that data into the TX buffer. The data is then sent to the Android device by utilizing a simple UART interface with a data rate of 1Hz. The RN42 Bluetooth module is in slave mode therefore the android phone will serve as the client and initiate the Bluetooth connection. In order to allow the application to run smoothly once connected the Bluetooth connection remains active until the entire system or Android device are powered off.

### 4.1.3 Wiring

One of the most reliable ways for bike dash to communicate with each other would be through a hard-wired system. This configuration would be a relatively a low cost option way for the bike dash. Though reliable, a hard-wired system could pose a lot of problems.

The wires would have to somehow be concealed to reduce the chances of it getting damaged and for visual reasons as well. Before construction of the circuits, each sub-system will be simulated in MultiSim to guarantee it is functioning correctly. Once each subsystem is verified an entire circuit diagram will be simulated to ensure it is functioning correctly as well. Each system inside Bike dash will be assembled separately on its own board for testing in the senior design lab. If one part isn‘t working correctly it could damage the components, and this must be avoided at all costs.

The design systems to be built are the power supply and voltage regulation circuits. Other devices such as GPS and temperature sensor will be interfaced one at a time to ensure proper functionality. First the power supply circuit will be constructed on its own perforated board. Voltage and current reading will be taken to acquire the transfer function of the circuit. The Android device and microcontroller are the next parts to be interfaced. This is done because they will be able to tell us if our circuit is working correctly on the data transfer level, which will be required for the subsequent devices to be interfaced. Once the microcontroller and Android device are working correctly we will be able to display information coming from the GPS module, temperature sensor, Accelerometer, and power being generated in the power supply. This will be accomplished with some testing routines written in the Testing section.

## 4.2 Software

In the software component of our design, we will create the architecture for our microcontroller as well as create our Android application. In this section we will not only outline our code intentions, but the intended development process as well.

### 4.2.1 Microcontroller

The first step in the software design stage for this project is to create the software for the microcontroller. The microcontroller we will be using is the MSP430G2553 and we will be using the Launchpad to program our microcontroller. With this MCU comes the ability to create our code in either assembly, C, or C++ (we will not include C++ in our discussion, however). When deciding which language we will use, we must consider a few things first. C is a very powerful language that offers its users the ability to create both highly technical solutions as well as applications that require little code, however, there is a cost that is associated with using a high level language like C known as overhead. This overhead is not as costly for simple applications but as code size grows, so will the amount of overhead. The upside to this, though, is that C offers both flexibility in development as well as faster development times. In order to make an educated decision we must consider the following questions:

- Time - How long will it take to develop a solution?
- Libraries - What libraries or open-source code is available
- Efficiency - How efficient will our code run?
- Memory - What are the memory restrictions?
- Maintainability – How easy is it to maintain? Can anyone perform maintenance?

When we are comparing the two languages (C and assembly) we must consider the time it will take to develop an effective solution using either language. While assembly may seem to be a simple language, due to the nature of direct register access and limited instruction sets assembly could prove to take a much larger time investment than using the C language. Using assembly language would require a large learning curve (as no group member has experience developing in this language) whereas experience with the C language is found in all three group members. For purposes of code maintenance, this situation would be ideal. In the best interest of time and efficiency we will create our MSP430 MCU architecture using the C language.

To write our microcontrollers software in the C language we did not simply employ a single IDE. To be able to successfully write and utilize C code on the MSP430G2553 we used TI's Code Composer Studio to understand and write the foundation code. We then translated all the code in CCS to the Energia IDE. The reason we used two IDEs is because while Code Composer Studio allowed us the ability to modify our MSP's functionality more precisely, Energia allowed us the ability to modify our code quicker as it included libraries not included in Code Composer Studio.

To write the actual code for the microcontroller we will set up our program in the same manner we would set up any other C project with the exception of the header file. In order to code a project in C we must include the header file MSP430G2553.h, which includes most of the MSP430 instructions.

Creating a complete architecture for our microcontroller will be a hefty task, and thus we must plan our approach carefully. In the interest of modularity, we will develop functions to handle individual components of the system. Within each function, we will handle incoming and outgoing data, as well as the processing of such data. The main function of the OS of the MCU is to accept, translate (in the case our ADC channels) and retransmit data to the appropriate channels. To handle the data in, data out, and data handling in our MCU we intend to have the following functions:

**MAIN:**

This function will be mainly responsible for the idle process of our microcontroller which will act as a decision engine to direct the call of our system sub-functions. The sub-functions that we will use will be outlined below. To set-up the idle process within the main function we will use a while loop that runs continuously. We will do this because our intention is to have to MCU run indefinitely while it is receiving power. To make sure that the while loop runs indefinitely we will set the while loop as follows: while(1){}. The main function will take in no parameters.

**BLUETOOTH:**

This function will be the function that handles any and all communication with external devices. For our design we will incorporate the use of a Bluetooth capable mobile Android device. The Bluetooth class will receive data from each of the other functions, which it will convert into packets and broadcast over the Bluetooth network. Information intended to be transmitted is sent through UART by writing the data to the TX buffer of the MSP430.

The MSP430's TX line is connected to the Bluetooth's RX line. Once data is passed to the Bluetooth device, it is transmitted automatically by the RN-42s embedded Bluetooth stack.

**GPS:**

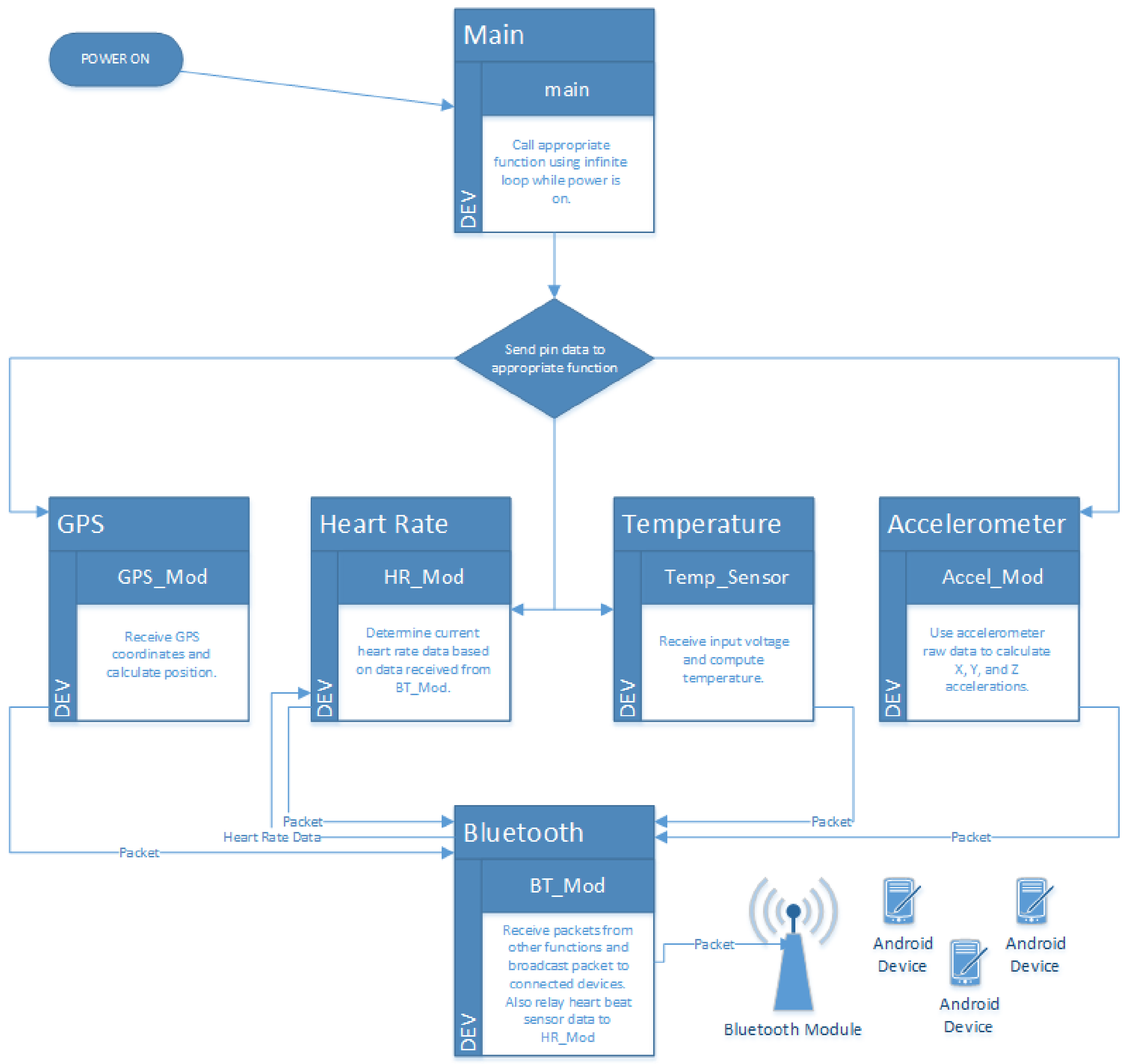
The GPS function will be responsible for taking in data from the GPS. The GPS data is collected in strings of 700 characters. This is done to ensure that (due to noise) we are able to receive a full RMC sentence with each data pull. Once we receive a full set of data, it will be passed to the Bluetooth function and sent out over the Bluetooth network to connected Android devices. All of the triangulation algorithms and positioning take place within the Android application in order to keep code size on the microcontroller to a minimum.

**ACCELEROMETER:**

The accelerometer function is responsible for receiving data from the accelerometer module and converting it in to useable data. Data is pulled from the accelerometer by reading the appropriate ADC channel. In our case, we need to read two ADC channels as we are using both the x and y axis of the accelerometer. We read first the x axis and then successively the y axis. Each data string respective to the axis is sent as it is received in an attempt to provide real-time statistics.

**TEMPERATURE:**

The temperature function is responsible for receiving an input voltage reading sent read from the temperature sensor. This voltage reading is representative of the current temperature and will be converted by the Android device. The temperature sensor outputs approximately 1mV per degree Celsius. Once we have read the temperature sensor's ADC channel, we pass the data to the Bluetooth function so it can be sent to a connected Android device.



POWER ON
Main
main
DEV
Call appropriate function using infinite loop while power is on.
Send pin data to appropriate function
GPS
GPS_Mod
DEV
Receive GPS coordinates and calculate position.
Heart Rate
HR_Mod
DEV
Determine current heart rate data based on data received from BT_Mod.
Temperature
Temp_Sensor
DEV
Receive input voltage and compute temperature.
Accelerometer
Accel_Mod
DEV
Use accelerometer raw data to calculate X, Y, and Z accelerations.
Packet
Heart Rate Data
Packet
Packet
Packet
Bluetooth
BT_Mod
DEV
Receive packets from other functions and broadcast packet to connected devices. Also relay heart beat sensor data to HR_Mod
Packet
Bluetooth Module
Android Device
Android Device
Android Device

*Figure 27 - Function Flowchart of MCU Code*

### 4.2.2 Android Application

The Android application will be another important factor or our design, as it will be utilized to display the information received from the onboard components into a well design GUI layout that allows the user to have access to the desired information. The Android application will also have the ability to view past workout in order give the ability to measure progress. While designing our Android application we will take in to consideration the following design issues:

- "Fit for purpose" -- The application must meet the user needs (display speed, location, time elapsed, etc.)
- Portability – Must have ability to operate on different Android phones
- Safety – Make information easy to read to allow rider to be safe will utilizing the application

The application must provide the rider with a rich GUI that will display the information necessary to provide the user with an enhanced biking experience. "Fit for purpose" is we aim to give the rider the ability to choose which desired information will be displayed onto the screen. By keeping safety in mind, we will make the application simple to read to allow the rider to mainly focus on his path and surrounding environment. The application that we will create will be compatible with Android API 2.3 to Android API 4.4, which includes Gingerbread, Honeycomb, Ice Cream Sandwich, Jellybean and KitKat. Through research done we found that 98.3% of Android phones in the market utilized these APIs, focusing on those APIs will allows us to target the majority of the Android users in the market today.

#### 4.2.2.1 Use Case Diagram and Description

Figure 37 shows the use case diagram of our application. The use case diagram is used to display how the user will utilized the application and in order to set requirement of needs of the software. A detailed description of the use case diagram for the Bike Dash Android application will proceed the use case diagram show on Figure 28.

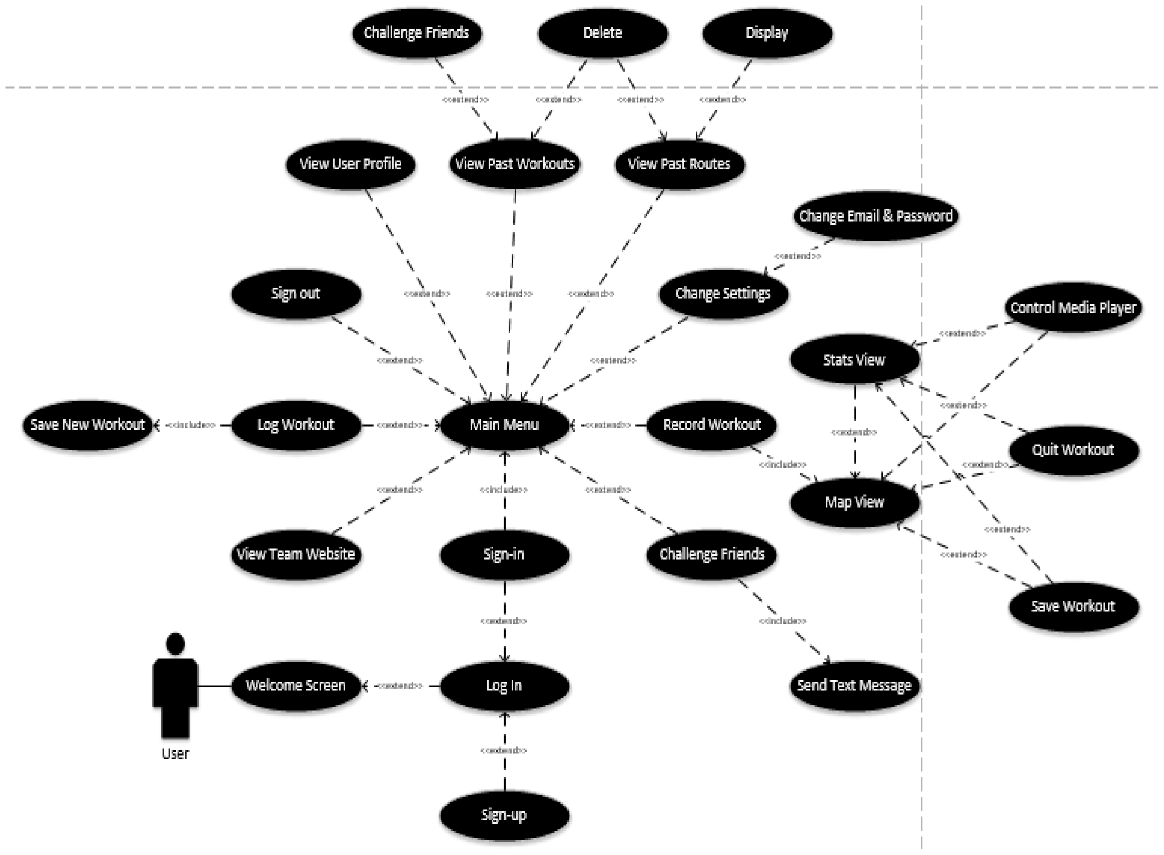


*Figure 28 - Use-Case Diagram for Android Application*

The Android application was created utilizing the Eclipse IDE along with the Android ADT/SDK plug-in. The application is limited to devices that support Android API 13 (Honeycomb) through the newest Android API 19 (KitKat) due to the fact that those APIs supports version 2 of Google Maps. The android devices used for testing were the LG L9 and the Samsung Galaxy S4 which are equipped with Android API 4.1 (Jellybean) and the Note 3 which is equipped with Android API 4.4 (KitKat). The android application will serve as the user interface that we provide to the user. The android application will receive data from the Bike Dash components via Bluetooth and use the data to provide the user with a rich GUI layout. In order to provide an optimal user experience we will provide the user with two ways of viewing the workout data collected by the sensors of the system.

The application was design so that once the user opens the application, they will be presented with a splash screen or “welcome” screen showing the application logo and a “Click to enter” button that allows the user to enter the application. Figure 29 shows an image of the layout of the splash screen. Once the button is clicked the user will be taken to a sign-in page in which new users can sign up and returning users can simply sign in and begin to use the application. Figure 30 shows the GUI layout of the sign-in page made up of two buttons and two text fields for their email and password, respectively. The sign-in page activity was designed so that it will inform the user of errors during sign in due to incorrect email and password or due to the fact the user has yet to create a profile.

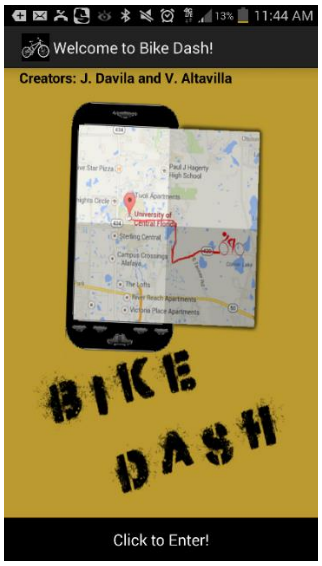


*Figure 29 – Bike Dash Splash Screen*

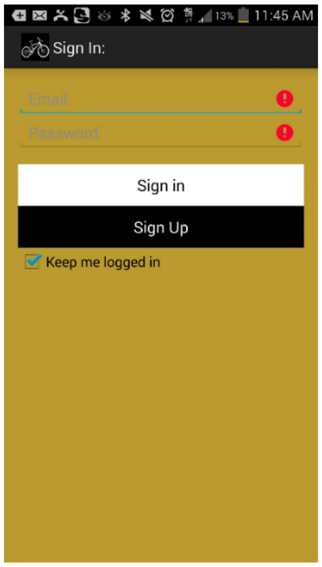


*Figure 30 - GUI Layout of Sign-in Page*

In order for users to utilize the Bike Dash application they must first sign up and create a user profile. Figure 31 shows the GUI layout that is shown to the user in order to fill out the new user form. In order to improve the usability of the application the text fields will provide a hint to the user of what information they should enter in the specific text field. Furthermore, the keyboard that's brought up for the user to use is determine by the expected data type of the input. For example numerical text fields will only bring up the numerical keyboard while text fields will show the user a full keyboard. This activity was also created to handle input errors from the user by not allowing the user to create a profile unless all data fields are filled in with the correct data input type.

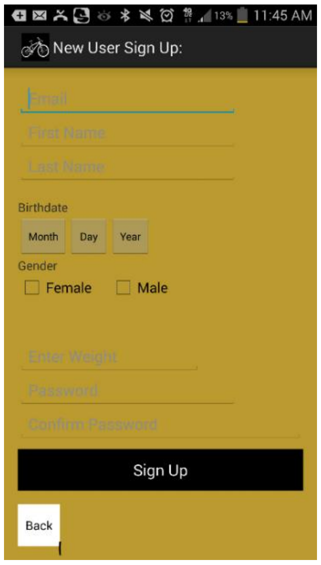


*Figure 31 - GUI Layout of New User Form*

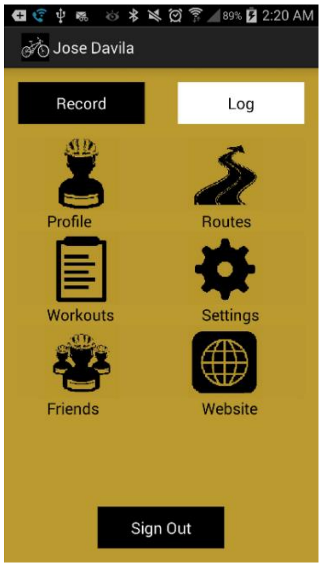


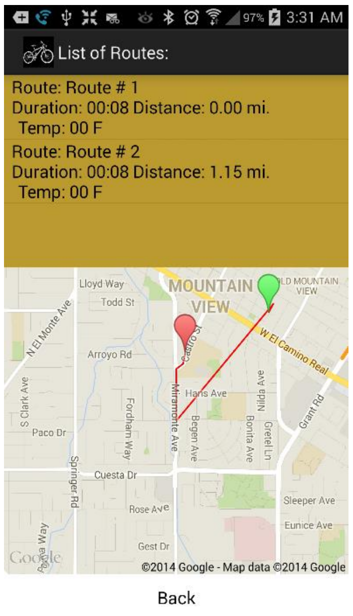
*Figure 32-GUI Layout of Main Screen*

The main activity of the application was designed to mimic the layout of some popular applications in the Android market such as Twitter. We wanted to provide the users with a familiar layout in order to increase the learnability of our system thus increasing the overall user's experience when utilizing our product. Therefore we created a rich GUI layout that's shown in Figure 41, which follows the "one click" rule for software. Allowing the user to navigate to their desired destination within the application with just one simply click. In the main application the user will have 9 options when navigating the application which include:

1. Record a workout
2. Log a workout
3. View user profile
4. View past routes
5. View past workouts
6. Change settings
7. Challenge Friends
8. View developers website
9. Sign out

The main application is made up of three buttons and six image buttons which are all clickable. When the user clicks any of the buttons the user will be taken to a different part of the application. From the main application, when the user clicks the profile image they will be able to view their profile which displays information about the user that was gathered from the sign up activity such as full name, email, height and weight. While also allowing access to the user's last workout data along with the overall total distance, workout time and calories burned. As mentioned before the data that's received via Bluetooth will be stored in the phone to later be used to access pass workout data and routes that the user may save. When the user chooses the workout image he will be brought to another activity of the application in which they will be able to view workout data and will have the ability to text that information to their friends.

The user also has an option to log their workouts into the system to allow for the application to act as a journal for their bicycle ride outside of Bike Dash. Another important design decision made for the user interface was the design of the route list. The route list was design to give the user visual feedback of recent routes, giving the user a map that plots the location points captured by the GPS sensor during that particular workout. Figure 33 shows the layout of the activity, in the figure the green represents the starting point and the red represents the ending point of the workout.


3:31 AM

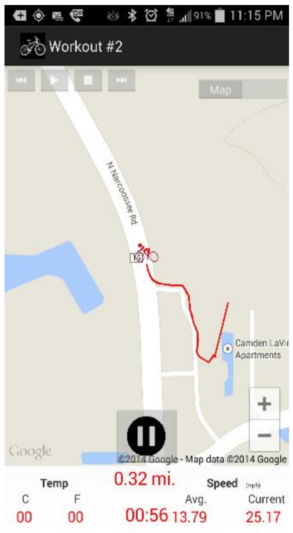
List of Routes:

Route: Route # 1
Duration: 00:08 Distance: 0.00 mi.
Temp: 00 F

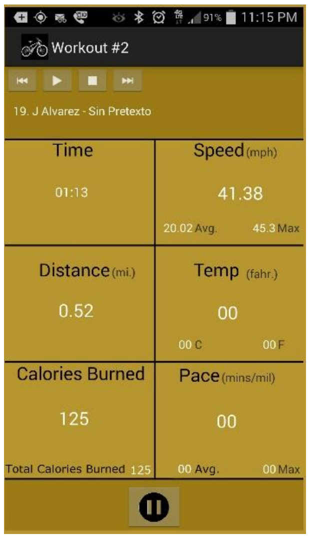
Route: Route # 2
Duration: 00:08 Distance: 1.15 mi.
Temp: 00 F

*Figure 33 - Android App Route List*

When the user choose to record a workout, the users will be brought to the map view. As mentioned before the user will be giving the option to view the statistical data in two different views a map view and a statistical view. The primary option will be a map view shown in Figure 34 which will provide the user with real-time visual representation of their current location, route and some important statistical data pertaining to their current workout such as time elapsed, distance, speed and ambient temperature in both Fahrenheit and Celsius representations. The map view was designed utilizing Google Maps APIv2 and required the registration of our Android Application officially through the Google Developer Console and obtaining an API key in order to receive support from the Google servers for displaying the map.



| Temp | | | Speed | |
|---|---|---|---|---|
| C | F | | Avg. | Current |
| 00 | 00 | 0.32 mi. 00:56 | 13.79 | 25.17 |


Workout #2

19. J Alvarez - Sin Pretexto

| | |
|---|---|
| Time<br>01:13 | Speed (mph)<br>41.38<br>20.02 Avg. 45.3 Max |
| Distance (mi.)<br>0.52 | Temp (fahr.)<br>00<br>00 C 00 F |
| Calories Burned<br>125<br>Total Calories Burned 125 | Pace (mins/mil)<br>00<br>00 Avg. 00 Max |

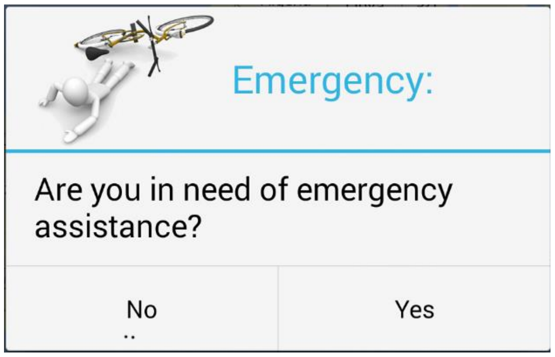
*Figure 34 - Map View*

*Figure 35 - Stats View*

The secondary option will provide the user with far more statistical data about their current workout.  Along with some of the data present in the map view, the statistical view will also boast the calories burned, the pace in minutes per mile and the maximum and average value for each category.  Figure 35 shows the statistical view that will be presented to the user.

With the android application we aimed to provide the user with motivation to improve their workout experience and allow them to workout longer and more often.  Therefore the user interface includes an essential necessity for every workout, a music player.  The application searches for all media files with the extensions MP3 and WMA located on the SD card of the Android device, retrieves them and stores them in an array of songs.  Which then allows the user to control the music stored on their phone directly from the application without ever having to navigate outside of the application.  Figure 43 and 44 show the music player control buttons on the top-left corner of the screen.  Furthermore we provide the user with the ability to share their workouts with their friends via text message.  The application accesses the contacts that are stored in the phone (SD card, phone storage, google contacts, etc.) and turns it into a list displaying the friends name and phone number.  In which the user can then choose which friend they would like to "challenge" to beat their workout time or distance.

By utilizing the accelerometer and monitoring the data coming into the android phone via Bluetooth we were able to provide rider's with the ability to detect crashes.  Whenever the rider is involved in an accident that causes him to drop his bicycle to the floor, the application automatically detects the accident.  In order to act accordingly the application alerts the user via an alert dialog and audio feedback, informing them that the system thinks that an accident has occurred.  If it has, the system will automatically dial emergency services along with send an emergency text message containing your precise location via longitude and latitude.  Figure 36 shows an example of the emergency alert dialog displayed to the user.


Emergency:

Are you in need of emergency assistance?

Figure 36 - Emergency alert dialog

## 4.3 MCU and Sensor Integration

One of the most integral parts of our design process is integrating our MSP430 microcontroller with all of our sensors. In doing this process, we must be sure to adhere to strict ESD (Electrostatic Discharge) guidelines as not following these procedures could result in system damage. To keep from damaging our system, we will incorporate the use of both an ESD work mat as well as an ESD bracelet. To aid in the storage of our equipment and parts, we will acquire ESD boxes capable of protecting our items similar to the one show below (Figure 37). Using these bins will provide sufficient storage space as well as ESD protection and because they are stackable, they will help to maintain workspace cleanliness.

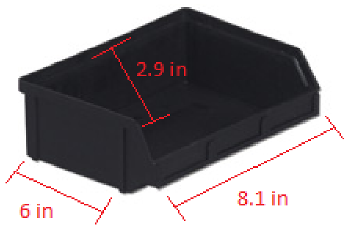


*Figure 37 - ESD Bin*
*(www.LewisBins.com)*

**TEMPERATURE SENSOR:**

The temperature sensor we are using will require us to connect the output of the sensor to an ADC channel of our microcontroller. To get a proper temperature reading, we will run a voltage to the temperature sensor where it will incur a resistance. At the output terminal, a voltage will be passed in to the ADC channel (pin 19 – RA0/AN0) where it will be converted from an analog signal to a digital signal. The digital signal will then be ready for processing either in the microcontroller or from within the Android application.

**GPS MODULE:**

To integrate our GPS module with our microcontroller we will first have to connect our module to a power source. This power source will be a general line run for the entire MCU and sensor system. To connect the data to our microcontroller, we must connect both of the GPS module's UART lines to the microcontroller through its UART channels. The two pins on the MSP430G2553 that handle UART data are pins 9 and pin 7. For reception of data to our microcontroller we must use pin 9 as it is designated as a UART receiving channel. For transmission of our data, we will connect the other UART port on our GPS module to pin 7 on our microcontroller which is designated as a UART transmission channel, however, for our application will will not connect the RX pin of the GPS unit.

**<u>BLUETOOTH MODULE:</u>**

For our Bluetooth module, we must ensure that voltage in remains within the range 2.2 V to 4.8 V. To do this, we will first run the power source to a 3 V voltage regulator and then pass the voltage through to our Bluetooth module. This should ensure that the voltage received is continuous and within the acceptable range. The next step in integrating the Bluetooth module is to run the transmitting and receiving lines to the appropriate channels of the microcontroller. The pins that we will utilize on the microcontroller are pins 9 and 7. Pin 9 will be used for the receiving channel, which will be connected to pin 2 on the Bluetooth module (this is the Bluetooth module's Tx pin). Pin 7 of the microcontroller is designated as the transmitting pin and will be connected to pin 3 on the Bluetooth module (pin 3 is the receiving pin on the Bluetooth module). The pin diagram for the Bluetooth module can be seen below in Figure 38.

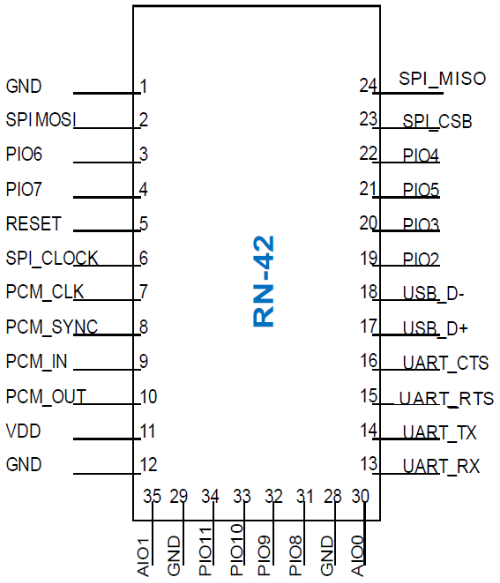


*Figure 38 - Bluetooth Module Pin Diagram*
*(Roving Networks)*

**<u>ACCELEROMETER:</u>**

To integrate the accelerometer into our system through the microcontroller, we must connect the accelerometer to ADC pins. For the accelerometer, we will utilize pins 18 and 17. Both of these pins are ADC and will accept the analog voltage output by the accelerometer. The voltages that the accelerometer outputs are specific to the x and y measurements of acceleration. The accelerometer delivers the analog signal through pins 10 and 12. Pin 12 is used for the measurement of acceleration in the X direction and pin 10 outputs a voltage relative to the acceleration in the Y direction. These measurements also have the ability to be converted into tilt and roll, which we will use to determine the inclination angle of the climb. The accelerometer must also take in a voltage that falls within the range of 2.4 V and 5.25 V so in order to keep a continuous voltage to our sensor, we will make use of a 3 V voltage regular connected in series.

## 4.4 Printed Circuit Boards

Below are the two board schematics created for production of the PCBs. To construct the PCBs, we used Cadsoft Eagle and created a board output file. These board output files were then sent to Advanced Circuits, a PCB manufacturer, where they produced the printed circuit boards. Once we received the circuit board, we mounted and soldered all of our parts according to the schematics created during the Design phase.

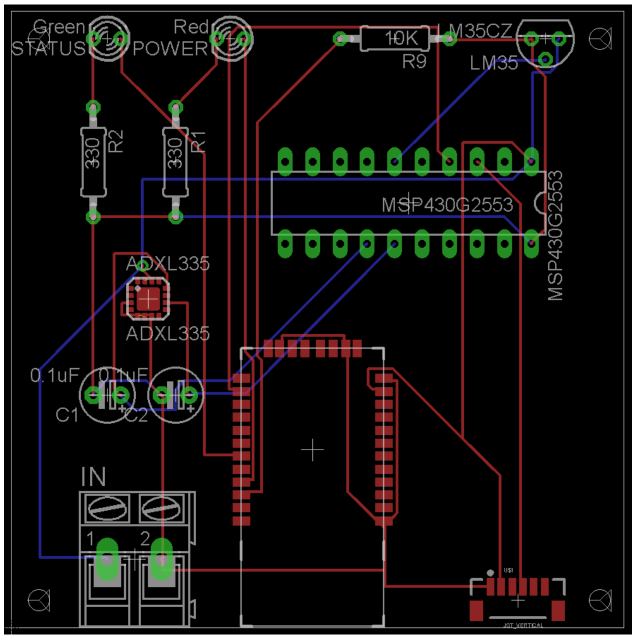


*Figure 39 - Bike Dash Board Layout*

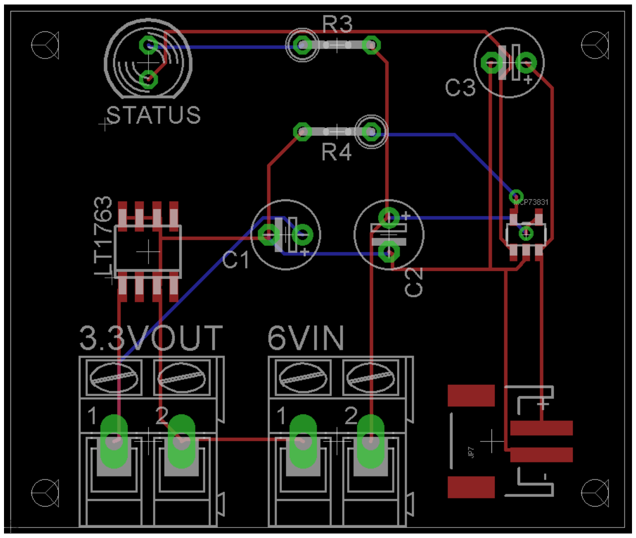


*Figure 40 - Charging System Board Layout*

## 5.0 Project Prototype Building and Coding

To make sure the system meets all performance requirements, each member of the group will test every sub-system separately. Each of us is responsible for that particular module. We will develop a logical approach for testing. A set of criteria will be collected for every hardware and software component in the system. Then we will prototype all the hardware and software designs and perform test runs on every component or a device and check whether performance results satisfy the stated criteria. Any mistakes will be examined for errors and faults, modified accordingly and tested again. This process will continue until every system passes all of the test runs. Once we are certain that every component is working correctly, they will be merged together and embedded into a single platform. The final phase of the test involves a full test run under different circumstances that will evaluate accuracy and overall performance of the bike dash.

## 5.1 Parts Acquisition

The parts for bike dash project will be acquired over the period of December 2013, early January 2014. Most of the parts we need are available. We will purchase all of the components needed online. Though, we are considering several suppliers and looking for the best prices we can get on each part. The table below shows the each component supplier that will utilize in bike dash, along with the model number from the specific manufacture.

| Parts Acquisition | | |
|---|---|---|
| **Component** | **Model** | **Supplier** |
| **Micro-controller** | PIC18F14k22 | Microchip Direct |
| **Power Generator** | Shimano Alfine DH-3N80 | www.excelcycle.com |
| **GPS** | GMS-HPR | Global top technology inc. |
| **Bluetooth** | CC2564-PAN1326 | Ebay.com |
| **LCD** | CFAH1604B | www.crystalfontz.com |
| **Temperature Sensor** | LM35 | Texas Instrument |
| **PCB** | | PCB Express |
| **Hearth Rate Sensor** | | |
| **Accelerometer** | ADXL362 | TI |
| **USB (cable)** | 3.0 | Discountechnology.com |

*Table 23 - Parts Acquisition*

## 5.2 Prototype Building

When it comes to building the bike dash, each system inside will be constructed separately on its own board for testing in the design lab that located in Harris lab building. Every sub-system will be debugged to see if it is working properly. It is necessary for us to make sure the construction of the device is functioning correctly as well. The separable systems to be built are the voltage regulation circuits and power supply. Other devices such as temperature sensor and GPS will be interfaced one at time to make sure proper functionality. First the power supply circuit will be assembled on its own power board. Current and voltage reading will be taking to attain the transfer function of the circuit. The LCD screen and microcontroller are the next segments to be interfaced. This is done so they will be able to tell us if our circuit is working accurately on the data transfer level, which will be necessary to the subsequent devices to be integrated.

We will be using a USB port installed in the front side of the bike. This port will serve as power source as we described in Section 4.1.7. It will be used to power on devices, also will charge the battery of devices such as cell phone. This indicates that the bike is generating the power to charge the battery, and at the mean time we can use a portion of the battery to manage other devices that support USB power sources. This will makes user to be more ease when new devices need to be installed. Once the MCU and LCD are working appropriately we will be able to display information coming from GPS module, temperature sensor, and the heart rate sensor as well in addition to the power being generated in the power supply. This will be accomplished with some testing procedures. The next step is to place the LCD and power circuit board in a plastic cover on the front handlebars of the bike. It will have the main power switch and LED indicating whether it's on or off. Then, the LCD will be housed so that it is level with the top of the housing. The rest of the components inside will be protected from the elements since it will be exposed to the outdoor environment on a continuous basis. The purpose of this house or cover is to protect to circuit from water, dust.

The bike dash housing will be attached to the handlebars using a quick mount as pictured in figure below. Before mounting the circuit board inside the housing, the mount will be attached to the box. The mount can simply be fixed on any apparatus with the same size. It also can easily be adjusted depending on the viewing angle. Known that power cable runs from the circuit board housing to the generator, it will be attached to the underside of the main truss with locks and enough restraint in the front allowing the bike turn freely.

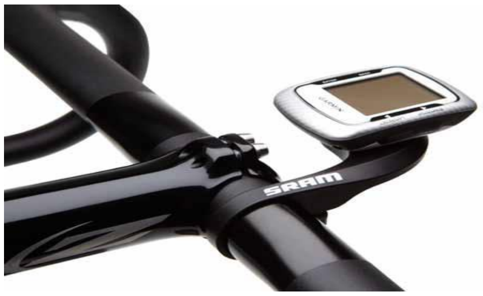


*Figure 41 - Quick Mount*

## 6.0 Testing

For the system testing phase of our project we will attempt to subject our design to many test cases in order to locate and fix any errors that may exist in the system. The following section will outline the test procedures for both the software and hardware portions of our design. The purpose of this section is test the system as a whole and ensure that all parts have integrated properly and are able to communicate consistently. For example, in order for the user to receive a GPS coordinate we must ensure that the GPS can communicate with the microcontroller and that the microcontroller in turn is able to communicate with the Bluetooth module. The Bluetooth module must then be able to communicate (or broadcast clearly) with any connected Android devices. To differentiate between hardware testing and software testing we will consider hardware to mean any physical component connected directly to the microcontroller or the PCB board. These are to include but are not limited to the dynamo, temperature sensor, GPS module, Bluetooth module, ect. Hardware to be included in testing will be the enclosure that houses the above listed components (excluding the dynamo). For these pieces, we will test the durability and weatherproofing. Software is to be considered solely as the Android application. Though the software component of the microcontroller is considered a major software component in our system, it is assumed that it will be sufficiently tested while testing the hardware components of the system (excluding the dynamo). The reason we can come to this conclusion is because it is necessary to utilize the software on the microcontroller in order to test the integration of the hardware.

## 6.1 Hardware

To test the hardware components of our system, there are a few tools that we will need to use. The tools that we will use for purposes of testing and debugging the hardware are as follows:

- Oscilloscope – To view the waveform produced by electrical components
- Multimeter – To measure voltage and current of various components
- PIC3Kit – A debugger/programmer specific to PIC microcontrollers
- Windows-based PC – To view contents of PIC and to run debugging software
- Bluetooth enabled device – To test the Bluetooth module’s connectivity
- Debug Program – A self-made program written in C# to monitor data over Bluetooth

When it comes to the testing of operational hardware it is important stage for any engineering project. Without a thorough testing, the product is almost worthless regardless of the source of its origination. Similarly, the most reliable and respected corporations, manufacturers and university inventors implement difficult testing and quality control procedures. There many factors that influence the difficult testing and endorsement those professionals include in their decision-making processes and, yet the choice of this bike dash project is intended for senior level university students.

Reputations can quickly be made, both negatively and positively, on the result and presentation of a body of work. Whether it’s a university level senior design project or a new contour of microcontrollers by Texas Instruments. Thus, if the end creation turns out to have a lot of problems, the professor or buyer will be lead to the idea that there were mistakes in the testing procedures.

### 6.1.1 Description of Test Environment

Our device will utilize rechargeable batteries as the project specifications specify that the device must run on 6 volts. The Environmental issues we will face are, quality and safety issues are always of concern. So, for that reason the battery that we considered was the one with the least toxicity. The memory effect indicates to having damage when the batteries are not discharged and charged completely. As individuals are less likely to completely run the rechargeable batteries down before recharging, by having a device run automatic discharge cycles will assist maintaining battery life. The rechargeable batteries generate more than enough power to support the whole circuit. The purpose is to deliver appropriate power to all parts of the system. The output currents go as high as 1200 mA while using a single cell Li-Ion battery, and discharge it down to 2.5V or lower.

The converter we are using is based on pulsewidth-modulation (PWM) controller using rectification to attain maximum efficiency. At low load currents, the converter enters power saving mode to maintain high efficiency over wide-ranging load current. Thus, all parts of the device are adequately powered.

### 6.1.2 Durability Testing

The most important type of testing for this project will be during different types of weather conditions. This is an important factor because of the fact that the consumer will want to use this device in most areas (mostly outside for cycling), and weather has to be consideration when thinking about this. The device must be able to withstand most types of weather conditions. What we will do for this is to test our powered tracking device for its ability to handle itself even in the rainiest of conditions, because if it can’t handle itself during those types of conditions then it won’t hold up well for us in the long run. This durability testing will test our LCD to see if there are well-built for our project as well as our sensors to be able to work during these conditions even if there is no sunlight being provided to the powered tracking device. We must finalize our design for these types of conditions so that it won’t delay us on presentation day. With this durability testing, there will need to be taken into consideration what type of material will be best for our testing purposes. Through the dependency of the device as well as how it performs throughout our step-by-step building sequence of this project will show to us how well this project will hold out in the future. This is important mainly when we have to take other clients into consideration so that this project will not just work when we are testing it but for other people as well. Once our durability testing is all-inclusive, and then we will be able to know that our project is stable enough to enter into the real world. The only worry as of right now is if our cost of the total project will support our amount of funding requested. The funding is also an aspect that will determine the durability of the project since the amount of funds will determine what kinds of material we will be able to acquire.

## 6.1.3 Test results

The testing of equipped hardware is very important stage for our project. Without thorough testing, the product is nearly worthless regardless of the source of its origination. Similarly, the most credible and respected corporations, manufacturers and university inventors implement several testing and quality control procedures. There many elements that influence the testing and authentication those professionals include in their decision-making processes and, yet the scope of this bike dash project is intended for senior level university students. The motivations for delivering a quality product reflect those of industry giants like Texas Instruments, ST Microelectronics and Linear Technologies.

Reputations can quickly be made, both negatively and positively, on the outcome and presentation of a body of work. It doesn't not matter who make a project, if the end product turns out to be full of problems, the professor or buyer will be lead to the idea that there were mistakes in the testing procedures.

**DEBUG PROGRAM:**

The debug program that we will create for this project will be all original and will act as a primary source of receiving raw data. The program will simply allow the programmer to connect to the microcontroller and monitor the data sent and received by the sensors using the onboard Bluetooth module. The program will be written in C# as using this language will provide for a fast production time allowing more time for debugging. A screen shot of the user interface can be seen below in Figure 42.

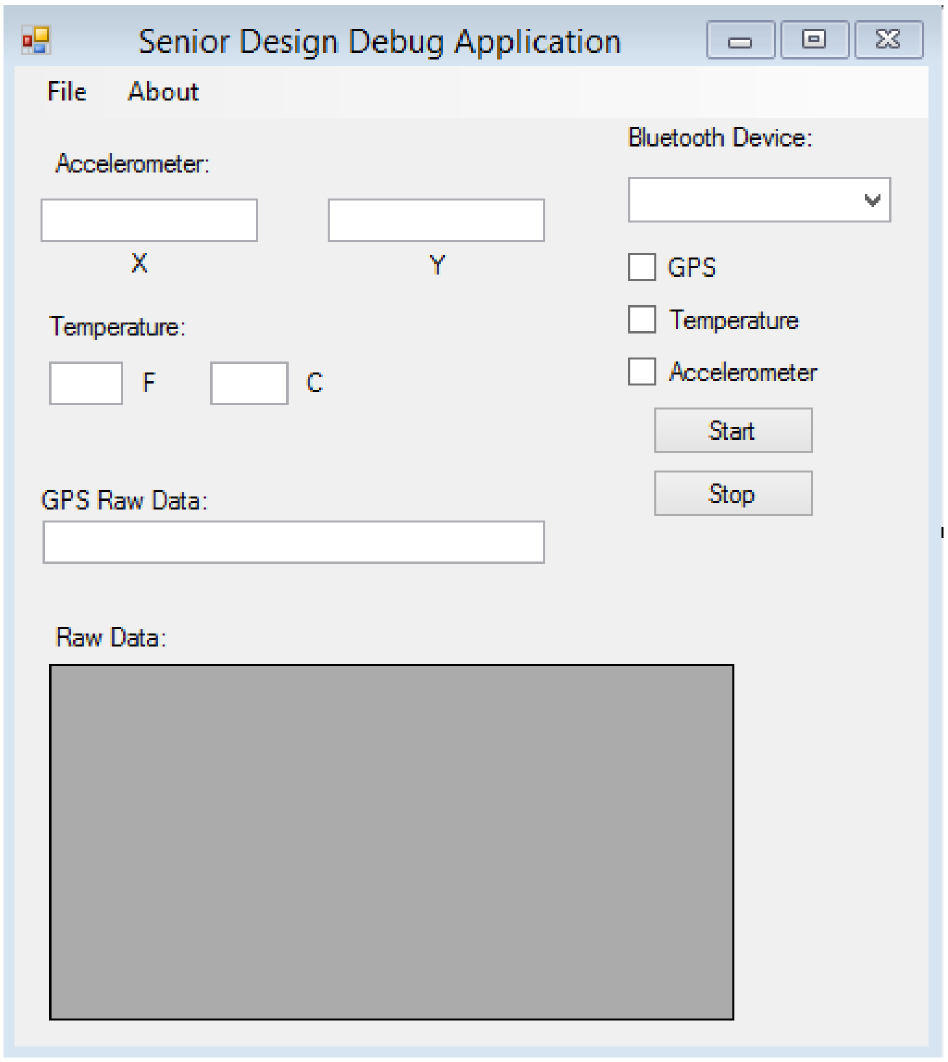


*Figure 42 - Debug Application*

To use the debug application, the user would simply check the box next to the sensor they wished to populate, select a Bluetooth enabled device from the dropdown menu, and hit the start button. This should populate the sensor readings in the appropriate box as well as provide the user with an option to see the raw data produced from the microcontroller and sensors. These same features (minus the ability for raw data) will be available in the Android application in a more user friendly manner. This application will serve strictly as a debugging application and will not be included in our final project. For our final product, we intend on reading out all of the sensor data into the Android application and presenting it as a “dashboard”.

**<u>POWER:</u>**

The first set of tests (and possibly the most essential series of tests) are the system power tests. These tests will test the voltage and current draw of the system and its components. The first place we will test the voltage is at the rear wheel where our dynamo is located.

We will connect the positive and negative leads to the dynamo and pedal (or simulate pedaling) and measure the voltage and current coming out of the dynamo. We expect for the voltage coming out of the dynamo to be at or near 6 V with a power of at or near 3 W. If we do not achieve these results, we will investigate the testing methods of the dynamo as well as determine if the wiring has been done properly. The next power test that we will run is to ensure that each of our components on the PCB board is receiving power. To do this, we will utilize the multi-meter and measure the voltage across the element (or at $V_{in}$ depending on the component). The specific voltages for each of our sensors can be found in the design section of this report.

## <u>**COMMUNICATIONS:**</u>

The next set of tests that we will run during the testing phase are our communications tests. Our communications testing will consist of testing all of our devices that are required to send and receive data. This is to include, but not limited to the GPS module, the Bluetooth module, and the microprocessor. The communications testing is another very important step in our testing process as this step will also aim to test the integrity of the data on the microprocessor side of the Bluetooth module. The first, and most important part, component that we will test first is our Bluetooth module. To test our Bluetooth module, we will .The part of our communications system that we will test next will be the GPS module. For the GPS module, there are a few things that we must test. First, as we did in the integrity testing, we must test the GPS to ensure that we are receiving signal from at least two satellites. To do this, we will output the GPS raw data in our debug application and examine the "position fix" section of the string. We can see the position fix segment highlighted in the sample string below (Figure 43).

```
$GPGGA,064951.000,2307.1256,N,12016.4438,E,1,8,0.95,39.9,M,17.8,M,,*65
```

*Figure 43 - Sample GPS Raw Data String*

In order for our system to be able to acquire an accurate position, we must have at least two satellites visible and locked on. The reason we must have at least two satellites visible is due to the triangulation algorithm used in calculating the coordinates. In the above example, we are receiving one satellite and therefore would not be able to calculate proper coordinates. The ideal connection is A3, or a connection to three satellites. If we receive any less than two satellites during our test, we will investigate any obstructions or the possibility of disconnect between the GPS module and the antenna. The next test that we will run would be to verify the accuracy of the GPS raw data. To do this we will use the coordinates provided by the string and match them to our location using a handheld GPS or mobile device with GPS integrated. It is more likely, however, that we will use the latter. If we match the coordinates that the GPS enabled mobile device produces to the coordinates output by our GPS, we can consider the test a pass. If we do not receive correct coordinates, we will again examine the ability of our device to gain the attention of at least 2 satellites. It is possible though that we may not receive an accurate reading from our GPS module without and A3 (or three satellite) connection. The last method that we will use to test our GPS module's integration is to verify that the coordinates and information received by the GPS module is the same data set received by our application (or mobile device). To do this, we will use the Launchpad to monitor the data received by the MCU and compare the raw data string to it. If the data strings do not match, and we have run the above tests to verify integrity, we must investigate the data being stored in the MCU and the manner in which it is being transferred to the PC.

The other part of our communications system that we will test is the Bluetooth module. When we test the Bluetooth module, we will be looking to examine both the signal and the data transmitted. In order to test the strength of the signal, and to ensure our device acts accordingly, we will test our device in both and open space as well as an area with multiple obstructions. The distance of the Bluetooth is estimated to reach upwards of 33 feet. In order to verify this, we will mark out 33 feet and measure signal strength at various intervals. For example, we may decide to take a reading every 5 feet. With this data, we will be able to construct a plot of strength vs. distance and compare it with our expected results.

**ACCELEROMETER:**

For the accelerometer testing segment, we will verify that our accelerometer is producing accurate results relative to our device. We will ensure that we are receiving proper zero coordinates (when the device is stationary on a level surface) as well as receiving proper measures of acceleration during both free fall and travel. First, we will test our accelerometer measurements while at a zero position. To do this, we will place our integrated device on a level surface and ensure that both the X and Y directions have accelerations of 0 m/s$^2$. We will use the debug program in order to monitor the output from our accelerometer sensor during the testing period. If we find that our sensor is not reading zero, we will attempt to recalibrate the sensor until we are able to receive a proper zero reading. Next we will test the measurement of our accelerometer during free fall. To do this, we will use the drop test. Using the drop test will include enclosing the device in a padded enclosure (or having a padded landing pad) and dropping the device from x distance. We will then monitor the output on the sensor to see if our device measures the expected 9.81 m/s$^2$. If the device does not output a measurement equal to that of the standard rate of acceleration due to gravity, we will attempt to recalibrate our sensors and retest our system.

## 6.2 Software

For the software testing segment, we will be testing the Android portion of our software component. For our Android program testing, we will test each module individually as well as test the system as a whole. To test our Android program, we will use multiple environments to ensure that our program is cross compatible. The first environment that we will test our Android application on is and Android simulator that is provided with the Android SDK. Using this Android simulator will afford us the opportunity to test multiple APIs without having to have physical devices containing these APIs. It will also allow us the ability to change device configurations easily without affecting any of the group members' personal devices. We can see an example of the Android simulator below in Figure 44. The Android simulator that we are running is utilizing Android 4.3 (API Level 18) as we want to make sure first and foremost that our application works properly with the newest version of Android. As we progress in our application testing, we will work our way through the APIs to ensure we can run our application on previous versions of Android. Though we intend on offering support for previous versions of Android, we will not test our application past Android 4.0 (API Level 14) and cannot guarantee the operation of our application on those systems.

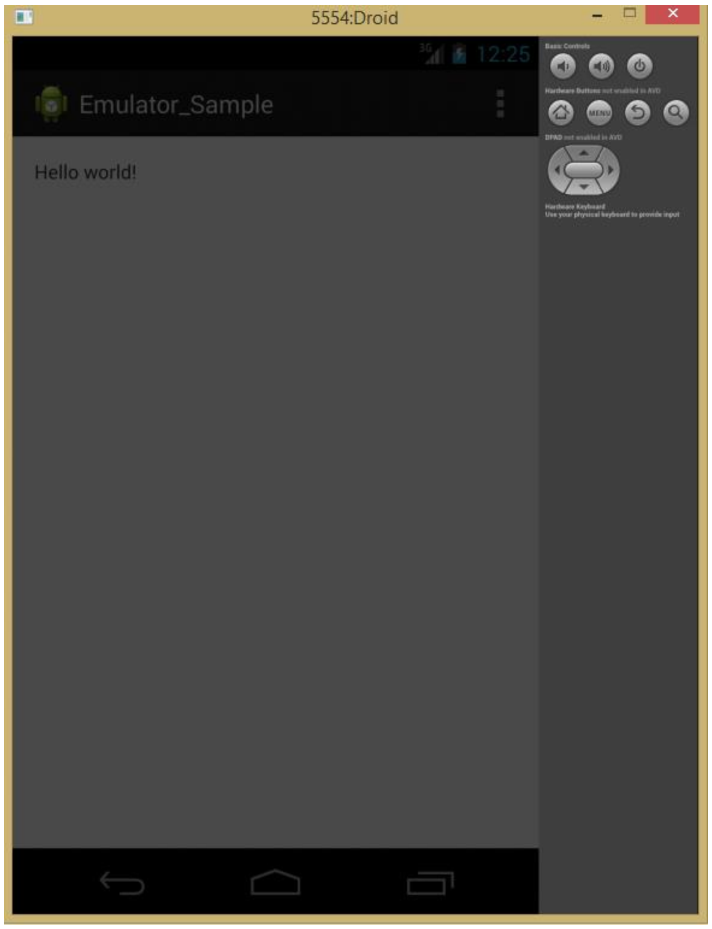


*Figure 44 - Android Simulator Running Hello World Program*

The other way we will test our application is by using physical Android phones. The two phones we will use are the Samsung Galaxy S4 and the LG Optimus L9. Though the two phones are different, they both contain the Android operating system but offer differences in the API levels which will allow us to test our application across multiple physical platforms. A side by side comparison of the two phones and the AVD (Android Virtual Device) can be seen below in Table 24. To test our application on these devices, we will connect our mobile device via a USB cable while running the Android SDK. Once we have connected our device, we compile and run our application and select our connected Android device as the appropriate device to install the .apk file (to use a virtual device, we will select AVD and create a virtual device). The file will be installed on our device and the application will run. From here we will proceed with our testing. After the device is disconnected, the application will still remain on the Android device. We may also place the .apk file on the phone or microSD card and run it directly from the phone.

| | Samsung Galaxy S4 | LG Optimus L9 | Android Virtual Device |
|---|---|---|---|
| **Processor** | 1.9 GHz Quad-Core | 1 GHz Dual-Core | 1 GHz Dual-Core |
| **Screen** | 5" HD AMOLED Touch Screen | 4.5" qHD Touch Screen | 4.7" |
| **RAM** | 2048 MB | 1024 MB | 768 MB |
| **Battery Life** | 2600 mAh | 2150 mAh | - |
| **Operating System** | Android 4.3 – KitKat | Android 4.2 - Jellybean | Android 4.1 – Ice Cream Sandwich |

*Table 24 - Side by Side Comparison of Testing Environments*

### 6.2.1 Application Testing

For verification of our applications integrity, we will utilize functional testing procedures. Using this methodology will allow us to ensure that our application (at a bare minimum) meets the requirements and expected results of our initial design. For the functional testing of our device we will begin by testing the primary features and functions of our application and making our way to less important, but still desirable features. The first test that we will run on our application is the system health test.

**SYSTEM HEALTH TEST:**

The system health test is our primary test and will test that our application boots up properly, without error, and allows the user to interact with the top level menu. In this test, we will run the application from the desired device. Once we have reached the menu screen we will test that we are able to enter all sub-menus (such as settings, sound, attachments, etc.) and that we are able to enter all system functions. We will attempt to enter each area and back out to ensure that no errors will occur. We expect all menus and sub-functions to be available to the user and that the application will boot properly.

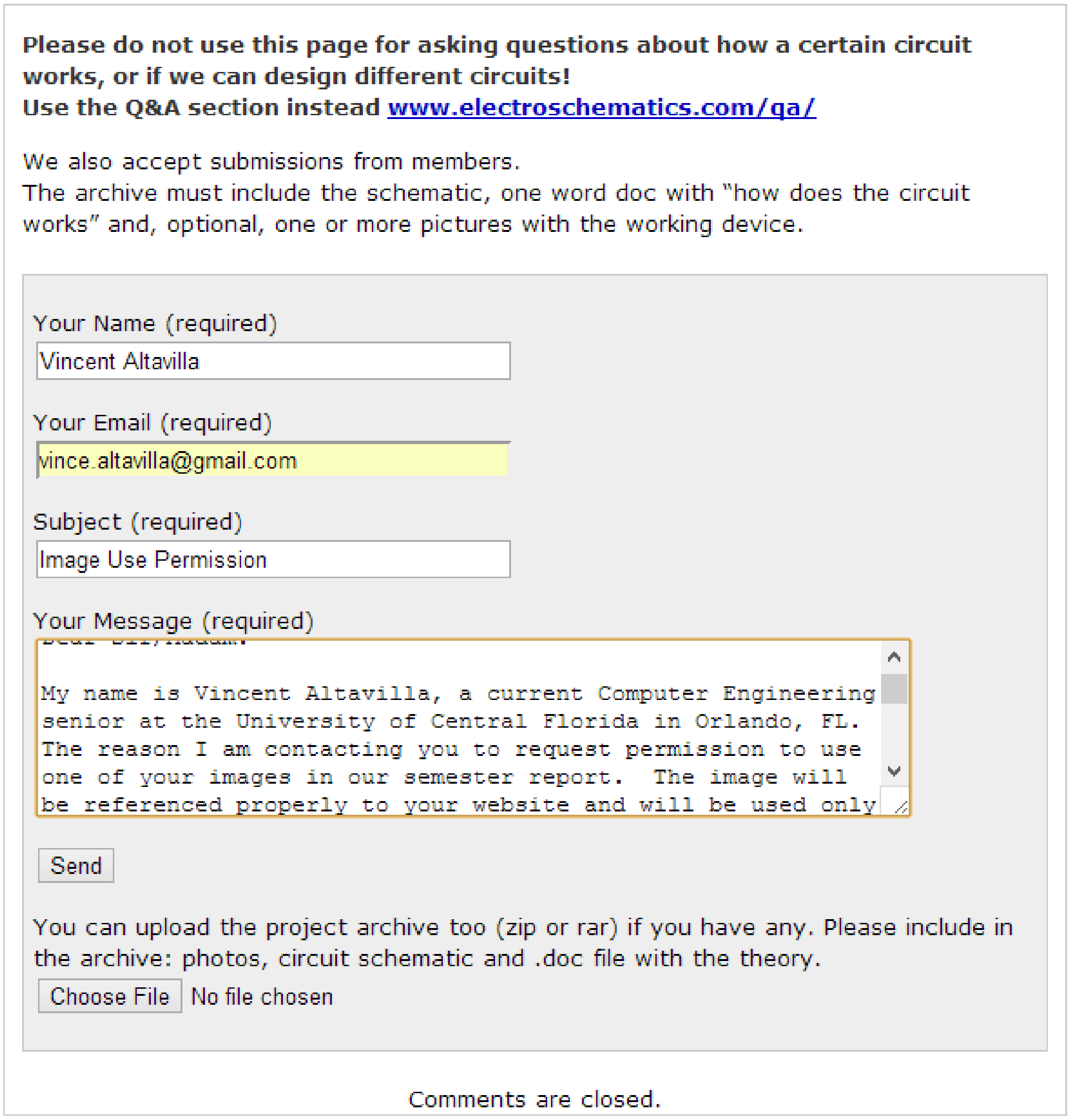
**FUNCTION TEST(S);**

In our function testing section, we will test the availability and usability of each of our individual functions. We will test the primary application functions such as creating or entering a workout, saving and reviewing previous workouts, data integrity, and the ability to switch between a basic and full view in real-time. We start our function testing by first attempting to create/record a new workout. Once we begin the workout we will check to see that the display is showing all relevant sensor data and that it is displaying properly. While we are running our workout test, we will also test the ability of our application to switch between the Basic and Full View modes in real-time. We will verify both the functionality and data integrity in each of our views. Once we have tested both views we will end our workout and attempt to save it to memory. Once we have created and saved our first workout, we will be able to test our previous workout recall feature. This will entail viewing our previous workouts and viewing the data associated with the workout. Once we have recalled our previous workout we will compare the data to the original data recorded from the workout. If both sets of data match, we will consider our workout feature to be functional. The last function that we will test is the quit application function by exiting and reentering our application. We will consider this test complete if we are able to retrieve previous workouts after successfully quitting and restarting the application.

# 7.0 Project Summary

Our end goal with this project was to offer an attachment to a bicycle that would give the user feedback on their ride. It was intended that we would appeal to riders of all skill levels and encourage the population to become more active in their life. It would allow users to count calories better, encourage improvements, and empower users to get out and exercise. Bike Dash was designed to be light and portable so that it would not impede the riders' normal range of motion and travel. One of the most difficult tasks when designing our product was trying to provide the best product for the lowest price. To overcome this, we examined cost vs. performance and made our decision with respect to primary and secondary feature requirements. Overall we expect the project to offer a simple implementation as we have also picked parts that are known to integrate well with each other. In doing this we hope to increase efficiency during the build stage of our design. During our integration process we expect to utilize UCF's labs as well as home work benches to accomplish soldering and mounting tasks. We will build our enclosure by hand in an attempt to keep costs at a minimum. To accomplish a task such as this project requires a lot of team work and in order to keep a team working efficiently, a project management plan must be in place. The reality of the situation is that no project management plan is perfect, but adhering to a timeline can save headaches down the line. We have made sure to follow our Senior Design One milestone chart and intend on achieving similar results in Senior Design Two. The end product of this project will result in a fully functional, fully ride-able bike dashboard system, complete with a power management system. We expect to have the project built, tested, and ready for delivery by April 17, 2014.

## Appendix A – Copyright Permissions



**Please do not use this page for asking questions about how a certain circuit works, or if we can design different circuits!**
**Use the Q&A section instead www.electroschematics.com/qa/**

We also accept submissions from members.
The archive must include the schematic, one word doc with "how does the circuit works" and, optional, one or more pictures with the working device.

Your Name (required)
Vincent Altavilla

Your Email (required)
vince.altavilla@gmail.com

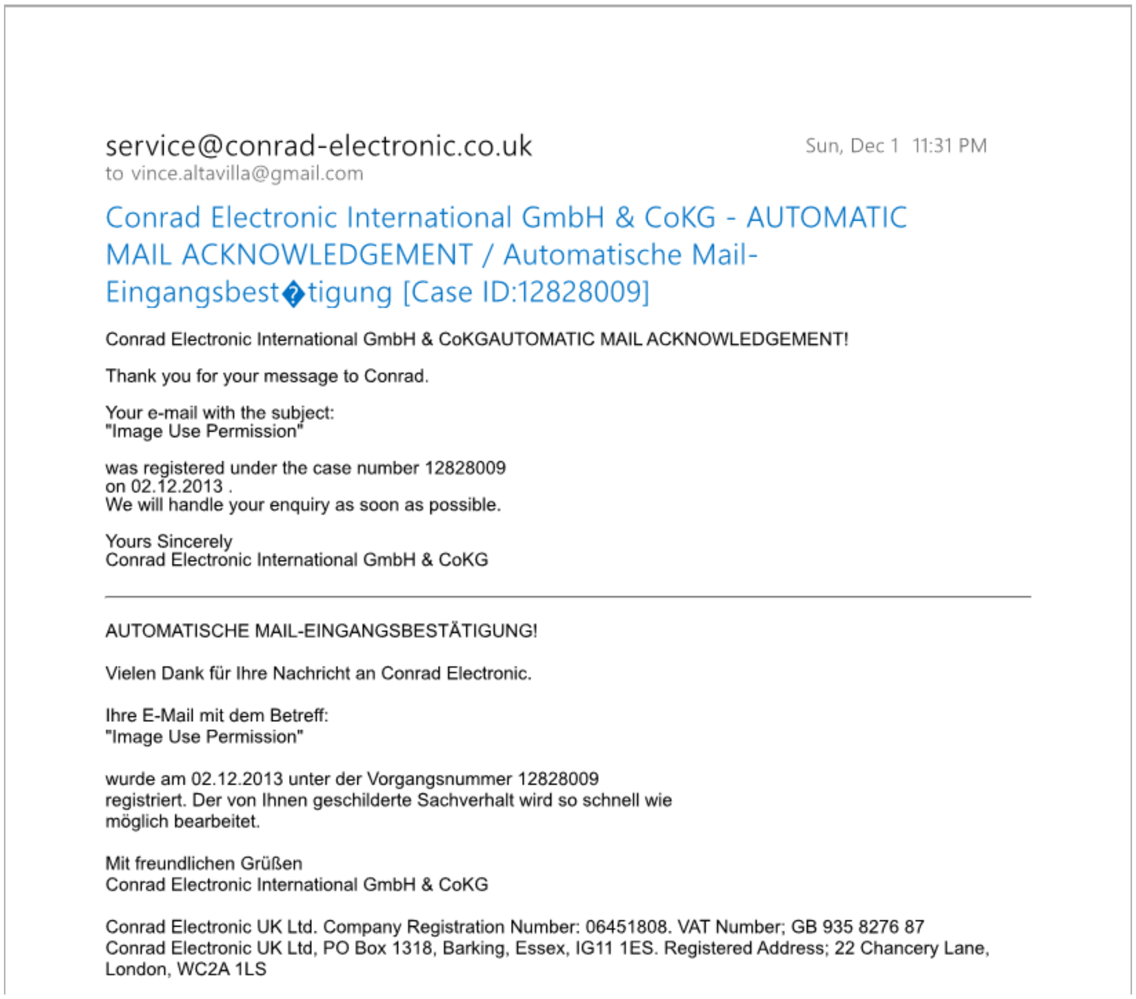
Subject (required)
Image Use Permission

Your Message (required)

```
My name is Vincent Altavilla, a current Computer Engineering
senior at the University of Central Florida in Orlando, FL.
The reason I am contacting you to request permission to use
one of your images in our semester report.  The image will
be referenced properly to your website and will be used only
```

You can upload the project archive too (zip or rar) if you have any. Please include in the archive: photos, circuit schematic and .doc file with the theory.

Comments are closed.



service@conrad-electronic.co.uk Sun, Dec 1 11:31 PM
to vince.altavilla@gmail.com

Conrad Electronic International GmbH & CoKG - AUTOMATIC MAIL ACKNOWLEDGEMENT / Automatische Mail-Eingangsbest�tigung [Case ID:12828009]

Conrad Electronic International GmbH & CoKGAUTOMATIC MAIL ACKNOWLEDGEMENT!

Thank you for your message to Conrad.

Your e-mail with the subject:
"Image Use Permission"

was registered under the case number 12828009
on 02.12.2013 .
We will handle your enquiry as soon as possible.

Yours Sincerely
Conrad Electronic International GmbH & CoKG

---

AUTOMATISCHE MAIL-EINGANGSBESTÄTIGUNG!

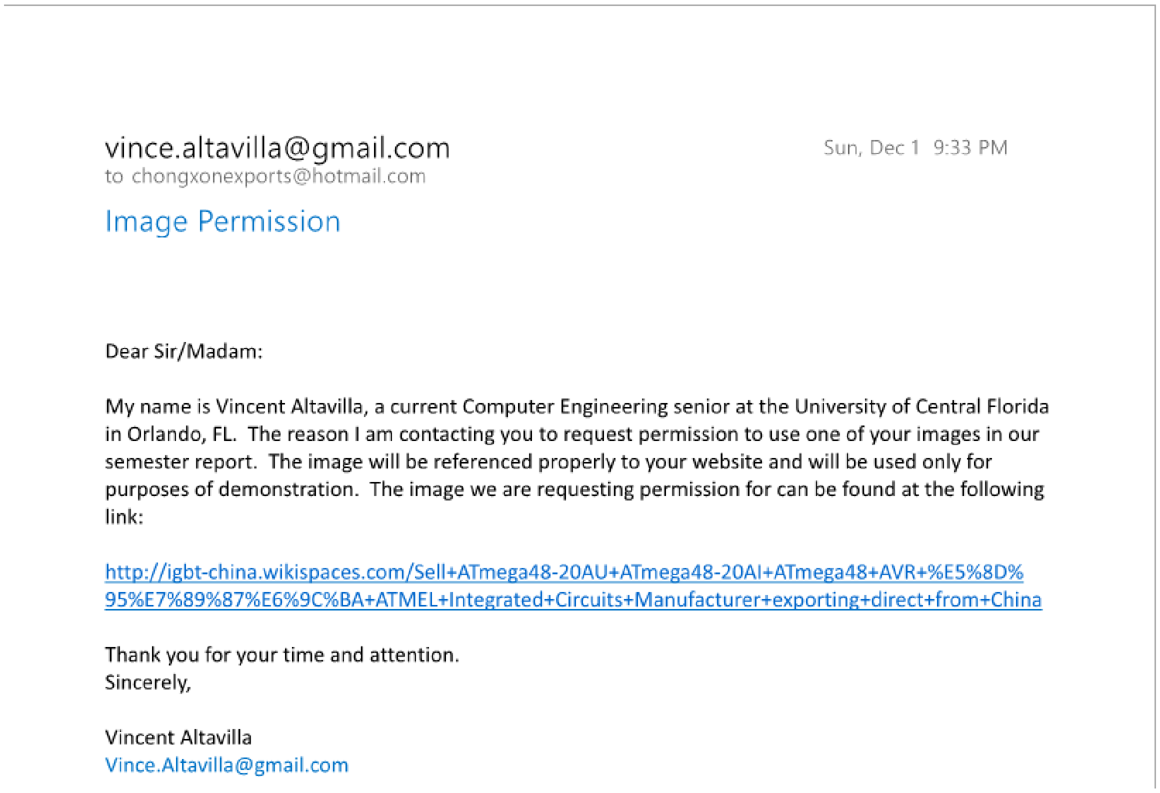
Vielen Dank für Ihre Nachricht an Conrad Electronic.

Ihre E-Mail mit dem Betreff:
"Image Use Permission"

wurde am 02.12.2013 unter der Vorgangsnummer 12828009
registriert. Der von Ihnen geschilderte Sachverhalt wird so schnell wie
möglich bearbeitet.

Mit freundlichen Grüßen
Conrad Electronic International GmbH & CoKG

Conrad Electronic UK Ltd. Company Registration Number: 06451808. VAT Number; GB 935 8276 87
Conrad Electronic UK Ltd, PO Box 1318, Barking, Essex, IG11 1ES. Registered Address; 22 Chancery Lane,
London, WC2A 1LS



vince.altavilla@gmail.com

to chongxonexports@hotmail.com

Sun, Dec 1 9:33 PM

## Image Permission

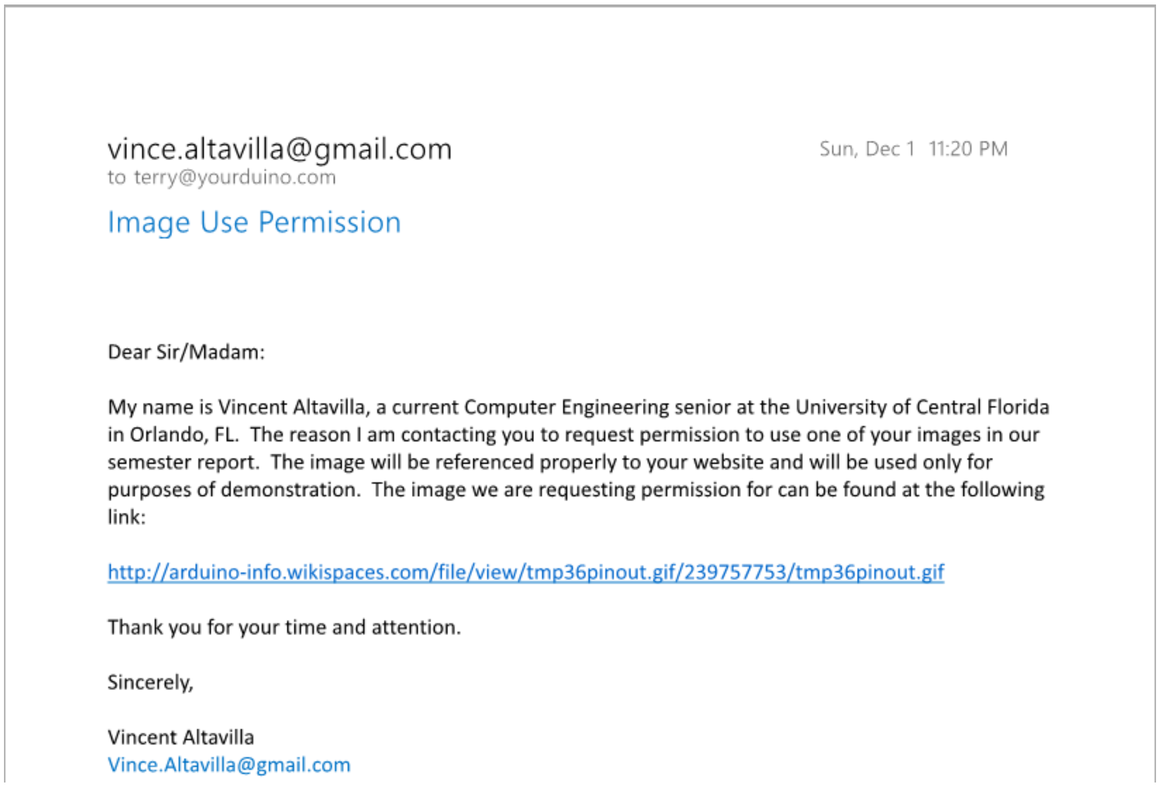
Dear Sir/Madam:

My name is Vincent Altavilla, a current Computer Engineering senior at the University of Central Florida in Orlando, FL. The reason I am contacting you to request permission to use one of your images in our semester report. The image will be referenced properly to your website and will be used only for purposes of demonstration. The image we are requesting permission for can be found at the following link:

http://igbt-china.wikispaces.com/Sell+ATmega48-20AU+ATmega48-20AI+ATmega48+AVR+%E5%8D%95%E7%89%87%E6%9C%BA+ATMEL+Integrated+Circuits+Manufacturer+exporting+direct+from+China

Thank you for your time and attention.
Sincerely,

Vincent Altavilla
Vince.Altavilla@gmail.com



vince.altavilla@gmail.com

to terry@yourduino.com

Sun, Dec 1 11:20 PM

## Image Use Permission

Dear Sir/Madam:

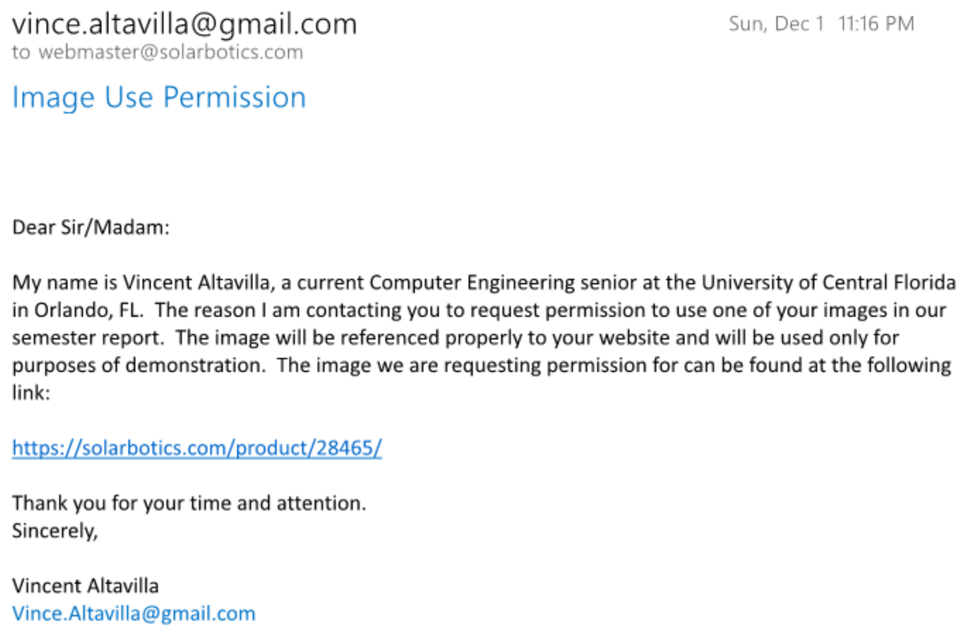
My name is Vincent Altavilla, a current Computer Engineering senior at the University of Central Florida in Orlando, FL. The reason I am contacting you to request permission to use one of your images in our semester report. The image will be referenced properly to your website and will be used only for purposes of demonstration. The image we are requesting permission for can be found at the following link:

http://arduino-info.wikispaces.com/file/view/tmp36pinout.gif/239757753/tmp36pinout.gif

Thank you for your time and attention.

Sincerely,

Vincent Altavilla
Vince.Altavilla@gmail.com



vince.altavilla@gmail.com  Sun, Dec 1 11:16 PM
to webmaster@solarbotics.com

Image Use Permission

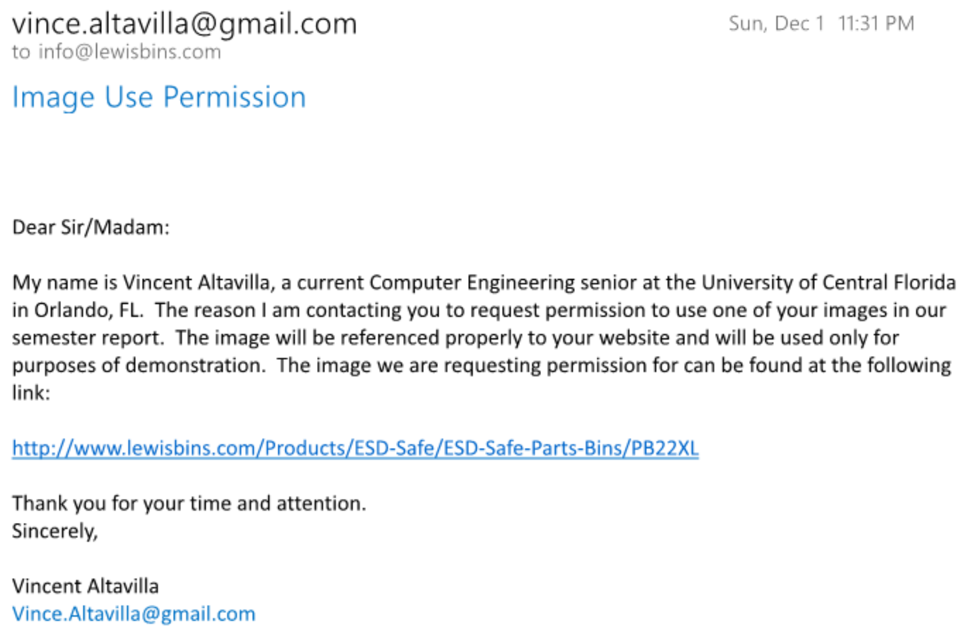
Dear Sir/Madam:

My name is Vincent Altavilla, a current Computer Engineering senior at the University of Central Florida in Orlando, FL. The reason I am contacting you to request permission to use one of your images in our semester report. The image will be referenced properly to your website and will be used only for purposes of demonstration. The image we are requesting permission for can be found at the following link:

https://solarbotics.com/product/28465/

Thank you for your time and attention.
Sincerely,

Vincent Altavilla
Vince.Altavilla@gmail.com



vince.altavilla@gmail.com  Sun, Dec 1 11:31 PM
to info@lewisbins.com

Image Use Permission

Dear Sir/Madam:

My name is Vincent Altavilla, a current Computer Engineering senior at the University of Central Florida in Orlando, FL. The reason I am contacting you to request permission to use one of your images in our semester report. The image will be referenced properly to your website and will be used only for purposes of demonstration. The image we are requesting permission for can be found at the following link:

http://www.lewisbins.com/Products/ESD-Safe/ESD-Safe-Parts-Bins/PB22XL

Thank you for your time and attention.
Sincerely,

Vincent Altavilla
Vince.Altavilla@gmail.com

Fri 11/29/2013 1:12 PM
**To:**

customerservice@sparkfun.com;

To whom it may concern,

My name is Aziz Elouali and i am a senior electrical engineering student at University of Central Florida. I am emailing to request permission to use the picture of ADXL335 Accelerometer for my senior design research project. I appreciate your assistance and consideration in this matter, Thank you.

The associated Links are below:

Figure - ADXL335 Accelerometer
https://www.sparkfun.com/products/9269

## Permission to use some information

Aziz2010@knights.ucf.edu
Fri 11/29/2013 1:02 PM
**To:**

linda.kincaid@analog.com;

To whom it may concern,

My name is Aziz Elouali and i am a senior electrical engineering student at UCF. I am emailing to request permission to use some information such as graphs, schematics and data for my senior design research project. I appreciate your assistance and consideration in this matter, Thank you.

The associated Links are below:

Power Comparison Chart ADXL362

http://www.analog.com/en/mems-sensors/mems-accelerometers/adxl362/products/product.html

ADXL362 Accelerometer Block diagram

http://www.analog.com/en/mems-sensors/mems-accelerometers/adxl362/products/product.html

## Permission to use information

Aziz2010@knights.ucf.edu

Mon 12/2/2013 12:43 AM
**To:**

Support@ti.com;

To whom it may concern,

My name is Aziz Elouali and i am a senior electrical engineering student at University of Central Florida. I am emailing to request permission to use some information specifically the typical application of LM3658 for my senior design research project. I appreciate your assistance and consideration in this matter, Thank you.

The associated Links are below:

LM3658 typical application
http://www.ti.com/lit/ds/snvs328e/snvs328e.pdf

Aziz Elouali
Aziz2010@knights.ucf.edu

## RE: GEN, Email Technical Support, www.ti.com, LM117 1-1226020214

support@ti.com

Fri 11/29/2013 1:09 PM
**To:**

Aziz2010@knights.ucf.edu;

Get more apps

Action Items

*****************************************************************************************************

Thank you for contacting the Texas Instruments Semiconductor Customer Support Center.  This is an automatic acknowledgement of the receipt of your email.  Your inquiry has been assigned Service Request# 1-1226020214 and is currently being worked by our Technical Support Staff.

If additional information becomes available or you have file attachments that you wish to submit, please reply to this email with that information and/or file(s).  Please ensure the thread ID at the end of this email is included so your reply is automatically linked to your original request.

**Please note:** If there is no Service Request # listed above, then we are manually processing your inquiry.

For online assistance, please visit our Technical Support KnowledgeBase homepage at: http://support.ti.com/sc/knowledgebase

Looking for more information on our DSP and Analog products?  We are offering free DSP and Analog Seminars, which deliver more information than ever before to make your job easier. For more information, go to http://focus.ti.com/docs/training/traininghomepage.jhtml

Regards,
Texas Instruments
Semiconductor Technical Support
http://support.ti.com/

[THREAD ID: 1-K9XV9A]
[SR THREAD ID: 1-K9XV92]
[SYS: Acknowledged]

-----Original Message-----
From: Aziz2010@knights.ucf.edu
Sent: 11/29/13 12:08:27 PM
To: support@ti.com
Subject: GEN, Email Technical Support, www.ti.com, LM117

[This Email Sent From: Email Technical Support
http://www.ti.com/general/docs/contact.tsp]

[wfsegen]

[DATE / TIME (UTC): Fri, 29 Nov 2013 18:08:26 GMT]

[CUSTOMER'S REGIONAL LOCAL TIME: 11/29/2013 1:08:26 PM]

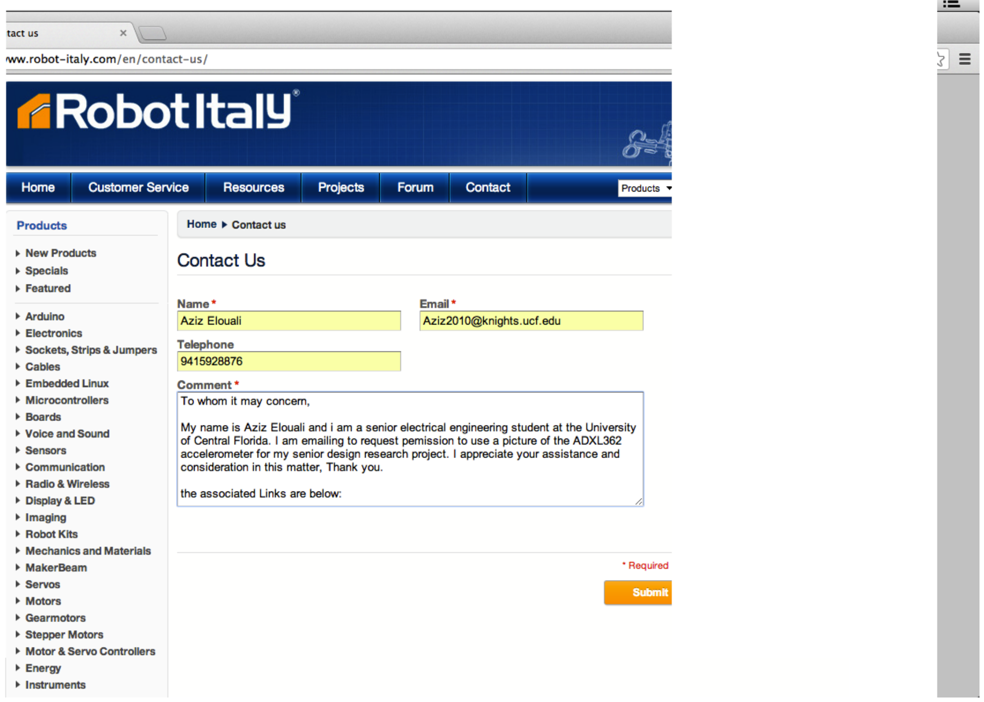
[Name: Aziz Elouali]

[Prefix: Mr.]
[First Name: Aziz ]
[Last Name: Elouali]
[Job Title: ]
[Company: University of Central Florida]
[Email: Aziz2010@knights.ucf.edu]
[Phone: 9415928876]
[FAX: 9415928876]
[Country: USA]
[Address1: 12524 Crest Springs Ln. Unit 1212]
[Address2: ]
[City: Orlando]
[State: FL]
[Postal Code: 32828]
[Part# or Description: LM117]
[Category: Price and Availability]
[Application: Other]
[Design Stage: New design]
[Estimated Annual Production: 0 units]
[Production Date: n/a]

[Problem:
To whom it may concern, My name is Aziz Elouali and i am a senior electrical engineering student at UCF. I am emailing to request permission to use some information such as graphs, schematics and data. for my senior design research project. I appreciate your assistance and consideration in this matter, Thank you. The associated Links are below: LM117 Current Limited Voltage Regulator http://www.ti.com/lit/ds/snvs774n/snvs774n.pdf LM3658 typical application http://www.ti.com/lit/ds/snvs328e/snvs328e.pdf ]

[Steps Needed to Recreate Problem:

]


Name * Aziz Elouali
Email * Aziz2010@knights.ucf.edu
Telephone 9415928876
Comment *
To whom it may concern,

My name is Aziz Elouali and i am a senior electrical engineering student at the University of Central Florida. I am emailing to request permission to use a picture of the ADXL362 accelerometer for my senior design research project. I appreciate your assistance and consideration in this matter, Thank you.

the associated Links are below:

## Appendix B - References

1. Figure 1 – ATMEGA48-AU
   http://igbt-china.wikispaces.com/Sell+ATmega48-20AU+ATmega48-20AI+ATmega48+AVR+%E5%8D%95%E7%89%87%E6%9C%BA+ATMEL+Integrated+Circuits+Manufacturer+exporting+direct+from+China
2. Figure 2 – PIC18F14K22
   https://solarbotics.com/product/28465/
3. Figure 3 – A2235-H
   http://www.bt2000.co.uk/Images/Maestro/A2235-H.PNG
4. Figure 4 – GMS-HPR
   http://www.gtop-tech.com/Templates/pic/Gms-hpr.jpg
5. Figure 5 – Venus638FLPx-L
   http://tekdis.com.au/news/27/Venus638FLPx_L_Venus638FLPx_D_1_nl.jpeg
6. Figure 6 – LM35
   http://www.electroschematics.com/wp-content/uploads/2011/04/lm35.jpg
7. Figure 7 – TMP35
   http://arduino-info.wikispaces.com/file/view/tmp36pinout.gif/239757753/tmp36pinout.gif
8. Figure 8 – TD5A
   http://www.conrad-electronic.co.uk/medias/global/ce/5000_5999/5000/5050/5053/505306_RB_00_FB.EPS_1000.jpg
9. Figure 9 – ESD Bin
   http://www.lewisbins.com/Products/ESD-Safe/ESD-Safe-Parts-Bins/PB22XL

10. Figure 10 - Tilt Sensor
http://www.pyroelectro.com/tutorials/accel_intro/theory.html
11. Figure 11 - ADXL362 Accelerometer
http://www.robot-italy.com/en/triple-axis-accelerometer-breakout-adxl362.html
12. Figure 12 – Power Consumption Chart
http://www.analog.com/en/mems-sensors/mems-accelerometers/adxl362/products/product.html
13. Figure 13 – ADXL335 Accelerometer
https://www.sparkfun.com/products/9269
14. Figure 14 - ADXL362 Accelerometer Block diagram
http://www.analog.com/en/mems-sensors/mems-accelerometers/adxl362/products/product.html
15. Figure 15 – Bicycle Dynamo Mechanism
http://www.gcsescience.com/pme21.htm
16. Table 15 – Features of Joule 3 Dynamo
http://www.thinkbiologic.com/sites/default/files/pr/2012/08/120829-BioLogic-Joule3-HG-v4_0.pdf
17. Table 16 - Shimano Alfine DH-3N80 Features
http://sheldonbrown.com/harris/lighting/shimano.html
18. Figure 17 – Full bridge rectifier
http://en.wikipedia.org/wiki/Diode_bridge
19. Table 18 – Features of LM117
http://www.ti.com/lit/ds/snvs774n/snvs774n.pdf
20. Table 19 - Standard 4 Layer PCB Construction (0.062")
http://www.sunstone.com/products-services/quickturn-proto-boards/construction-detail.aspx
21. Figure 26 - 4-layer PCB
http://www.4pcb.com/pcb-stack-ups-0.062.html
22. Table 22 – Characteristics of Batteries
http://batteryuniversity.com/learn/article/charging_lithium_ion_batteries
http://www.premiumbikegear.com/biologic/convenience/reecharge-power-pack.html
23. Figure 31 – LM3658 typical application
http://www.ti.com/lit/ds/snvs328e/snvs328e.pdf
24. Figure 35 – Quick Mount
http://www.bikerumor.com/2012/11/10/sram-releases-quickview-bicycle-computer-mount/

## Appendix C – Abbreviations

1. MCU (Microcontroller Unit)
2. GPIO (General Purpose Input/Output)
3. ADC (Analog to Digital Conversion)
4. DAC (Digital to Analog Conversion)
5. PWM (Pulse Width Modulation)
6. GPS (Global Positioning System)
7. GNSS (Global Navigation Satellite System)
8. DoD (US Department of Defense)
9. TTFF (Time-to-First-Fix)
10. UART (Universal Asynchronous Receiver/Transmitter)
11. OS (Operating System)
12. SPP (Serial Port Profile)
13. EDR (Enhanced Data Rate)
14. BLE (Bluetooth Low Energy)
15. BT (Bluetooth)
16. FCC (Federal Communications Committee, USA)
17. IC (Industry Canada)
18. EC (European Commission, R&TTE)
19. RF (Radio Frequency)
20. A2DP (Advanced Audio Distribution Profile)
21. AVRCP (Audio/Video Remote Control Profile)
22. HCI (Host Controller Interface)
23. PCM (Pulse-Code Modulation)
24. LCD (Liquid Crystal Display)
25. USB (Universal Serial Bus)
26. LNA (Low Noise Amplifier)
27. SAW (Surface Acoustic Wave)
28. LDO (Low-Dropout)
29. 1PPS (A Pulse Per Second)
30. TTL (Transistor-Transistor Logic)
31. NMEA (National Marine Electronics Association)
32. RMC (Recommended Minimum Navigation Information)
33. UTC (Coordinated Universal Time)
34. GGA (Global Positioning System Fixed Data)
35. I2S (Integrated Interchip Sound)