Luggage Link

Project Documentation

Senior Design II

Group 15
David Farrell
Evan Husk
Jose Mousadi

December 3rd, 2012

Table of Contents

Tables

Figures

1 Executive Summary

This project is a Global Positioning Satellite (GPS) based luggage tracking system designed to provide the capability to track one or more pieces of luggage anywhere in the world. The core components of this project consist of Luggage Tracking Units (LTUs) which can be placed inside a suitcase, backpack, briefcase, or any other piece of luggage in order to track that item anywhere in the world. Each individual piece of luggage requires a corresponding Luggage Tracking Unit designated to that specific luggage item. In order to demonstrate the ability to track multiple pieces of luggage, it was necessary to assemble more than one LTU. These tracking units receive GPS signals and transmit their current location to the user via cellular network. Users receive notifications from their LTU(s) in the form of text messages and/or emails. These notifications contain GPS coordinates, error reports, and other pertinent data.

The main objective of this project was to demonstrate tracking capability for a single Luggage Tracking Unit. The most basic way to accomplish this was to send the unit's GPS coordinates to a user's phone via text message. Once this capability was achieved, more advanced features and communication methods were attempted to be integrated into the design. It should be noted that the purpose of these additional features was to expand upon the basic functionality of the system. That is, the basic functionality of the system was not significantly changed, but rather the additional features served to augment the system's ease of use. The majority of these additional features were implemented in software, while the final hardware product remained relatively unchanged from the initial prototype.

The technical approach applied to this project was one in which components and features were implemented with two factors in mind: necessity and risk. In the initial prototype, only the essential components needed for basic functionality were designed and implemented. Once this prototype was developed and basic tracking functionality had been achieved, additional features and components were incorporated. These features were not essential to the basic functionality of the product, but served to improve its overall functionality and design in various ways. By executing this particular approach, our goal was to eliminate any unnecessary risks that would jeopardize completion of the project, and at the same time, to mitigate the risk of any components which were essential to the project.

2 Project Description

2.1 Statement of Motivation

According to the March 2012 Air Travel Consumer Report published by the U.S. Department of Transportation, in January of 2012 there were 139,118 reports filed for mishandled luggage in the United States. While this statistic has decreased slightly since January of 2011, with a total of 181,608 reports filed, the problem of lost, delayed, or otherwise mishandled luggage is a significant one.

It should be noted that "mishandled luggage" includes not just reports of lost luggage, but also reports of damaged luggage, missing items from luggage, and the other similar incidents. However, it should also be noted that this statistic only represents claims of mishandled luggage for domestic flights within the United States. Thus, these numbers represent only a fraction of the total number of incidents occurring each month on international flights from the U.S., as well as the countless other flights happening worldwide every day.

The figure below provides a more comprehensive perspective on the issue of mishandled baggage with a comparison of annual statistics from 1990 to 2010. As mentioned previously, even though the numbers do seem to be on the decline, they are still exorbitant and unacceptable given the resources and technology available today to correct such problems.

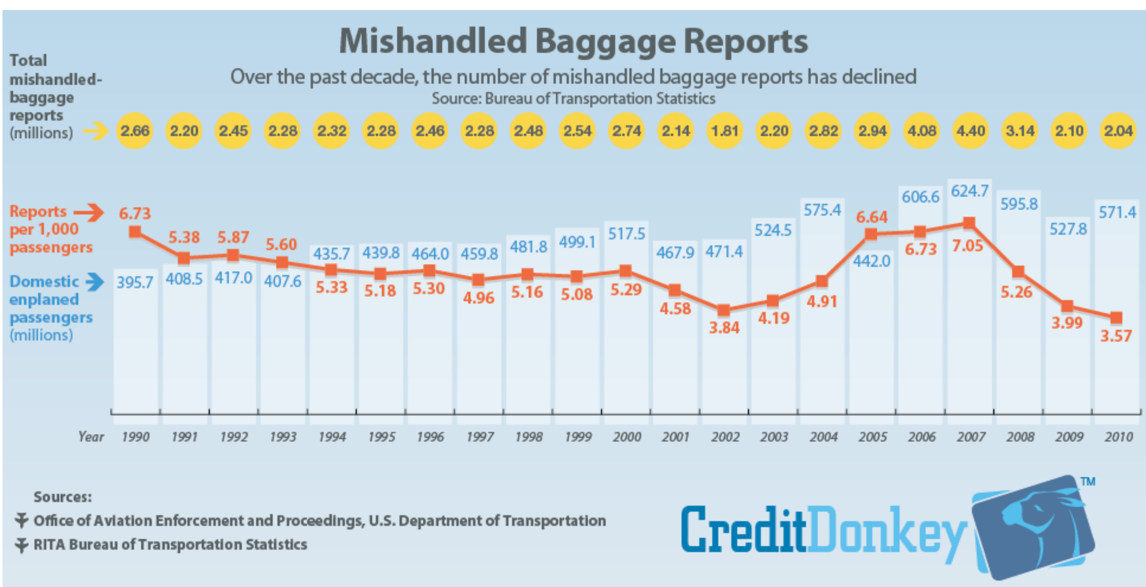

Figure 2-1. Mishandled Baggage Reports
Used with permission from CreditDonkey.com

The inability to fully rely on companies to process and handle luggage has become a significant problem, as well as a major inconvenience, for travelers.

The motivation for this project is to help solve the problem of lost luggage and set those who use the system at ease by providing a simple, reliable way for travelers to locate and track their luggage at anytime, anywhere in the world.

2.2 Goals and Objectives

2.2.1 Top Level Goals and Objectives

The first goal for the Luggage Tracking Unit was to be able to track a piece of luggage reliably anywhere in the world, at any time. At certain time intervals, status updates and notifications were to be sent via text message and/or email. These notifications contain the information necessary to properly identify the location and current status of the item. This information was to include the current coordinates of the item, its velocity, and any other relevant information, so the user could easily identify the location of their luggage.

With the help of cellular networks, a reliable connection was to be established and maintained so that the correct coordinates would be received in a timely manner. That is, the user would not be receiving invalid coordinates and information based on the item's previous locations. Also, the coordinates were to provide a sufficient degree of accuracy so the article's position could be identified even when inside of the airport.

Another important feature that was absolutely necessary to provide a reliable system was a robust power supply that could support the basic functionality of the LTU for an extensive period of time. The LTU needed to be able to maintain its functionality for a number of days in order to allow any lost or misplaced luggage to be located and retrieved.

2.2.2 Hardware Goals and Objectives

From a hardware perspective, a highly portable, affordable, and reliable LTU needed to be designed that could provide an accurate position and could send Short Message Service (SMS) text messages, as well as emails, to the user. In order to properly design an efficient and accurate unit, many hardware sub-goals were set and achieved.

Because the small size of the unit was a vital concern, an important hardware design sub-goal was to ensure that all components were placed in the housing in a compact, orderly manner in order to minimize the overall dimensions of the unit. Another sub-goal was finding and integrating a reliable, yet, somewhat inexpensive GPS module which could be easily integrated into the relatively small, portable encasement. In order to successfully send SMS text messages and emails, a Global System for Mobile Communications (GSM) module was incorporated in the hardware design. Again, these units were to be small and affordable so they could be easily integrated into the compact design.

Another important hardware sub-goal was to find and incorporate a power source that could support the multiple modules in the system. It was imperative that the battery be reliable and able to work for extended periods of time in between charges. With the accomplishment of each hardware sub-goal, the main hardware objectives were inevitably realized.

2.2.3 Software Goals and Objectives

In order to develop a system which supports the functionality discussed in the previous sections, software components were also integrated. The most important piece of software was the on-chip software, or firmware. The on-chip software allows all of the other components and modules on the LTU to exchange information in a reliable and efficient manner. It also needed to direct the functionality of each of the components, such as commanding the GSM module to send SMS text messages and emails. A logging mechanism was also set in place so that times and coordinates could be stored for later retrieval if the cellular connection had been lost and data could not be transmitted.

A peripheral software component that was initially planned to be implemented was a GPS server. However, only the initial setup and configuration of the server was achieved due to time constraints.

In addition, at the beginning of the project, the final software goal was to implement an iOS application. However, due to time constraints, development of the iOS application had to be canceled.

2.3 Project Function

The luggage tracking system allows users to carry a small, lightweight LTU in their luggage and easily track that luggage with the help of multiple, user-friendly wireless communication interfaces. The LTU utilizes GPS technology to provide real-time locating and tracking of the user's item(s). The design of the actual LTU is a highly portable, lightweight device that can be easily placed within any item the user would like to track. It utilizes available GSM networks to send notifications at regular time intervals to keep the user updated on the status and location of their luggage.

Peripheral hardware features were added to provide additional information to the user on the status and location of their luggage. A barometric pressure sensor was integrated onto the LTU in order to monitor the air pressure around the LTU. If a significant drop in pressure is detected during a certain period of time, it is assumed that the LTU is onboard an aircraft. At that point, a notification is sent to the user informing them of this event, and GSM cellular communication is disabled in adherence to Federal Aviation Administration regulations. When a significant increase in pressure is detected, the aircraft carrying the LTU is assumed to have landed, and GSM cellular communication is enabled.

A phototransistor light sensor was also integrated onto the LTU in order to alert the user if their luggage is opened. If the luggage is opened and the phototransistor is exposed to light, a notification is immediately sent to the user alerting them that their luggage was opened.

3 Specifications and Requirements

3.1 Hardware Requirements

3.1.1 PCB

All major internal components shall be mounted to a printed circuit board (PCB). The PCB shall provide all data storage and processing components for the LTU. The physical PCB shall be designed and manufactured to fit within the maximum size requirement of the enclosure listed in section 3.1.10. The PCB shall also provide capability to handle all power regulation and distribution from the LTU's power source. If it is decided that a port will be used to manually reprogram the LTU's settings, the PCB shall provide such a port in an easily accessible location.

3.1.2 Microcontroller

A programmable microcontroller is required for this project. It should have at minimum of 4 digital outputs, 3 digital inputs and 1 Analog input along with the ability to convert analog voltage to digital. Internal memory should be large enough to withhold the LTU program. The microcontroller should have USCI and SPI capabilities in order to communicate with the cellular module and the flash memory respectively. Not knowing how extensive the program would be, a flash memory of 16KB and SRAM of 512B will be sufficient. Since the LTU is a portable device, power is a concern, so the microcontroller should draw the lowest current possible.

3.1.3 GPS Module

The LTU contains a GPS module to provide real-time position tracking. The GPS provides position information accurate to within +/- 15 meters. The GPS provides velocity information accurate to within +/- 0.2 meters per second.

3.1.4 GSM Module

The LTU contains a GSM module. Based on settings determined by the user, the GSM module transmits SMS text message and email alerts with GPS coordinates and other pertinent information. The GSM module contains an externally accessible slot for a subscriber identity module (SIM) card.

3.1.5 Antennas

The LTU contains separate antennas for the GPS and GSM components. Both antennas provide enough receptivity to operate inside of a multi-storied building.

This is for the purpose of ensuring that the LTU is able to receive GPS satellite and GSM network signals when inside an airport terminal.

3.1.6 Memory

The LTU contains a rewriteable form of memory such as Flash, RAM, or EEPROM. This memory component is mounted on the PCB and interfaced with the GPS Module, GSM Module, and other data processing components as necessary. The memory is used to store the GPS data log, alert settings, and all other relevant data.

3.1.7 Indicators

The LTU contains three externally-viewable Light Emitting Diodes (LEDs), one to indicate power, one to indicate GPS status, and one to indicate GSM status. The power indicator LED is turned on when the LTU is powered on and remains on until the LTU is powered off. The GPS status LED turns on and flash at intervals based on its current mode of operation. The GSM status LED turns on and flashes at intervals when searching for a GSM network or when trying to register on a GSM network. The LED shall blink at 3-second intervals when the LTU has established a connection with a GSM network. The GSM status LED remains in this state until the connection is broken or until the LTU is powered off, at which point the LED is turned off.

3.1.8 Power

The LTU contains a rechargeable battery capable of powering the system for a minimum of 72 hours of normal operation without recharging. A power adapter and charge controller is implemented in order to recharge the battery.

3.1.9 External Controls

The LTU includes a normally-open momentary push button which turns the device on and off. The switch should be resistant to pressure from incidental contact, but easy for the user to depress intentionally. A normally-open momentary push button is also included as a manual reset button.

3.1.10 Enclosure

A two-piece enclosure is utilized to house the LTU. The enclosure contains apertures for the Mini Universal Serial Bus (USB) port, power switch, and LED indicators. The maximum dimensions of the enclosure are 4” × 4” × 0.5”. The enclosure has a durable, robust design, yet be easy to open for access to internal components. If possible, the enclosure should be waterproof or at least water resistant in order to protect the internal components.

3.1.11 Power Port

The LTU has an external power port into which the power adapter may be plugged in order to recharge the battery. The power port was designed in order to provide minimal possibility of damage or entrance of debris or other foreign objects.

3.2 Software Requirements

3.2.1 GPS Server

A GPS server was implemented for the purpose of functioning as the central data collection and distribution point between the LTU and the user. It was to be configurable with multiple LTUs and to provide the capability of receiving and processing data from those LTUs. The GPS server was also intended to distribute data to the web portal and iOS application for retrieval and analysis by the user. Due to time constraints the GPS server was only partially implemented and full functionality was not achieved.

3.2.1.1 Alert Control

Alert control software was developed to determine which alerts are sent to the user and when they are sent. This software monitors the incoming GPS data and compares the LTU’s location with its intended destination. If conditions are met that qualify an alert to be sent, the appropriate data is gathered and sent to the GSM module for transmission.

3.2.2 Firmware

The LTU firmware was implemented using the C programming language. It is capable of correctly interpreting and processing information concerning all data from the GPS/GSM module, as well as all other components on the LTU. The basic processes of the LTU’s hardware are also handled by the firmware, certifying correct communication between modules.

3.2.2.1 Interface Control

Interface control software was implemented in order to handle the interaction of the various components on the PCB. This software included any necessary features such as clock generators, watchdog timers, and commands for sending and receiving data to and from various hardware components.

3.2.2.2 Data Logger

A data logger was implemented using an Atmel flash memory chip in order to store LTU events and other data. Because a cellular connection cannot be guaranteed at any given time or location, it was necessary to store the events occurring within the LTU for later transmittal to the GPS server and/or directly to the user. Some of these potential events are successful GPS fix acquisitions, light sensor events, and pressure sensor events, as well as any errors related to those events.

3.2.2.3 Power Control

Power controls were incorporated into the software in order to regulate the power modes of the internal components of the LTU. These controls were implemented in such a way as to enable full power mode for the various components only when necessary, and to command them into a low power mode when not in use.

3.3 Functional Requirements

3.3.1 Multi-device Tracking

The luggage tracking system provides the capability to track two or more LTUs simultaneously. This was accomplished simply by assembling two LTUs, each with an independent set of components that function without relation to the other LTU. This enables the user to receive notifications from both LTUs simultaneously, if desired.

3.3.2 User Alerts

The luggage tracking system provides alerts in the form of SMS text messages and emails for specific events such as errors or light sensor events. Alert settings are predefined and not configurable by the user.

3.3.3 User Control

A rudimentary capability for the user to communicate with the LTU via SMS text message was implemented and demonstrated. This functionality currently allows the user to power down the LTU remotely by sending a text message containing a specific command to the LTU.

4 Research

4.1 Existing Similar Projects

While researching this project, several products were found that were similar to our concept. One that stood out the most was a product called PocketFinder®. This product is marketed as a “personal GPS locator” useful for tracking people, pets and vehicles. In addition to the obvious similarity of being GPS-based, this product is similar to our design in that it also features GSM and GPS modules, and provides the same basic capabilities we implemented in our design.

While the PocketFinder is an excellent product, we believe our product is unique because it is designed specifically for tracking luggage. In all of our research, no product was found that was designed for this specific use. As technology in the GPS market improves and costs are driven down, we believe our product has the potential to be marketed as an affordable option to be purchased and placed into luggage. However, because the cost of GPS technology is still relatively expensive, the long-term plan for our product is to integrate it into a line of luggage. The PocketFinder tracking unit is shown in Figure 4-1 below:

Figure 4-1. PocketFinder® GPS Tracking Unit

Used with permission from Location Based Technologies, Inc.

Luggage is already a very expensive commodity, so adding roughly 100 dollars to the cost would not be as significant of an expense to prospective consumers as would a small electronic device that must be purchased separately. Luggage manufactures would, of course, advertise the fact that their luggage lines feature built-in GPS tracking units to help provide a worry-free travel experience.

Another similar project discovered while researching is the UberTracker. The UberTracker is a GPS tracker produced by Sparkfun Electronics™ and is marketed as a potential project component for hobbyists and electronics enthusiasts. The UberTracker is much more "open source" than the PocketFinder product in the fact that the UberTracker has significantly greater amount of information available regarding its internal components and features.

Figure 4-2. UberTracker GPS Tracking Unit
Used with permission pending from Sparkfun Electronics™

We initially planned to use the UberTracker as a guideline throughout the development of our project as it is very similar in both design and implementation to what we hoped to accomplish. However, in actuality we utilized this particular resource only during the beginning weeks of the project as we determined the hardware components and architecture of our design. Once those features were decided upon, the majority of the effort in the project was software-related and the UberTracker example was consequently no longer useful.

4.2 Relevant Technologies

4.2.1 GPS

The technology most relevant to our project is most certainly GPS technology. Because GPS technology is satellite-based, it provides the most widely-available and cost-effective solution for wirelessly tracking a device anywhere in the world.

4.2.2 GPS Server

GPS servers are a relatively recent technology, but thankfully they have progressed swiftly to a level of reliability acceptable for this project. In addition, there are several open-source GPS servers available for free download and use, which was of great benefit to us both technically and financially. The main purpose of a GPS server is to receive incoming messages from a GPS tracking unit and distribute the relevant information to the user in various forms. Data is typically sent to the GPS server via GSM or General Packet Radio Service (GPRS) communication. Upon receipt by the server, the data is processed and often stored in a data logger for later retrieval. The data is then sent to devices such as a user's cell phone or email account. The data is also frequently used to populate a map so the user may view the current locations of his or her tracking units as well as their previous paths of travel.

4.2.3 GSM

GSM technology was another crucial part of this project. GSM was chosen as the preferred method of wireless communication because of its global coverage. The next best alternative to GSM for wireless data transmission would have been CDMA, which is currently available only in North America and parts of Asia. Because luggage might be sent anywhere in the world, either intentionally or unintentionally, it was imperative to select a technology with proven, reliable global communication capabilities.

4.3 Strategic Components

4.3.1 GPS Module

The GPS module is a strategic component in our design for two main reasons: connectivity and power. Although these two features are proportional in amount, unfortunately the affects they have on the system are inversely proportional. In other words, GPS modules vary in receptivity. The better the reception capabilities, the better connectivity the device will possess. The negative aspect is that the better the receptivity of the GPS module, the more power it consumes. Therefore, a balance must be achieved between the two that provides a high level of reception while consuming as little power as possible.

4.3.2 GSM Module

The GSM module was a strategic component in our design for reasons similar to that of the GPS module. The tradeoff between reception capability and power consumption was a delicate one that had to be balanced and optimized. For this project, we chose a GSM module that was both cost effective and had a reasonable reception capability. Another significant factor in the decision process was the option of choosing a module that contained both a GPS and a GSM module. This proved to be an excellent option for the purposes for this project, as it significantly mitigated risk by eliminating the necessity to accommodate for communication between two different components as opposed to just one component.

4.3.3 Firmware

Because our project was developed in phases, the software components we designed played a very strategic role. If not designed properly, and in an object-oriented fashion, the software would need to be redesigned and rewritten in order to allow for additional features to be implemented.

This was especially true for the firmware in this project, as it needed to be designed with future capabilities in mind. Initially, the data was to be received from the GPS module, processed into a text message, and then sent to the GSM module for transmission. However, in the later phases of development, a GPS server was to be implemented. In order to interface properly with the GPS server, data would need to be processed and distributed in a different manner. Instead of being compiled into a simple text message, outgoing data would need to be generated and transmitted in a format compatible with the GPS server.

With this in mind, the firmware was designed to be as object-oriented as possible. This allowed for relatively easy changes to be made without affecting the software as a whole. The principle of object-oriented design is one that was applied to the entire project, but especially to the software components developed by the team.

4.3.4 GPS Server

As mentioned previously, the plan for this project was to implement a GPS server. The GPS server was a strategic component primarily because it would allow for multiple LTUs to be tracked and viewed simultaneously. Without a GPS server, the extent of the user's capability to manage his or her devices would have been limited to receiving and viewing individual text messages and emails from each individual LTU. With a GPS server, the user would have the ability to view the status of any or all LTUs from one source, greatly reducing the interaction and work required on the user's end. A GPS server would also provide the ability to store and process data at a central location, as all LTUs belonging to the user would send their data to the same server. As indicated in prior sections, the full functionality of this GPS server was not able to be obtained due to time constraints.

5 Design Details

5.1 Design and Development Strategy

In order to ensure a functional product was developed, our strategy was to design and develop prototypes in phases. These phases started with a design that met the basic requirements of the project and then increased in complexity and robustness with each subsequent phase. As the phases of development progressed, the initial prototype was expanded upon and a second prototype was assembled in order to incorporate additional features and to provide a second source for testing. The purpose of this strategy was to ensure that a working prototype would be available well in advance of the final demonstration and presentation at the end of the Senior Design course.

The initial plan put in place consisted of three phases of development: A, B, and C. These phases were as described in the table below:

Table 5-1. Design and Development Phases

Phase	Capabilities and Features
A	Ability to receive GPS signal and send GPS coordinates via SMS text message. In this phase we are simply making something that works and can be demonstrated, nothing more.
B	All capabilities of Phase A plus a GPS server interface which receives data from one or more LTUs and displays a map with the location of the user's LTUs. Explore transmission protocol options such as GSM vs. GPRS. Explore web interface options and features such as geozones, customized alerts, etc.
C	All capabilities of Phase B plus an iOS app interfaced with the GPS server that provides a map of LTU locations updated in real time. Also look into the option of integrating the LTU with actual luggage rather than having it as a standalone unit.

With respect to hardware prototypes, our strategy was to leave the Phase A prototype untouched once it was completed. In doing this, we eliminated the risk of damaging or causing errors within the unit. The Phase A prototype was to be the last-resort plan if implementation of the subsequent phases was unsuccessful. Were an acceptable level of confidence in the implementation of the GPS server component be achieved in Phase B, steps could be taken to alter the Phase A prototype to also be compatible with the GPS server. However, due to limited availability of time, Phase A became the sole implementation for this project, while Phase B was partially implemented and Phase C was aborted.

5.2 Risk Assessment

The overall scope of this project contained several features which presented significant risk factors in the design and development process. The main cause for these items being a high risk was due to a lack of experience in those areas. Although our team had a wide variety of background and experience, several areas remained in which our team had little or no experience. The amount of risk involved with these items contributed significantly to the decision to divide the development process into phases. Only the most critical functions were included in the initial phase, while each subsequent phase added more features that increased in risk but decreased in importance.

5.2.1 Phase A Risks

Because Phase A was the most crucial phase of our development process, a great amount of attention was focused on any issues that presented a risk to the successful completion of this phase. One area that presented a significant concern was the software control of the GSM/GPS module. The GSM and GPS components within this module needed to be able to communicate in order to achieve the basic function of transmitting GPS coordinates and other information to the user. Data gathered by the GPS module needed to be processed into an SMS text message format and sent to the GSM module for transmission. This was the issue that was addressed from the beginning and whose functionality was established as early as possible in order to mitigate risk associated with the core functionality of the project.

5.2.2 Phase B Risks

The only high-risk item in the Phase B development period was the GPS server. Initially, none of the members of our team had any relevant experience with server development, setup, or administration. For this reason, the GPS server used for this project was a free, open-source software package that was largely pre-configured.

However, even with the benefit of utilizing a pre-configured software package, there was a significant amount of risk involved with simply integrating the server into our system and interfacing it with LTUs. This task quickly proved to be a very time-consuming undertaking, and it was decided as a team that the amount of time left to implement the server was not sufficient. Once that decision was made, all further effort toward Phase B was halted, and the remaining amount of time was used to enhance and refine the features implemented in Phase A.

5.2.3 Phase C Risks

Another at-risk item, and the one that presented the greatest risk of failure to achieve implementation, was the iOS application. As was the case with the GPS server, going into this project, our team had no experience in the realm of mobile application development. In order to develop an iOS application, the iOS programming language and developer environment would have had to be learned before development could begin, and the amount of time in which this needed to be accomplished was very short. As development of the project progressed, it soon became apparent that this phase would need to be dropped, which further contributed to the fact that having a scalable project was a beneficial approach. By the mid-semester demonstration, it was determined that Phase C would be canceled in order to ensure the implementation of Phase A and possibly Phase B. In hindsight, this was unquestionably the correct decision to make.

5.3 Functional Architecture

5.3.1 GSM Module

When the LTU is powered on, the GM862 GPS/GSM is immediately powered on by the microcontroller. Once the GM862 is powered on, the GSM and GPS modules both enter full power mode by default. A simple command is then sent to the GM862 to verify communication between the MSP430 microcontroller and GM862 module. If this communication is not verified, the module does not attempt to perform further actions that require the use of the GPS/GSM module. Upon verification of successful communication, the GSM and GPS modules are immediately commanded into a low power mode and an operation cycle is initiated. “Operation cycle” is the name given to the series of tasks that are executed by the LTU every certain number of minutes. The first task executed in the operation cycle is a communication test. This is the same test that is run when the LTU is first powered on.

After the communication test, the GPS module makes a maximum of three attempts to obtain a GPS fix. If a GPS fix is successfully obtained, the data is saved in the LTU’s memory and transmitted to the user according to predetermined notification settings. If the GSM module is not able to send the data, error reports are be stored in memory and the GSM module reenters low power mode. During the next operation cycle, the LTU attempts to send any messages that were unsuccessfully sent in the previous operation cycle as well as any new messages generated in the current operation cycle.

Air pressure is another condition that is checked before allowing the GSM module to transmit messages. Because the LTU was designed for tracking luggage, it may often be the case that the luggage being tracked is on an aircraft. Since most luggage is carried in unpressurized storage compartments underneath the aircraft, a barometric pressure sensor may be used to take readings of the surrounding air pressure. The Federal Aviation Administration has strict rules prohibiting the operation of cellular devices while onboard an aircraft, so in order to maintain compliance with these regulations, the GSM module must not be allowed to power on or remain on when it is determined that the LTU is on an aircraft. One of the simplest ways to determine whether or not the LTU is on an aircraft is to monitor the surrounding air pressure. If a significant drop in pressure is detected during a certain period of time, it is assumed that the LTU is onboard an aircraft. At that point, a notification is sent to the user informing them of this event, and GSM cellular communication is disabled in adherence to Federal Aviation Administration regulations. When a significant increase in pressure is detected, the aircraft carrying the LTU is assumed to have landed, and GSM cellular communication is enabled.

If all conditions for transmission are met, the module attempts to transmit any messages in the transmit buffer at that time. Once the messages have been sent, the module returns to a low power mode. A maximum of 3 attempts are allowed for any event notification to be sent to the user. If a GSM network has not been found, or if a network has been found but the signal strength is too weak to transmit messages, the initial attempt(s) to send a notification may fail. If, after three attempts, the notification has not been sent successfully, the memory location of that notification is stored in the transmit buffer so the message can be transmitted at a later time. An error report is also generated and stored in memory, and finally the module returns to a low power mode.

Once the module enters the low power mode, the operation cycle timer begins. When the timer reaches its stopping point, the GSM module reenters full power mode and begins the message transmission process again.

The activities performed by the GSM module are summarized in the activity diagram below:

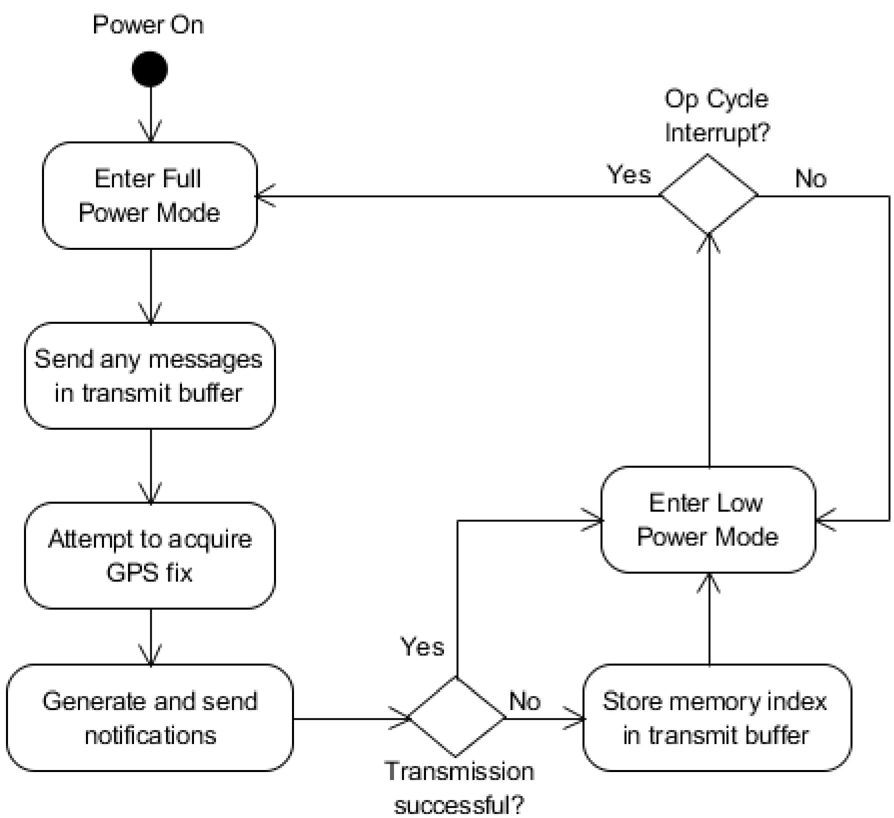

Figure 5-1. GSM Module Activity Diagram

If, at any point during these operations, an error occurs, the error is logged in memory and an error notification is sent to the user, provided the conditions for transmission are met.

5.3.2 GPS Receiver

The activity cycle of the GPS receiver is very similar to that of the GSM module. The GPS receiver is powered on when the LTU is powered on and is initially in full power mode. Once communication with the GM862 has been verified, the

GPS module is commanded into low power mode. Unlike the GSM module, the GPS receiver has no restrictions for use onboard a commercial airplane. For this reason, it is not necessary to check any conditions before the module is allowed to search for available GPS signals. If a sufficient number of GPS satellites are in range, enabling a successful GPS fix, the data obtained is processed into individual buffers, stored in memory, and compiled into a message to be transmitted by the GSM module. Once the data has been stored in memory, the GPS receiver will reenter lower power mode in order to minimize power consumption. In the case that the receiver is not able to immediately acquire a GPS fix, it is allowed two more attempts to do so. If three attempts are made and a successful GPS fix has not been obtained, the receiver reenters a low power mode.

After the GPS module enters the low power mode and all other necessary operations are completed, the operation cycle timer is started. This is the same timer that controls the GSM module operation. When the timer reaches its stopping point, the GPS receiver will reenter full power mode and begin the GPS satellite acquisition process and data storage process again. The activities performed by the GSM module are summarized in the activity diagram below:

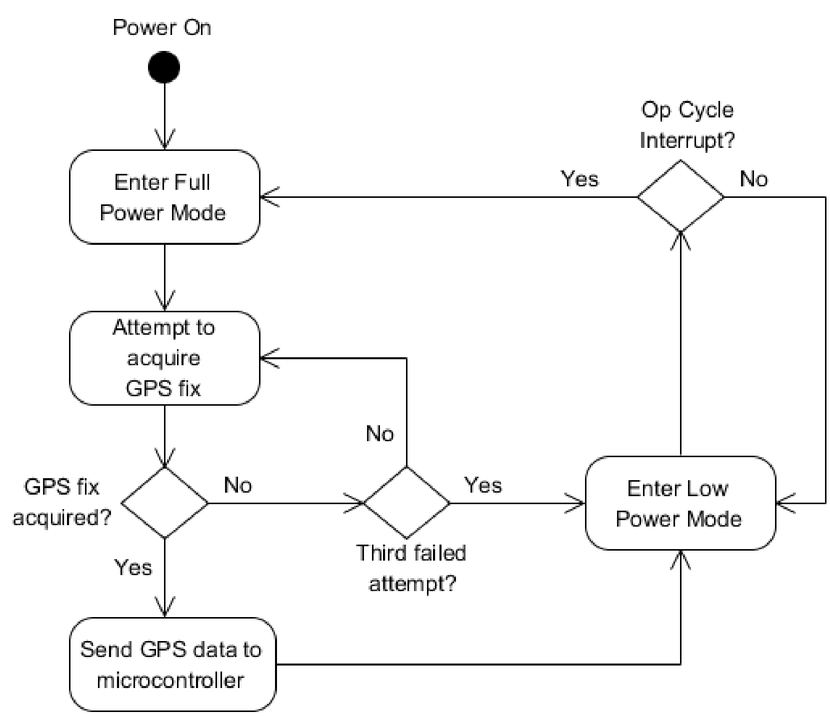

Figure 5-2. GPS Receiver Activity Diagram

5.4 Hardware Architecture Diagrams

Figure 5-3 depicts an overview of interconnections for the hardware components. The microcontroller serves as the central controller for handling all communications for the LTU. Moreover, the microcontroller monitors the subsystems in order to coordinate sequences of operation and minimize power consumption. The microcontroller interfaces with the GM862 via Universal Serial Communication Interface (USCI). The microcontroller communicates with the flash memory using Serial Peripheral Interface (SPI). For Network status and GPS status, two digital outputs provided by the GM862 are utilized. These digital outputs control two LEDs: a green LED for GSM network status and a yellow LED for GPS status. A red LED is connected to an output pin on the microcontroller to indicate different states of the LTU. The microcontroller also monitors the barometric pressure sensor through an input pin that reads analog voltage levels. A different pin, set for digital input, is connected to the phototransistor. The GSM and GPS antennas are connected to the SMA receptacles on the GM862. The connections are made via coax cables and MCCX male connectors. The two-position switch turns the LTU on and off and controls the power distribution throughout the PCB. The charge manager and the USB Micro B receptacle utilize the 5 VDC from the power adapter to charge the Li-Ion battery properly.In order to turn the GM862 on and off, an output pin on the microcontroller is connected to the on/off pin of the GM862 and provides the required two-second digital low signal.

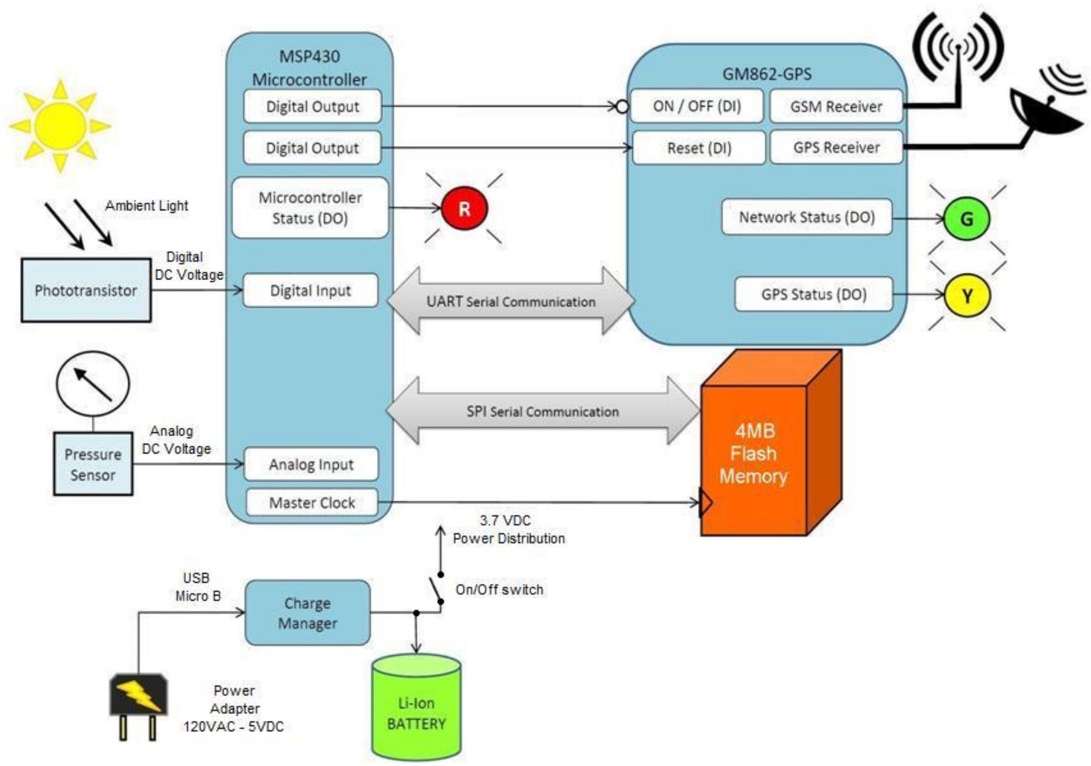

Figure 5-3. Hardware Architecture Diagram

5.5 Hardware Components

5.5.1 Microcontroller

Device: MSP430 ultra-low-power 16-bit microcontroller.
Part Number: MSP430G5223
Manufacturer: Texas Instruments.
Description: 16-bit RISC CPU, 16-bit registers, 16-bit timers, 10-bit analog-to-digital (A/D) converter, 16 IO pins, support USCI, UART and SPI.
Power: 1.8 V to 3.6 V.

Figure 5-4 shows the MSP430 microcontroller in a package type 20-TSSOP, which allows for removal of the 20-pin IC socket for programing with the MSP430 Launch Pad. It has 16 user selectable I/O pins, a 10-bit analog-to-digital converter (ADC10), up to 16MHz clock, Low-Frequency (LF) Oscillator, two 16-bit timers with three capture/compare registers that are used for interrupts, ultra-low power consumption and power saving modes. The MSP430 also offers a Universal Serial Communication Interface (USCI) with UART supporting automatic baud rate detection, synchronous SPI and I2C communication protocol interfaces.

Figure 5-4. Texas Instruments MSP430G2553 16-bit Microcontroller
Used with permission pending from Texas Instruments.

5.5.2 GPS and GPRS/GSM Module

Device: GPS receiver and GPRS digital communication GM862-GPS
Part Number: GM862-GPS
Manufacturer: Telit
Description: Quad-band GSM/GPRS modem with 20-channel high sensitivity silicon radio frequency (SiRF) Star III GPS receiver

Figure 5-6 shows the general dimensions for the GM862 module. In order to interface with this module, a male connector SMD 50-pin is used on the PCB.

The GM862 also has a Mini-SIM card holder or receptacle, an SMA female connector for GSM/GPRS antenna and an SMA female connector for a GPS antenna.

Figure 5-5. Telit GM862-GPS Module

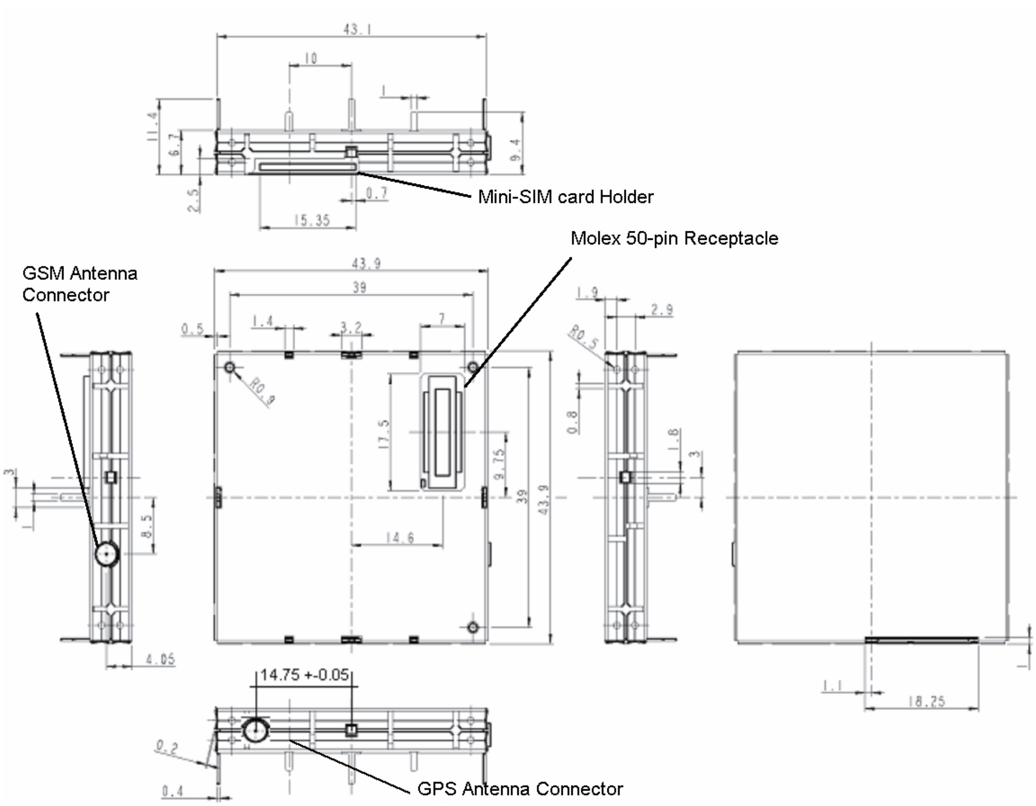

Figure 5-6. Telit GSM/GPRS/GPS Module GM862-GPS
Used with permission pending from Telit® Wireless Solutions.

5.5.3 GSM Antenna

Device: GSM/GPRS Antenna.
Part Number: FXP07.09.0100A
Manufacturer: Taoglas.
Description: Flexible PCB Penta Band GSM Antenna.

In order to select a suitable GSM/GPRS antenna for the GM862 module, mobile network operating frequencies were defined. Since the GM862 supports the following frequencies: 80 MHz in EGSM 900, 70 MHz if GSM 850, 170 MHz in DCS, 140 MHz PCS band. For the LTU the frequency selected is 70 MHz which corresponds to GSM 850. The mobile service provider that supports GSM 850 is AT&T.

Moreover, the GSM antenna was selected following the specifications of the GM862: 50 Ω impedance, less than 3dBi gain, and input power higher than 2 W peak. In order to match the female RF connector of the GM862, a coaxial cable with MMCX Angle Plug Crimp is used.

Figure 5-7. Taoglas Flexible PCB Penta Band GSM Antenna.

The GSM antenna is connected to the side of the GM862 module, and the cable is routed along the side wall of the enclosure and extends through the outlet hole. The antenna cable is secured using a nut and star washer that is provided with the bulkhead/swivel antenna.

5.5.4 GPS Antenna

Device: GPS Antenna.
Part Number: GPS3620
Manufacturer: EAD.
Description: Internal Actve GPS Antenna.

The GPS antenna was selected according to the specifications of the GM862. The frequency of GPS L1 is at 1575.42 MHz, bandwidth of +/- 1.023 MHz, between 1.5 dBi and 4.5 dBi gain, 50 Ω impedance, typical amplification of 20 dB, between 2.5 to 5 V DC supply voltage, and 20 mA typical current consumption.

The GPS antenna is connected to the MMCX plug located on the side of the GM862 module. In order to save space, the antenna was routed towards the bottom of the GM862 module and the PCB.

Figure 5-8. EAD Internal Active GPS Antenna. GPS3620

5.5.5 PCB and Supporting Components

A PCB was designed in order to hold all circuitry components that support the power supply, on/off button, signaling, input and output interfaces, as well as antennas.

5.5.5.1 Connectors

The following sections contain information on the various connectors utilized on the LTU and its internal components.

5.5.5.1.1 50-pin board-to-board connector for GM862-GPS

Device: 50-pin, board-to-board female connector
Part Number: 501920-5001
Manufacturer: Molex
Description: Board-to-board header SMD 50 pin connector, female, 0.5mm pitch

GM862-GPS module has a board-to-board male connector type CSTP 50 pin vertical SMD Molex 52991-0508. The required mating connector that is mounted to the PCB is the Molex 501920-5001, and its layout is shown in Figure 5-9.

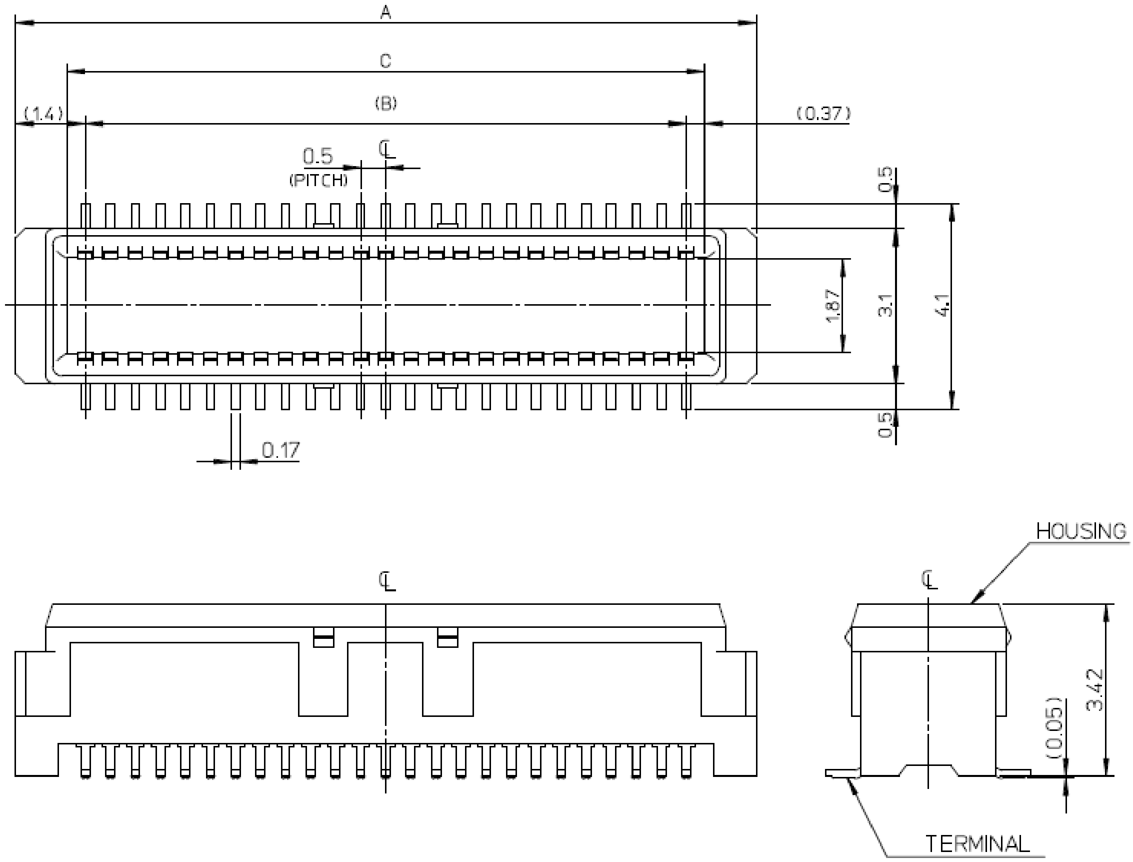

Figure 5-9. Molex Board-to-board connector dimensions. 501920-5001
Used with permission from Molex®.

5.5.5.1.2 20 pin IC Socket for Microcontroller

Device: 20 pin IC socket
Part Number: D2820-42
Manufacturer: Harwin.
Description: 20 pin IC socket assembly 2.54mm pitch.

In order to facilitate programming of the MSP430 microcontroller, a 20-pin IC socket will be used on the PCB unit.

5.5.5.1.3 USB Micro-B Serial Receptacle

Device: USB Micro-B serial receptacle
Part Number: D2820-42
Manufacturer: Harwin
Description: 20 pin IC socket assembly 2.54mm pitch

The USB Micro-B receptacle allows connection to the Universal Power Charger in order to provide power to the charge manager, which properly charges the lithium ion battery. The benefits of using this connector and its corresponding cable and charger are that since 2009 several cellphone manufacturing companies have agreed to embrace to a Universal Charger Solution which defines the Micro and Mini connectors as the choice for future design. This allows users to utilize chargers across different products and whenever they change cellular devices. This connector is labeled "USB" on the schematic diagrams. Figure 5-10 shows the USB Micro-B receptacle and the charge manager IC.

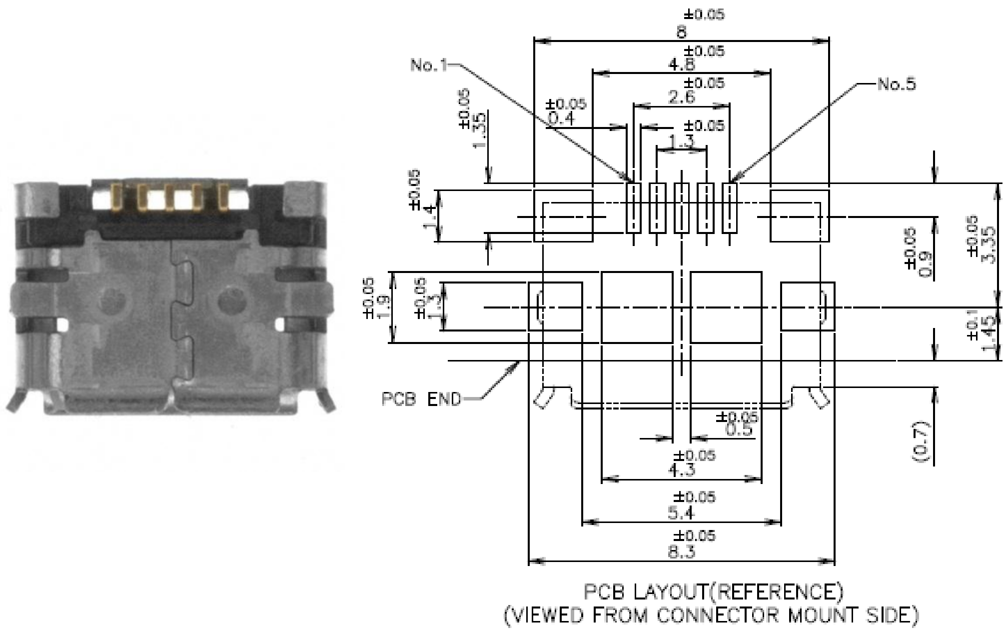

Figure 5-10. TE Connectivity Micro-B USB Receptacle Diagram
Used with permission pending from TE Connectivity.

5.5.5.1.4 Battery Receptacle

A high current, compact, two-connector receptacle designed for surface mount will be used to connect the battery to VBATT pins 1, 3, 5, and 7. This connector corresponds to the proper gender mate for the battery and allows for easy interchange in case of malfunction or damage to the battery. This connector is labeled "CON3" on the schematic diagrams.

5.5.5.2 On/Off Switch

5.5.5.2.1 ON/OFF Switch

Device: Miniature Slide Switch
Part Number: OS102011MA1QN1
Manufacturer: CK-components
Description: SPDT 2-position slide switch, 90 degree angle through hole

A switch mounted on the side of the PCB allows the device to be turned on and off. The tab actuator of the switch protrudes through the box so the user can access it easily. This switch is a 2 position slide with 90-degree through-hole solder pins. The switch disconnects all power downstream of the charge manager output and the battery. This allows the LTU unit be turned on or off. See the figure below for details on this component.

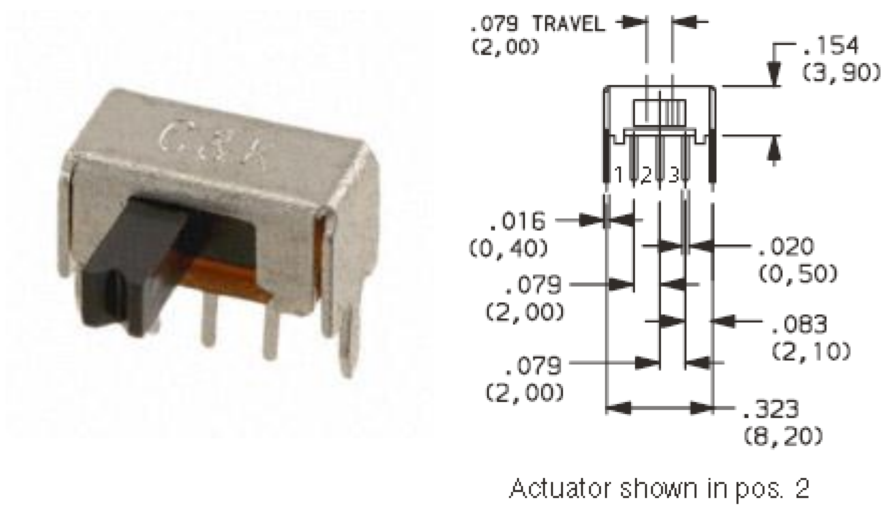

Figure 5-8. CK-components Slide switch picture and dimensions. OS102011MA1QN1
Used with permission pending from CK-components.

5.5.5.3 Output Signals and Status Indicators

The following sections contain details on the various LEDs implemented in this project for use in providing status to the user as well to accomplish other functions.

5.5.5.3.1 Network Status LED

A green LED is used to indicate Network Service Availability. This LED is connected to the open collector pin 35 STAT_LED of the GM862. The output of this pin is the inverse of the status signal. A 470Ω resistor is used in series to the 2V LED and is pulled up to 3.7V VBATT for correct operation.

5.5.5.3.2 GPS Status LED

A yellow LED is used to indicate GPS status and availability. This LED is connected to pin 31 GPIO1 of the GM862. The output of this pin is the inverse of the status signal. A 470Ω resistor is used in series to the 2V LED and is pulled up to 3.7V VBATT for correct operation. The GM862 does not offer an output with GPS status. Therefore, with the use of AT commands, the output pin GPIO1 is programmed in order to provide GPS signal status and availability.

5.5.5.3.3 Microcontroller Status LED

A red LED is used to indicate microcontroller status and program debugging.

This LED is connected to Output 2.2 pin 11 of the MSP430. The output of this pin is the inverse of the status signal. A 470Ω resistor is used in series to the 2V LED and is pulled up to 3.7V VBATT for correct operation.

5.5.6 Mini-SIM card

The GM862 module has an internal SIM card interface that accepts and reads a Mini-SIM 1.8V and 3V (ISO/IEC 7810:2003, ID000). The SIM card plays an important role in establishing communication and allowing access to the mobile network provider. The SIM card is inserted into the GM862 module as shown in the figure below:

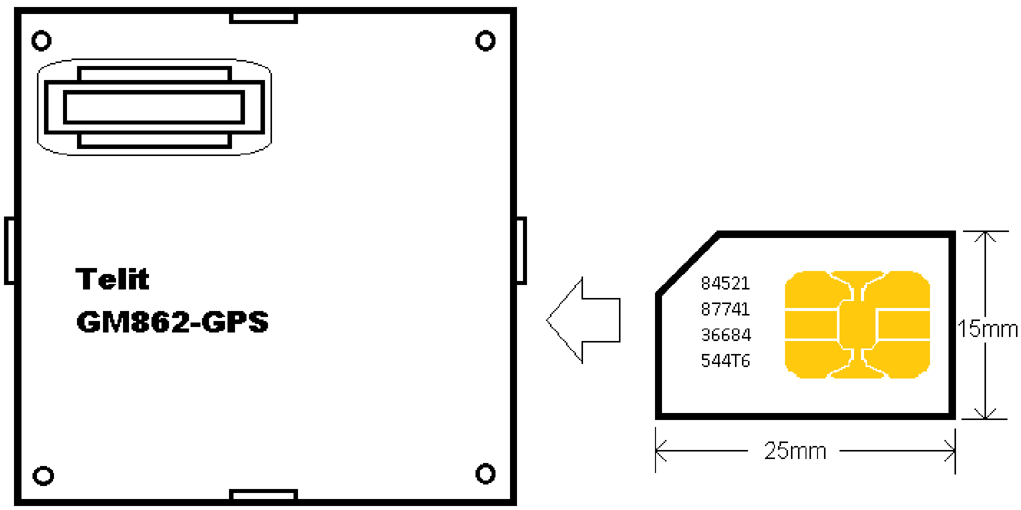

Figure 5-11. Mini-SIM card dimensions and insertion into the GM862-GPS
Designed by Jose Mousadi

Device: Enclosure, box.
Part Number: 71100-510-000 JM-24
Manufacturer: PacTec.
Description: 2.4” x 3.8” x 1” ABS box, black.

A plastic two piece enclosure will be used to house the LTU components. Preliminary dimensions allowed for a 4” wide by 4” high by 0.5” deep enclosure.. It was designed to provide a means for attaching the PCB board to the enclosure while providing enough space for routing the mini-coax cables for the two antennas and allowing the On/Off switch to protrude through the side wall. The enclosure is made of material that can be machine-drilled to create holes for LEDs and a Micro-B USB power receptacle.

5.5.7 Battery and Power Supply

5.5.7.1 Battery

Device: Lithium Ion Rechargeable Battery
Part Number: UBP002
Manufacturer: Ultralife
Description: 3.7V, rechargeable Lithium Ion Battery, 900mAh

Since all the peripherals are be driven by the GM862 module, the battery was chosen according to the requirements of the module. Modules with a SW release 7.03.x00 or newer require a nominal supply voltage of 3.8 V. Its normal operation voltage range is between 3.4 V to 4.20 V. This module also has an extended operating voltage range between 3.22 V and 4.50 V. The supporting circuitry has a 100µF installed in parallel to the VBATT pin of the GM862. See the figure below for details:

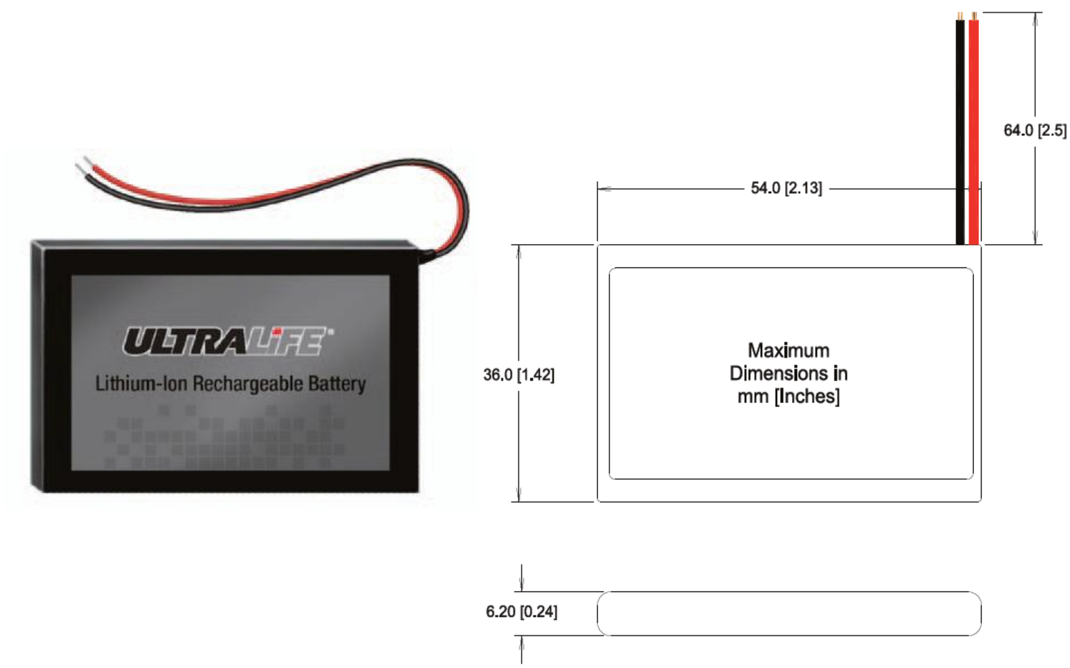

Figure 5-12. Ultralife Rechargeable Li-Ion Battery Diagram
Used with permission pending from Ultralife.

5.5.7.2 Charge Manager

Device: Li-Ion Charge Manager
Part Number: BQ24204
Manufacturer: Texas Instruments
Description: Li-Ion charge manager IC for current-limited applications

A charge manager is utilized to charge the lithium ion battery following the recommended charging current curve. This charge manager allows the user of the LTU to connect a wall adaptor used for cellular phones to charge the battery via a Mini-B USB receptacle. The charge manager has been selected to match the specification of the lithium ion battery. The battery charger is designed to support a 3.7 V Lithium-Ion rechargeable battery with a suggested capacity of 900 mAh. This IC provides a charge regulation voltage of 4.2V, battery detection, pre-charge conditioning, and charge termination. See the figure below for reference. Another benefit is that it is designed to work with current-limited wall supplies, so the user will be able to use the same wall adaptor as ones for some smartphones. A zener diode is also implemented in order to protect against inverted voltages and voltages over 6.4 VDC.

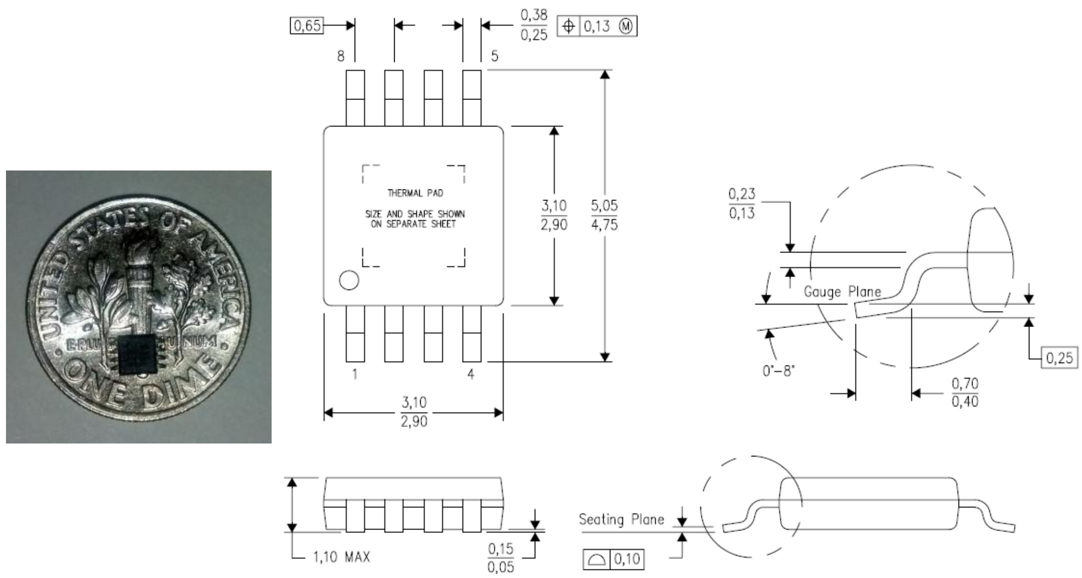

Figure 5-13. Texas Instruments Charge Manager Diagram
Used with permission pending from Texas Instruments.

5.5.8 I/O Components and Sensors

The following sections contain detailed information and specifications on the Input/Output (I/O) components and sensors in the LTU.

5.5.8.1 Ambient Light Sensor

Device: NPN Phototransistor
Part Number: BPW85A
Manufacturer: Vishay
Description: NPN Phototransistor T-1 package

An NPN phototransistor is connected to Input 1.3, pin 5 of the microcontroller. The phototransistor can detect wavelength between 450 and 1080 nm. See the figure below for Relative Spectral Sensitivity Vs. Wavelength. The gain has been calculated by adjusting resistor to 4.1KΩ. When no light is present, no current is flowing through the transistor, which keeps the input pin of the microcontroller at a logical zero (low). When ambient light is present, current passes through the transistor, and once the voltage threshold of 2.2V is surpassed, a logical high is read by the microcontroller. It is important to note that pin 5 of the microcontroller must be configured as an input pin.

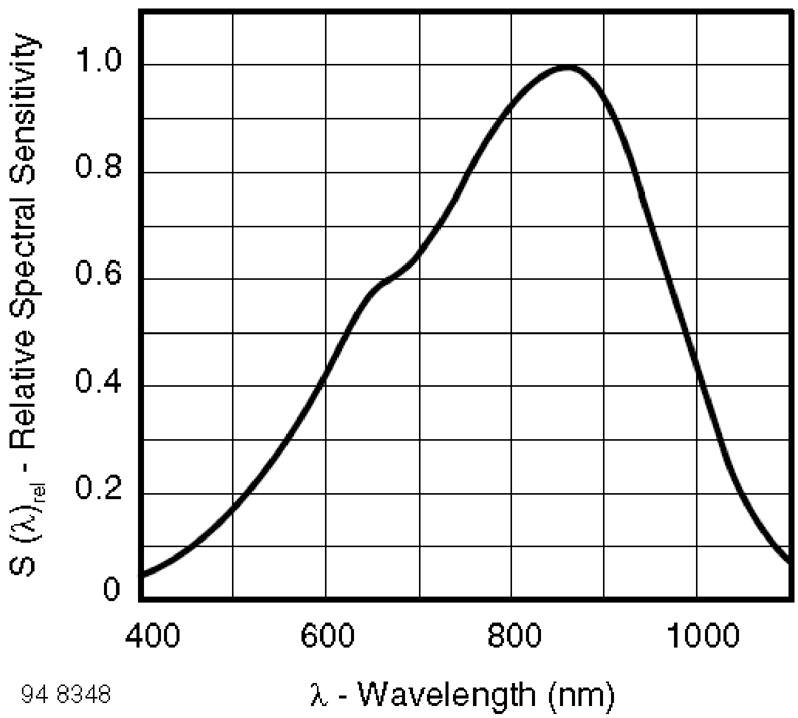

Figure 5-14. Vishay NPN Phototransistor Relative Spectral Sensitivity vs Wavelength
Used with permission pending from Vishay.

5.5.8.2 Barometric Pressure Sensor

Device: High Accuracy, Absolute Pressure Sensor IC.
Part Number: MP3H6115A
Manufacturer: Freescale Semiconductor.
Description: High temperature accuracy integrated silicon pressure sensor for measuring absolute pressure.

The IC is connected to the Input 1.0, pin 2 of the microcontroller. This IC measures absolute pressure between 150 to 115kPA (2.2 to 16.7 psi). The sensor's output is a DC voltage linearly proportional to the measured pressure. The following equation shows the transfer function between pressure and voltage.

$$Vout = Vcc \times (0.009 \times P - 0.095)$$

See Figure 5-15 and Figure 5-16 Pressure Sensor IC for additional information. Other properties of the pressure sensor are that it is temperature compensated, has a fast response time of 1/0 ms, and provides accuracy of 1.5% VFSS.

It is important to mention that the input pin on the microcontroller is set to read analog voltage. The microcontroller uses the Analog to Digital converter (ADC10) to generate a digital value for all calculations. The output pin 4 of the pressure sensor also has a capacitor and a resistor that functions as a low bypass filter thereby avoiding any high frequency transients.

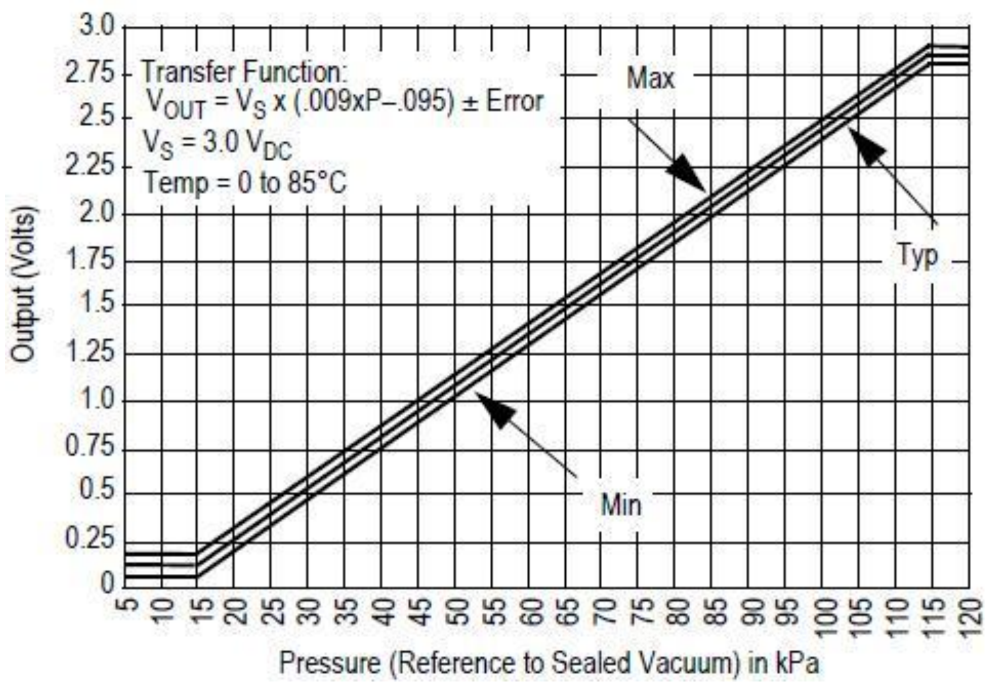

Figure 5-15. Freescale Semiconductors. Pressure Sensor Transfer Function.
Used with permission pending from Freescale Semiconductors.

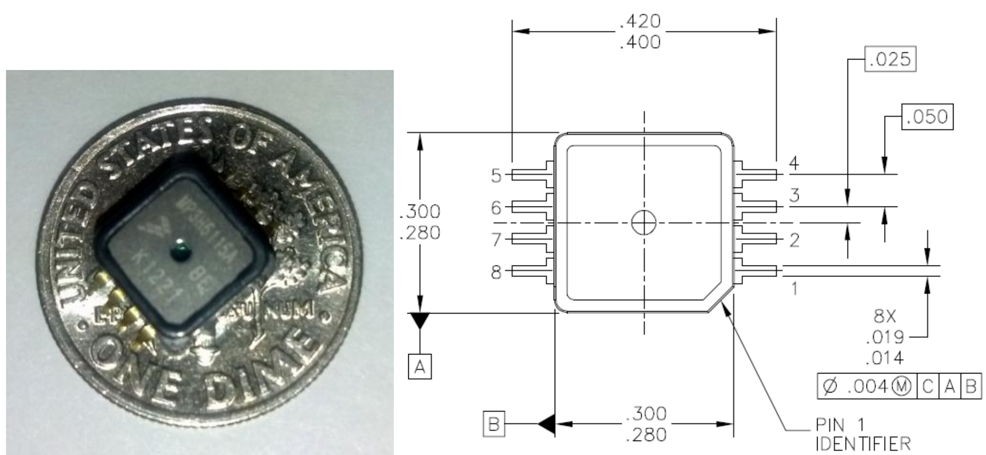

Figure 5-16. Freescale Semiconductors. Pressure Sensor Diagram
Used with permission pending from Freescale Semiconductors.

5.6 Software Components

5.6.1 On-Chip Software

The language used to write and simulate the on-chip software will be the C programming language because it is the required language for the MSP430 microcontroller.

5.6.1.1.1 Data Logger

The on-chip software is used to send and receive data to and from the Atmel flash memory chip, which functions as the LTU data logger. The Atmel has a 4 megabit capacity, with 4,325,376 total bits of memory and 2,048 pages, which is more than enough space for the purposes of this project. Due to this fact, there was no necessity to conserve memory space, and each event logged by the microcontroller was allocated one page in memory.

5.6.1.1.2 Flash Limits

A significant challenge faced in the software development process was the limited Random Access Memory (RAM) available in the MSP430 microcontroller. Only 256 bytes of RAM are available in this microcontroller, which significantly limited the number and size of variables that could be utilized by the on-chip software. Due to this, provisions in the microcontroller software had to be made to conserve as much RAM as possible and to use that space only for variables that were absolutely necessary.

5.6.1.2 GSM/GPS Module Software

The Telit GM862-GPS module, which contains both a GSM modem and a GPS receiver, is operated by the microcontroller through a serial communication interface using Attention (AT) commands. The serial communication protocol implemented for this interface is Universal Asynchronous Receiver Transmitter (UART).

The GPIO interface allows direct control of and communication with the GPIO pins on the GM862 module, increasing the overall efficiency of the module's operation. This interface is used to control such items as the status LED, which is used as an indicator of whether or not the module has established a connection with a GSM network and if so, what the signal strength of that connection is.

5.6.2 GPS Server

The GPS server was the most critical component for the purpose of meeting the multi-device tracking requirement. Without a GPS server, communication capabilities would be reduced to direct messages sent from each individual LTU to the user.

Due to the fact that our team did not have any experience with databases or servers, it was determined that, given the time constraints of the project, a pre-made GPS server would have to be downloaded from the internet rather than attempting to design and develop one from the ground up.

After extensive research, Open Source GPS Tracking System (OpenGTS) was selected as the best option for this project. OpenGTS is a free, open source software package developed by Geo Telematic Solutions Incorporated. According to the project website, OpenGTS is “the first available open source project designed specifically to provide web-based GPS tracking services for a ‘fleet’ of vehicles.” In the case of our project, of course, the “fleet of vehicles” is the user’s luggage.

While the OpenGTS software itself is coded entirely in Java, it utilizes a number of supporting software packages and features in order to function as a working GPS server. The figure below shows the overall architecture of OpenGTS as a system.

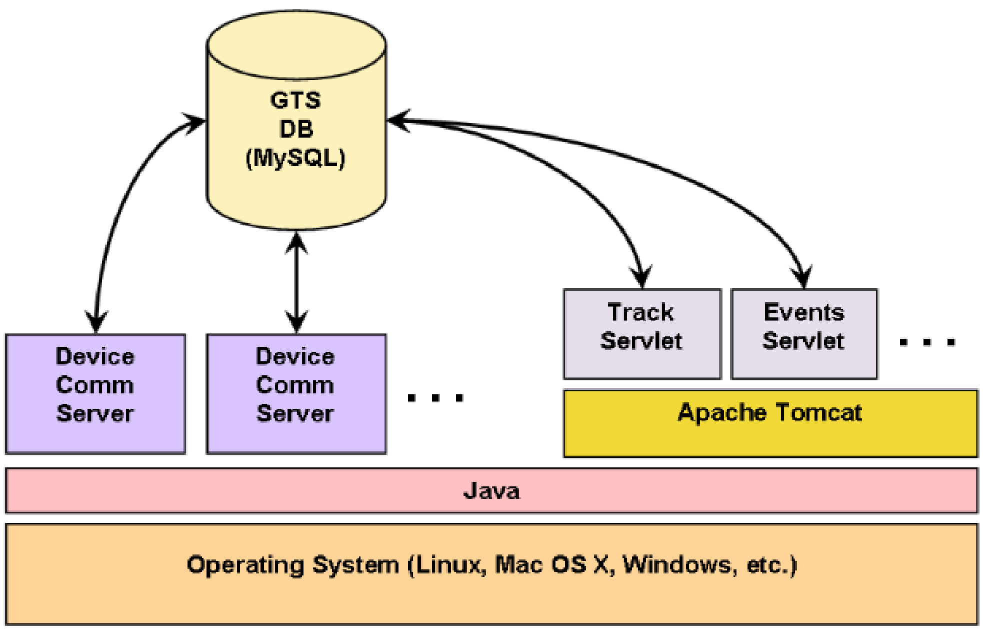

Figure 5-17. OpenGTS System Architecture

Although OpenGTS is developed in Java, as the figure above indicates, there are additional software packages which complement the core OpenGTS software. Apache Tomcat software is used for the web interface, and MySQL functions as the database backend.

One such feature is the highly customizable, front end web portal. This webpage is preconfigured to work in unison with the server and database backend, which was of great benefit to our team as we have very limited experience in the realm of web development. However, the OpenGTS developers have provided a variety of options which the user can customize as desired. For example, reports can be generated to show the status of one or more tracking devices. The reports themselves can be customized to show certain types of data and certain levels of detail. Another useful feature is the geozone or geozone option. Geozones are virtual boundaries that can be set, usually as a certain radius around a given point, and alerts can be set to notify the user when a tracking device enters and/or leaves a geozoned area.

Another feature of OpenGTS, and one that was crucial to its implementation in our project, is the fact that it is device independent. While a number of GPS tracking devices are already supported by the OpenGTS software, their website claims that “with custom coding, other devices can also be integrated.” Since a GPS tracking device is what we are developing, it will be necessary for us to research and produce whatever custom coding is necessary to integrate our device with the OpenGTS software.

5.7 Interface Control and Communication

5.7.1 GPS Satellite to LTU

In an ideal situation, the LTU would be constantly receiving signals from one or more GPS satellites. However, in reality this is certainly not the case. It is entirely plausible that at certain locations and for varying lengths of time, the LTU will not be receiving GPS satellite signals. This could be due to a dense building which blocks the signal, a remote or low-lying area with poor reception, or any number of other reasons. However, even if the LTU were to be in constant connection with GPS satellites, this would be an enormous drain on the power supply because the GPS receiver and microprocessor would be constantly operating and handling data.

In order for a GPS coordinate to be calculated, the LTU must be in range of at least four GPS satellites. This allows for the longitude and latitude to be determined based on the intersection of the four satellites’ coverage areas.

The LTU contains a dedicated antenna to receive GPS satellite signals. This antenna is located inside of the LTU enclosure in order to provide increased protection for the antenna and maintain the streamlined form factor of the LTU. Were the antenna located outside of the enclosure, this would increase the receptivity capability of the LTU. However, this would provide significant risk of the antenna being broken due to being caught on something as the LTU is being placed inside a piece of luggage or being removed from a piece of luggage. As no member of our team had experience in antenna design, the GPS antenna was purchased as a Commercial Off The Shelf (COTS) part from a vendor.

5.7.2 Microprocessor to GPS Receiver

The chief function of the microprocessor in relation to the GPS receiver was to control the following actions:

- startup and shutdown
- power mode changes
- data collection

Because the GPS receiver is a major consumer of power, it was imperative that the GPS receiver be used only when necessary. At initial power up of the LTU, the GPS module is commanded into a low power mode via an AT command sent from the microcontroller. The GPS module remains in this mode unless it receives another AT command. Once the received AT command has been executed, GPS module automatically returns to the low power mode it was in previously.

Incoming GPS signals are processed internally by the GPS receiver, and the resulting data is made available on external data ports as well as directly to the GSM module via internal serial line. There are two external data ports on the GPS receiver, a modem serial port and a GPS serial port. These ports are described in detail in the table below:

Table 5-2. GM862-GPS External Serial Ports

Port	**Format**	**Data Rate (bps)**	**Description**
Modem Serial	SiRF Binary	57,600	Control GPS and GSM via AT commands in a user-generated script.
GPS Serial	NMEA	4,800	Access NMEA sentences from GPS receiver.

It should be noted that while the Modem Serial port is external with respect to the GPS receiver, it is internal with respect to the GM862 chip itself. The Modem Serial port is, however, externally accessible on the GSM modem. A visual representation of these ports and their sources is provided in the figure below:

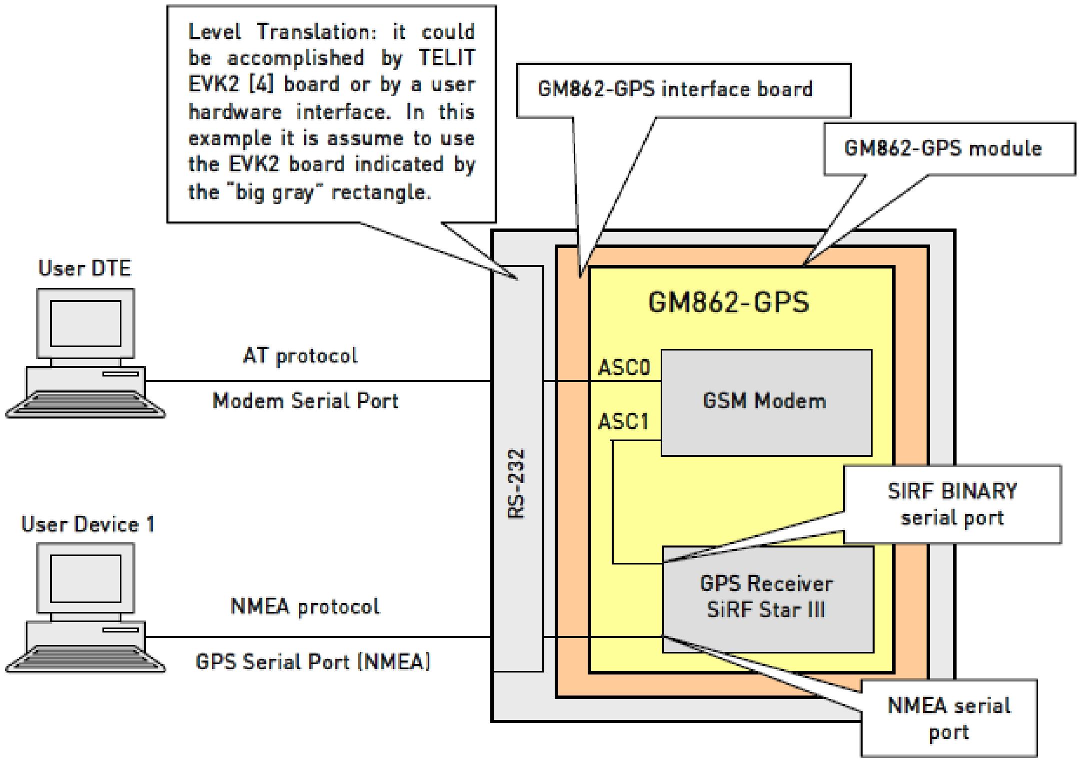

Figure 5-18. GM862-GPS Serial Ports
Used with permission pending from Telit® Wireless Solutions

The outer gray rectangle in this figure represents Telit's EVK2 evaluation board. This board was not used in the final design or for testing purposes due to a much less expensive evaluation board being obtained.

The primary mode of communication with the GM862 for this project was the SiRF Binary protocol. This method implements the use of AT commands, which are used to control the GSM modem directly and the GPS receiver indirectly. Thus the microprocessor sends commands to the GPS receiver via the ASC1 serial line connecting the GSM module to the GPS receiver.

5.7.2.1 GPS Receiver Control

In order to control the GPS receiver, the GSM module's ASC1 port must be set to Controlled Mode. The AT command used to set the ASC1 port to Controlled Mode is AT$GPSD=<device type>. In this case, the device type is 2, so the actual command that would be sent to the GPS receiver is AT$GPSD=2. However, the GPS module's default mode is Controlled Mode, so this command does not need to be sent unless the Controlled Mode is changed. Any AT commands sent to the GPS receiver before Controlled Mode is activated are not received.

5.7.2.2 GPS Receiver Power

The AT$GPSP command is used to turn the GPS receiver on or off. Sending the command AT$GPSP=0 powers off the GPS receiver, and sending AT$GPSP=1 powers on the GPS receiver. When the LTU is on, the GPS receiver is typically in either an active/ready state or a low power state. The power state of the GPS receiver is determined by the settings programmed in the firmware designed by our team members.

For the majority of the time the LTU is powered on, the GPS receiver will be in a low power mode. However, at predetermined intervals, the receiver will be activated to a full power mode and allowed to search for and acquire GPS signals. Once a sufficient number of GPS signals have been acquired and a GPS coordinate has been calculated, the GPS receiver automatically returns to a low power mode.

The low power mode that is used for the GPS receiver is called tricklepower mode. According to the Telit documentation, in tricklepower mode ”power to the SiRF chipset is cycled periodically, so that it operates only a fraction of the time; power is applied only when a position fix is scheduled.” The AT generic command for Power Saving mode is AT$GPSPS=<mode> [,<PTF_Period>]. The mode variable specifies what type of power mode to set the GPS receiver.

For this project, the default time between GPS fix attempts is a predetermined number of minutes that corresponds to the duration of the operation cycle. Every certain number of minutes, the GPS receiver enters full power mode and attempts to acquire GPS satellites and generate a fix. Upon entering full power mode, the GPS receiver will be allowed 60 seconds to accomplish this task. If a GPS fix is not acquired within 60 seconds, the receiver returns to a low power mode.

The GPS receiver can be woken from a low power mode using the AT$GPSPS command or by sending any other AT command to the receiver. Sending the command AT$GPSPS=0 sets the GPS receiver to full power mode. The difference between sending the AT$GPSPS command and sending any other AT command is that the AT$GPSPS command changes what power mode the GPS module returns to after processing a command.

5.7.2.3 GPS Position Data

The most recent GPS position is retrieved from the GPS receiver via the SiRF Binary port by sending the AT$GPSACP command. This command returns the latest GPS position as follows:

$GPSACP:<UTC>,<latitude>,<longitude>,<hdop>,<altitude>,<fix>,<cog>,<spk>,
<spkn>,<date>,<nsat>

The variables in the GPSACP status message are described in detail in the table below:

Table 5-3. GPSACP Status Message Details

Variable	Format/Values	Description
UTC	hhmmss.sss	Universal Coordinated Time (UTC)
latitude	ddmm.mmmm	Latitude (d = degrees; m = minutes)
longitude	dddmm.mmmm	Longitude (d = degrees, m = minutes)
hdop	x.x	Horizontal Dilution of Precision
altitude	x.x	Altitude (in meters)
fix	0,2,3	0 = Invalid fix; 2 = 2D fix; 3 = 3D fix
cog	ddd.mm	Course Over Ground (d = degrees; m = minutes)
spkm	x.x	Speed over ground (in km/hr)
spkn	x.x	Speed over ground (in knots)
date	ddmmyy	Date of Fix (d = day; m = month; y = year)
nsat	nn	Total number of satellites in use

A summary of the AT Commands used to control the GPS receiver are listed in the table below:

Table 5-4. GPS Receiver Commands

Command	Description	Variable Definition
AT$GPSD=<device_type>	Sets the ASC1 port to controlled mode.	<device type> 2 = Controlled Mode
AT$GPSP=<status>	Turns the GPS receiver on or off.	<status> 0 = Turn off GPS 1 = Turn on GPS
AT$GPSACP	Reads the acquired GPS position.	N/A
AT$GPSPS=<mode> [,<PTF_Period>]	Puts the GPS receiver in Power Saving mode.	<mode> 0 = Full power mode (disable Power Saving mode) 1 = Tricklepower mode 2 = Push-to-fix mode 3 = N/A (GE864 only) <PTF_Period> (0-300000) # = number of seconds

5.7.3 Microprocessor to GSM Module

5.7.3.1 Power Modes

When the GM862 chip is powered on, the GSM module is turned on automatically. In order for the GM862 to be turned on, the ON pin must be tied low for a minimum of one second. When the low signal is released, the GM862 powers on.

To help reduce power consumption and extend battery life, the GM862 module comes equipped with several different power saving modes. The command used to control the power mode is the AT+CFUN command. This command provides the option of selecting a number of different power modes which can be used to mitigate the overall power consumption of the GSM module.

Calling AT+CFUN=0 sets the GSM module to non-cyclic sleep mode. This mode is the lowest power mode available without turning the GSM module completely off. It provides only minimum functionality, and it is important to note that even the AT command interface is shut down in this mode. The only way for the mode to be exited and the GSM module brought back to full power mode is through the occurrence of a wake-up event or by the receipt of a high RTS signal. Either of these two events brings the unit out of the non-cyclic sleep mode and into full power mode. Because the non-cyclic sleep mode offers the smallest power consumption, this mode will be activated when the GSM module is not attempting to transmit a message. This means the GSM module is in non-cyclic sleep mode for 15-minute intervals, coming back to full power mode after each interval to attempt to transmit a message or multiple messages.

Calling AT+CFUN=1 sets the GSM module to full functionality and also disables power saving. When power saving is disabled, the module does not reduce its power usage during the time it is idle. This mode is the default setting of the GM862 module.

Calling AT+CFUN=2 disables the transmitting capability of the GSM module. In other words, the module is only able to receive incoming signals and messages while in this mode. For this project, the sole purpose of the GSM module is to transmit messages, so this mode was not utilized.

Calling AT+CFUN=4 disables both the transmitting and receiving capabilities of the GSM module. This mode may be used to deregister the module from a network or to deactivate the SIM card. This mode will be entered when a pressure decrease event is detected, signifying that the LTU is on an aircraft. Because the receive capability of the GSM module is disabled in this mode, the only way to exit this mode is by sending another AT+CFUN command with a parameter for a different power mode.

Calling AT+CFUN=5 brings the GSM module into a full power mode identical to that of setting number 1 with the exception of the power saving capability. Contrary to mode 1, mode 5 enables the power saving option. This means the GSM module enters a low power state to reduce power consumption when it is idle. When the module is not idle, it is in its full power state with all functionality available.

Calling AT+CFUN=7 activates the cyclic sleep mode. This mode is similar to the non-cyclic sleep mode, but the main difference between the two is that, in cyclic sleep mode, the AT command interface remains active. This allows the GSM module to be awakened with an AT command rather than waiting for a wake-up event or a high RTS signal. Another unique feature of the cyclic sleep mode is that it allows the GSM module to sense when data is being transmitted on the serial interface. As long as the serial interface is in use, the GSM module remains active. Cyclic sleep mode can only be deactivated by sending the AT+CFUN=1 command, which brings the GSM module to the full power mode. From there, other power modes may be selected as desired.

Calling AT+CFUN=9 puts the GSM module in a mode identical to that of mode zero with the exception of the case when a GPRS packet is received by the module. When in mode zero, an incoming GPRS packet event does not have any effect on the power mode. However, when in mode 9, an incoming GPRS packet event brings the GSM module out of its current power mode and into mode 1, which is the full power mode with power saving disabled.

The GM862 can be shut down using either the AT#SHDN command or by tying the ON pin low. As with the power on procedure, the ON pin must be tied low for at least one second and then released in order to initiate the shutdown procedure. While this is the proper way to shut down the GM862, for the implementation of this project, the module is powered off by simply removing power from the module. This action occurs when the user moves the power switch on the LTU to the off position.

While this is not the best practice for shutting down the module, it provides the easiest solution from both a hardware and software perspective. The GM862 module is capable of handling a force shutdown, so this method does not pose a risk of damaging the GM862 or other components within the LTU. Upon power up after a forced shutdown, the GM862 simply applies its default settings, and everything functions as it would after a proper shutdown.

5.7.3.2 Band Selection

The GSM module is a quad-band modem, meaning it can operate on four different bands: 850/1900 and 900/1800. Rather than attempting to manually connect to a specific cellular band, the GSM module is allowed to automatically select a band. This is accomplished using the following command: AT#AUTOBND=1. After setting the band selection to logic 1, the setting is stored in the GM862 and does not have to be reentered unless the GM862 is reset.

5.7.3.3 Storing SMS Messages

Because a cellular network may not be available every time the GSM module wishes to send an event notification, all event notifications are stored in memory before they are attempted to be sent. For this project, two different methods of storing these messages in memory were developed. The first method stores the flash memory index of the notification in a buffer only if the message was not sent successfully. During each operation cycle, the buffer is checked to determine whether any messages are waiting to be sent, and if so, attempts to send those messages again.

In the second method, the messages are generated and stored in the SIM card memory before they are attempted to be sent, and the SIM memory index is stored in a transmit buffer. If any of the messages are not sent successfully, their corresponding SIM memory index is not cleared from the transmit buffer. Again, during each operation cycle, the transmit buffer is checked to determine whether any messages are waiting to be sent. Implementing these measures ensures that messages are not lost when a network is unavailable, but instead messages are ready for transmission when a network is found and a connection is established.

The command used to store an SMS message in memory is the AT+CMGW command. Once a message is stored in memory, it can be sent using the AT+CMSS command. This command allows for a specific message to be loaded from SIM memory and sent to a given destination.

5.7.3.4 SMS Messaging via GSM

The GSM module offers two formats for working with SMS messages: Packet Data Unit (PDU) mode and Text mode. For this project, the Text mode was used exclusively as it is much more user-friendly and does not require the learning of a new syntax, which would be necessary if the PDU mode were implemented. In order to set the module to Text mode, the AT+CMGF command must be used. Sending the command AT+CMGF=1 will set the GSM module to Text mode.

Before this method of sending an SMS message is implemented, the following settings must be set for the GSM module:

- AT#SELINT=2
- AT#SMSMODE=1

To send an SMS message, use the AT+CMGS command. This command alerts the network that a message is going to be sent and specifies the destination address. After this command is sent, the device replies back with a standard message signifying it is ready for the actual SMS message to be sent.

At this point, depending on the current settings of the GSM module, the actual content of the SMS message is entered. The format in which the content is entered allows characters to be entered as they would normally be by a user, as opposed to being entered in a special format. Upon being entered, the characters are converted into the GSM alphabet by the module's internal software.

The GSM module provides the capability to send up to 10 concatenated SMS messages. This feature is handled automatically by the module. An error message is returned if more than 10 messages are attempted to be sent as a concatenated message thread.

After ensuring the GSM module is in text mode by sending the command AT+CMGF=1, the message is sent using the AT+CMGS command. This command allows the destination and content of the message to be specified. The character used to send the message is the Ctrl+Z character, which has a hexadecimal value of 1A.

A summary of the AT Commands used to control the GSM module are listed in table below:

Table 5-5. GSM Module Commands Part 1

Command	Description	Variable Definition
AT#AUTOBND=<value>	Controls automatic band selection.	<value> (0-2) 0 = disable automatic band selection 1 = enable automatic band selection (at next power-up) 2 = enable automatic band selection (immediately)
AT+CFUN=<fun>	Sets the power mode.	<fun> 0 = non-cycle sleep mode 1 = full power mode (power saving disabled) 2 = transmit disabled 4 = transmit and receive disabled 5 = full power mode (power saving enabled)
AT+CMGF=<mode>	Controls the SMS format.	<mode> 0 = PDU mode 1 = Text mode
AT+CMGD=<index>, <delflag>	Deletes an SMS message from the SIM card memory.	<index> *string* = SIM memory index of message to be deleted <delflag> 0 = delete only the message at index 1 = delete all messages except unread messages and sent or unsent mobile originated messages 2 - delete all messages except any unread messages and unsent mobile originated messages 3 - delete all read messages except unread messages 4 - delete all messages

Table 5-6. GSM Module Commands Part 2

Command	Description	Variable Definition
AT+CMGL=<stat>	Lists messages stored in the SIM card memory	<stat> "REC UNREAD" = display any unread received messages "REC READ" = display any read received messages "STO UNSENT" = display any stored messages not yet sent "STO SENT" = display any stored messages already sent "ALL" = display all messages
AT+CMGS=<da>,<toda>	Sends an SMS message via GPRS	<da> *string* = destination address <toda> 129 = national phone number 145 = international phone number
AT+CMGW=<da>,<toda>, <stat>	Stores an SMS message in memory.	<da> *string* = destination address <toda> 129 = national number 145 = international number <stat> “REC UNREAD” “REC READ” “STO UNSENT” “STO SENT”
AT+CMSS=<index>,<da>, <toda>	Sends an SMS message from memory.	<index> *string* = location of message in memory <da> *string* = destination address <toda> 129 = national number 145 = international number

5.7.3.5 Test Setup

The figure below shows an example of one of the initial test setups for establishing the communication interface between the microcontroller and the GSM module within the GM862. On the left hand side is the GM862 mounted on its evaluation board. The lithium ion battery, which was used to power the GM862 evaluation board, can be seen just below the evaluation board. At the bottom of the picture, the MSP430 can be see mounted on its evaluation board, the LaunchPad.

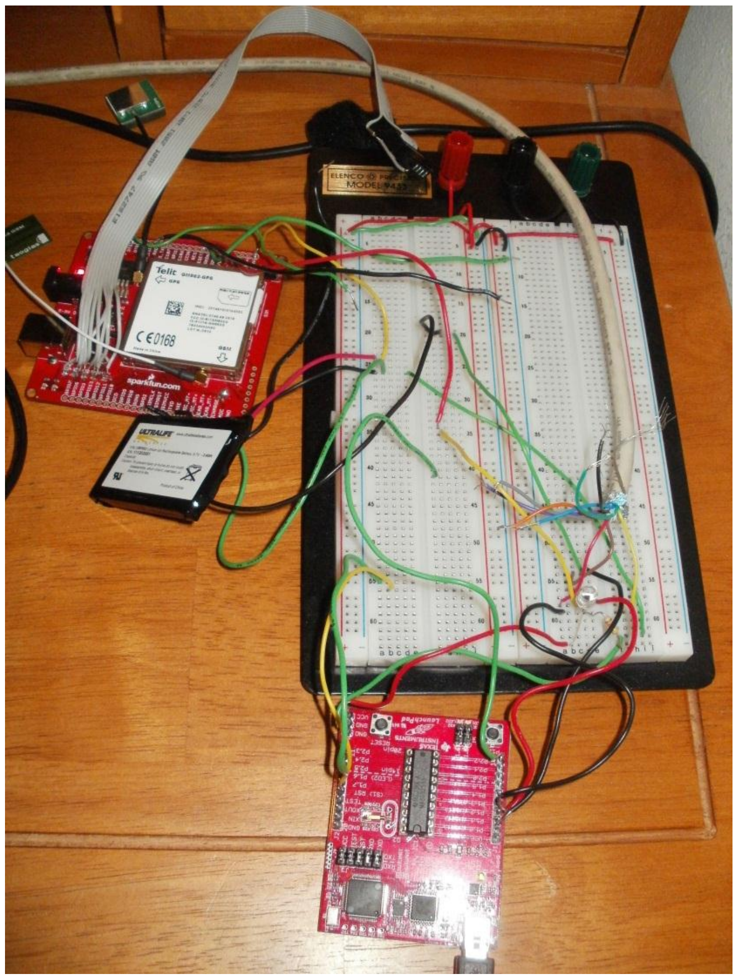

Figure 5-19. Test Setup for Microcontroller to GSM Module Interface

5.7.4 Microprocessor to Memory

The microprocessor serves as the control mechanism and data router for the purposes of the on-chip memory. The main purpose of the memory component is to serve as a GPS data log. While GPS fix data is stored in the GPS module's internal memory, it is also transmitted to the microcontroller, which in turn stores that data in flash memory. The capacity of the flash memory is significantly greater than the internal memory of the GPS module, thus providing a backup storage space should the GPS module's memory reach capacity. While this should not occur during normal operation of the unit, it is possible for this to occur if there is a malfunction in the GSM transceiver or if the LTU is in a location without GSM network coverage for an extended period of time.

The language used to control communication between the microprocessor and flash memory is the C programming language. The communication protocol used for the interaction between the microprocessor and flash memory chip is Serial Peripheral Interface (SPI). This communication protocol was chosen because it is the only protocol with which the flash memory is compatible.

The microprocessor is alerted any time a GPS module obtains a new GPS fix. The software controlling the microprocessor retrieves the GPS data from the GPS module and stores it in memory. Data is not retrieved from the memory chip unless a transmission failure occurs or a memory dump is initiated.

5.7.5 GPS/GSM Module to LED Indicator

The GPS/GSM module used in this project provides a General Purpose Output (GPO) pin which, in the case of this project, is used for a GPS status LED. One of the design requirements for this project was to provide an external indication of the current GPS activity, or lack thereof. This is accomplished through the use of this GPO pin. Control of the output on this pin is provided through the AT#SGPO command. Sending AT#SGPO=1 turns the LED on, and Sending AT#SGPO=0 turns the LED off. Anytime the GPS module attempts to acquire a GPS fix, this LED is flashed three consecutive times to provide a visual indication of the GPS module's activity.

The GM862 also provides a GSM status LED that is used to represent the current network status. The function of this pin is controlled by the AT#SGPO command, and while a number of possible functions are available for this pin, the default setting is utilized as it provides a simple yet informative signal pattern for the LED. The default setting for the status LED output signal is as shown in the table below:

Table 5-7. GSM Status LED Signals

Device Status	LED Status
Off	Off
Searching / Not registered / Shutting down	Blinking (every 1 second)
Connected and registered	Blinking (every 3 seconds)
Call in progress	On (solid)

It should be noted that, because the GSM module is not used for making phone calls, the status LED never reaches the solid "On" state. The default setting for the GSM status LED is as shown in the table above. Therefore no command must be sent to program this setting. The figure below shows the location of the GSM and GPS status LEDs on the LTU:

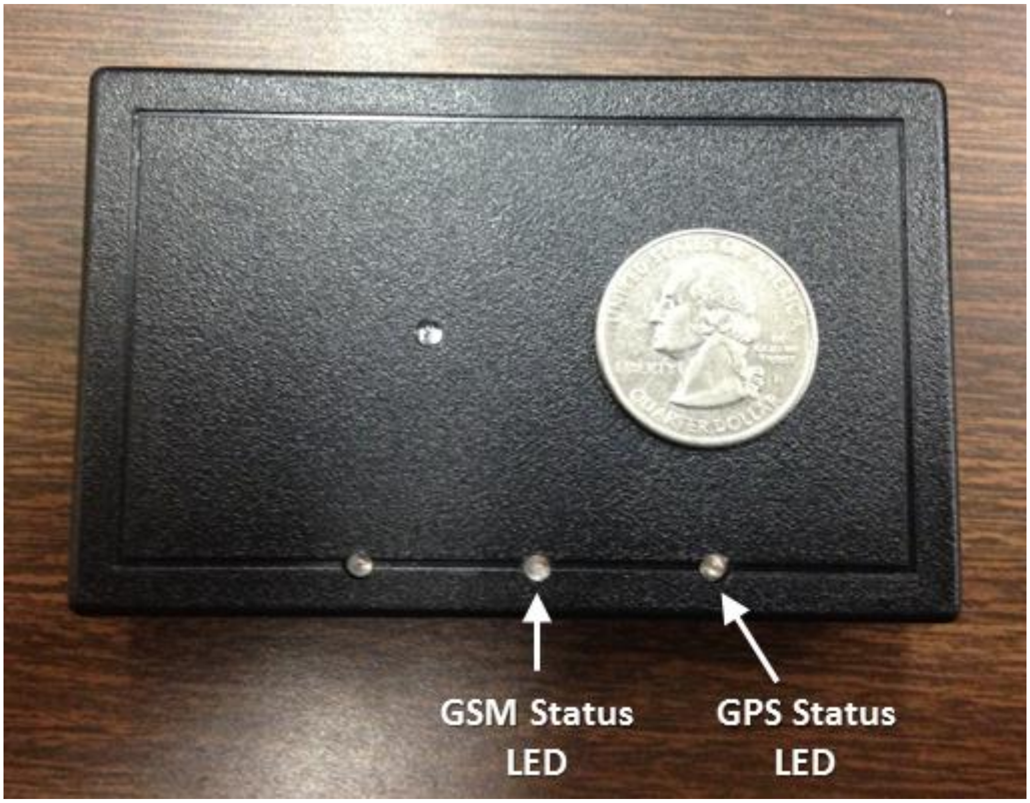

Figure 5-20. GSM and GPS Status LEDs

6 Design Summary

6.1 Overview

The overall function of our system is as follows. The GPS receiver within the LTU receives a GPS signal from the network of GPS satellites. The GPS receiver then processes this signal and generates GPS coordinates and other relevant data such as time, velocity, and altitude. This data is then sent via the GSM module from the LTU to the user in the form of SMS text messages and emails. Text message notifications will contain the time and GPS coordinates of the event, as well as a statement regarding the type of event that occurred. Emails will contain the time and a similar statement, but instead of providing raw GPS coordinates, a Google Maps hyperlink is provided with the coordinates already embedded. The user merely has to click on the hyperlink, and Google Maps opens with a point plotted on the location of that particular event. Figure 6-1 shows a top level functional flow of the luggage tracking system.

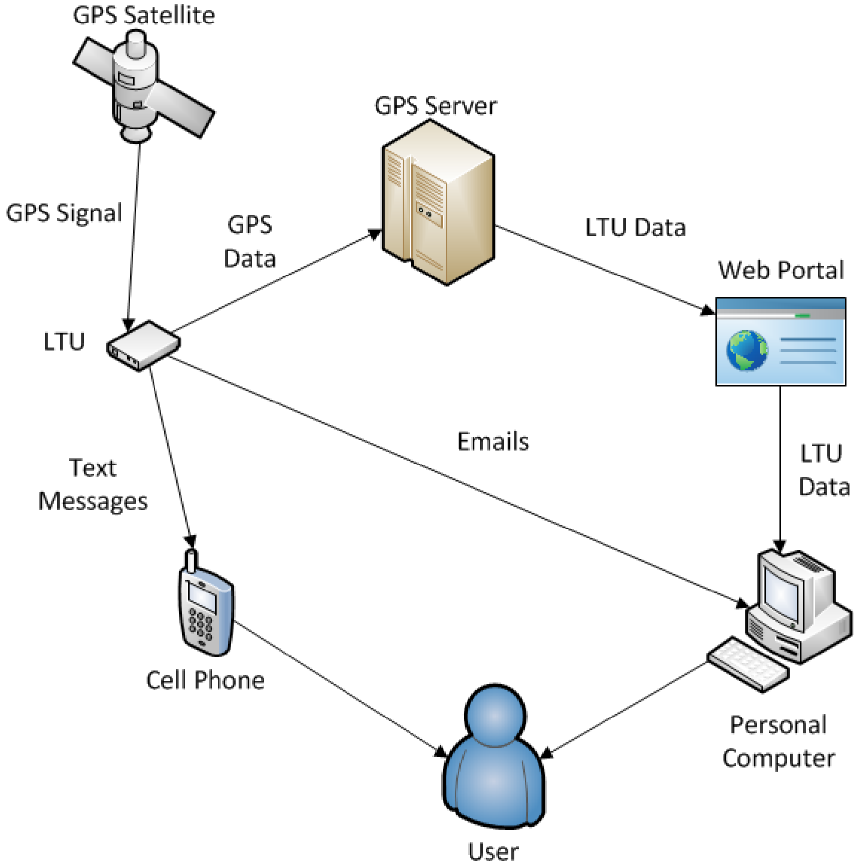

Figure 6-1. Luggage Tracking System

From the time it is powered on, the LTU remains in a continuous loop of operation cycles and waiting periods until it is powered off by the user. For the majority of the time the LTU is powered on, the internal communication components, namely the GSM module and GPS receiver, are kept in a low power state in order to minimize power consumption and extend the battery life as much as possible.

After it is powered on, the LTU first acquires a GPS fix if a sufficient number of GPS satellites are in range. If a fix is successfully acquired, the data is stored in memory and also compiled into a message to be transmitted by the GSM module. If a GPS fix is not successfully obtained, the LTU returns to a low power mode for a predetermined number of minutes before trying again. If, after three attempts, a GPS fix has not been acquired, an error report is logged in memory and sent to the user.

The actual generation of the status message is handled by the microcontroller. The microcontroller assembles the content of the message based on a predetermined format, adds the necessary destination information such as a cellular phone number or email address, and attempts to send the message through the GSM module. If the GSM module is connected to a cellular network, it sends this message as well as any messages stored in the transmit buffer. Once all messages in the buffer have been sent, the LTU returns to a low power mode. After a certain number of minutes have passed, the LTU reenters full power mode and begins its operation cycle again. This operation cycle of the LTU is captured in the activity diagram in the figure below:

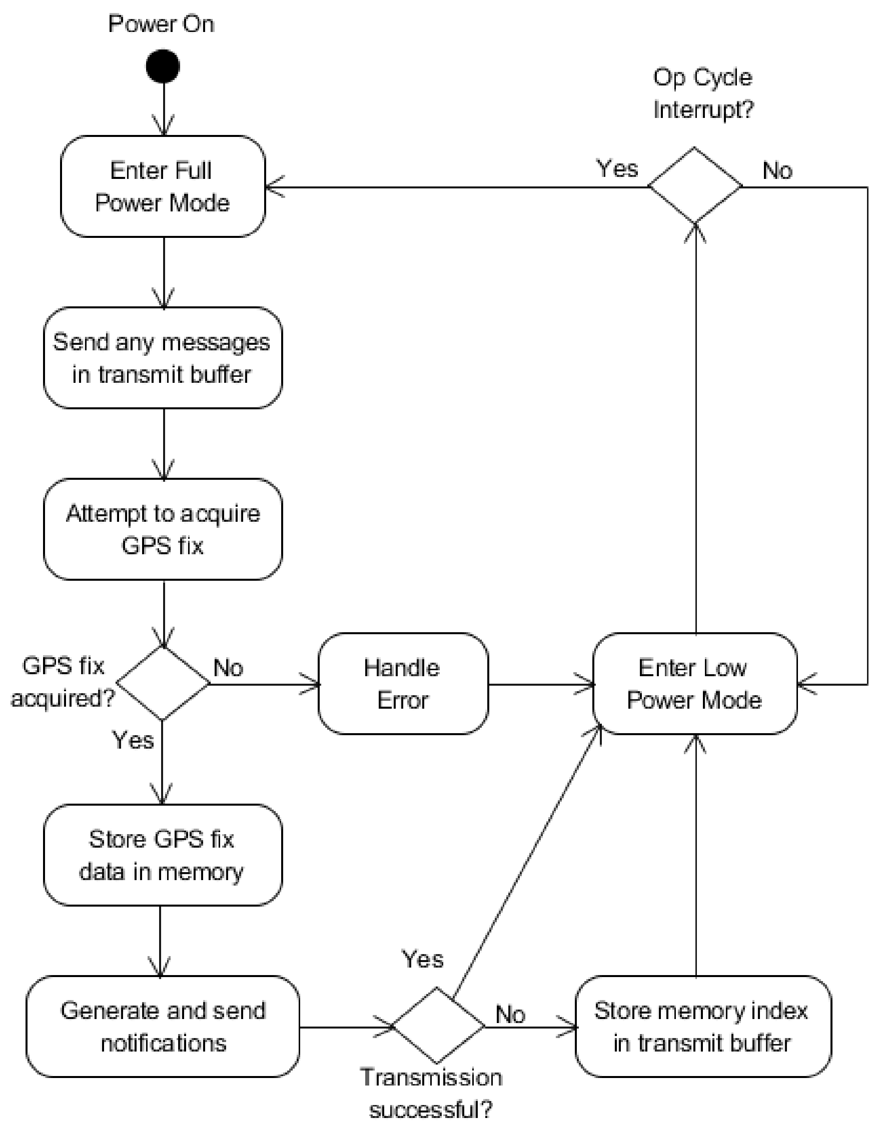

Figure 6-2. Top Level Activity Diagram

6.2 Hardware Summary

6.2.1 Electrical Schematic Diagrams

The following diagrams show the electrical schematics for power source management via charger, charge controller, battery, input diagram for on/off power, phototransistor, barometric pressure sensor, output diagram for network status, GPS status, and microcontroller status. It also shows serial communication between the microcontroller and the celular module as well as microcontroller and memory.

6.2.1.1 USB Micro-B and Charge Manager Schematic

Figure 6-3 shows the electrical schematic for connections to the power source. A zener diode labeled D1 was used in order to protect the load from incorrect reverse voltages to the input as well as to protect against short circuits. Also a 0.1µF has been included in order to satisfy any transient high current demands from the IC modules. A charge manager was included in order to satisfy correct charging requirements for lithium ion batteries.

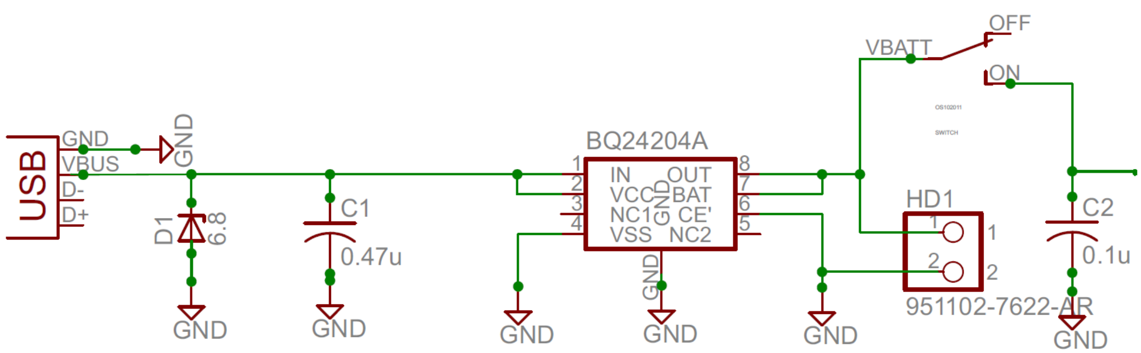

Figure 6-3. USB Micro-B and Charge Manager Schematic Diagram

6.2.1.2 Battery Connection

Figure 6-3 shows the electrical schematic for connections to the battery. The battery is not shown on the schematic, but means for a connection are provided in the form of a header labeled HD1. Since Ni/Cd or Ni/MH battery types can be over-charged, resulting in the voltage surpassing the allowed maximum for the GM862, a lithium ion rechargeable battery was selected in order to guarantee voltage stability of 3.7 V. Also, a 900mAh battery capacity was selected in order to satisfy any potential surge requirements of the GM862 module when simultaneously handling multiple processes (i.e. GPS, GSM, logging, etc).

6.2.1.3 GM862-GPS and Microcontroller Schematic

The figure below shows input signals for turning the GM862 on and off as well as the UART Serial Communication lines between the GM862-GPS and the MSP430 microcontroller.

The On/Off pin 17 is internally pulled up, so it operates when forcing a logical low signal. For the on/off operation the low level signal must be maintained more than 1 second. This is achieved by output 2.0, pin 8 of the MSP430 connected to an inverter via an NPN transistor. This provides a low signal with low current draw, when the microcontroller output is set to logical high.

Universal Serial Communication Interface (USCI) between the microcontroller and the GM862 cellular module was achieved by a two wire connection RXD and TXD. The implemented protocol is based on RS232 with one transmit line, one receive line, 8 bit data, one stop bit, and no parity bit. Because the UART protocol is used, there is no flow control.

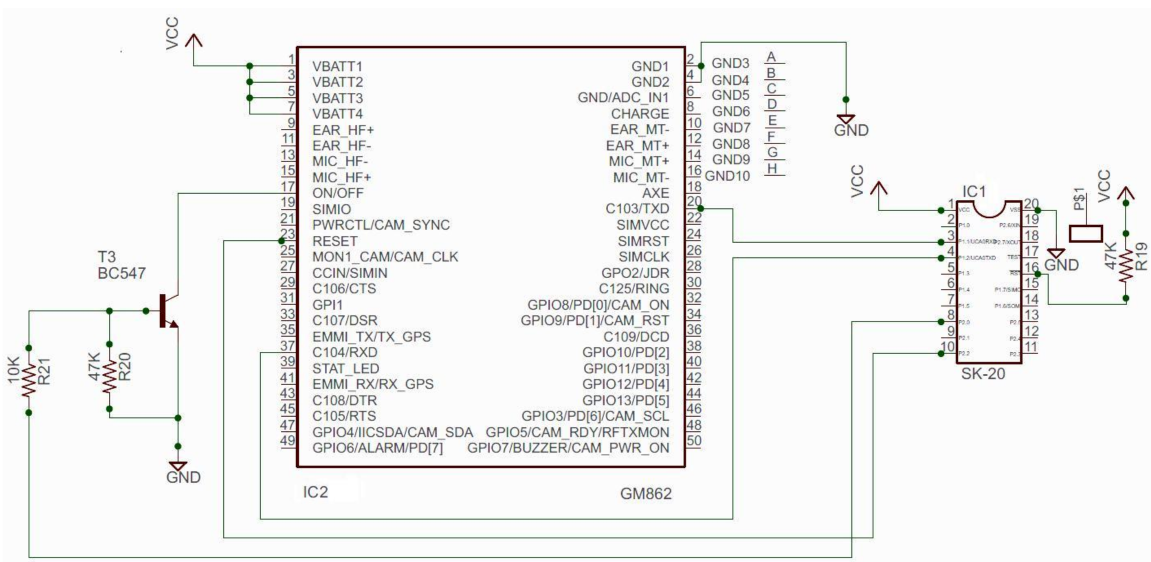

Figure 6-4. GM862-GPS and Microcontroller Schematic Diagram

6.2.1.4 Output Signals. LED Schematic Diagram

Figure 6-5 shows the output signals for network status, with the green LED labeled LED4 connected to pin 39 STAT_LED of the GM862, GPS status with a yellow LED labeled LED5 connected to pin 28 GPO2 of the GM862, and microcontroller status with a red LED labeled LED6 connected to pin 11 Output 2.3 of the MSP430. The LEDs are connected in series with 470Ω resistors. Since these output pins are open collectors, and the output voltage is the inverse of the signal status, the connection is pulled up to VBATT 3.7 V.

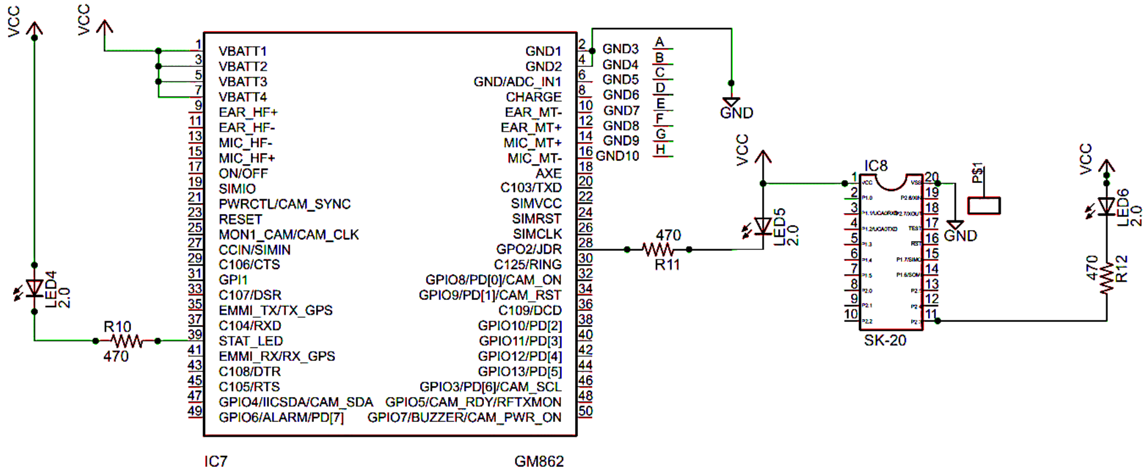

Figure 6-5. LED Schematic Diagram

6.2.1.5 Microcontroller and Memory SPI Serial Communication

The MSP430 microcontroller is able to request and collect GPS data from the GM862, add a time stamp, and log the data into the flash memory via universal serial communication interface using SPI. This serial bus is shown in Figure 6-6. The microcontroller will function as a Master for all communications. The flash memory is set as slave. During transmission through the SPI serial lines, the Chip Select (CS) line is enabled or disabled by the microcontroller output 2.4 pin 12. Data that needs to be saved is transmitted by microcontroller's output 1.7 pin 15 to the flash memory Slave In (SI) pin 1. To retrieve data, the microcontroller's Input 1.6 pin 14 receives from flash memory Slave Out (SO) pin 8. It is important to highlight that the data rate for communication to the flash memory has to be less than 66 MHz maximum clock frequency. Input SCK pin 2 of the flash memory is connected to output 1.5 pin 7 of the microcontroller. The clock is generated and controlled by the microcontroller's software.

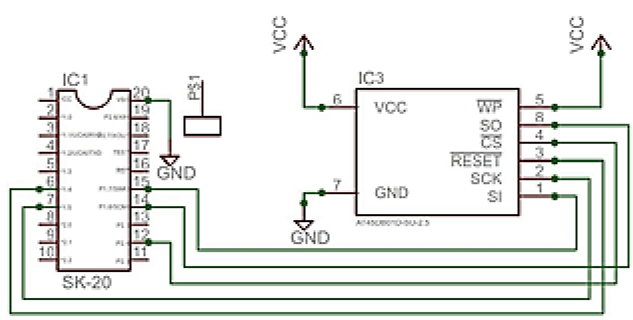

Figure 6-6. Microcontroller and Memory SPI Schematic Diagram

6.2.1.6 Barometric Pressure Sensor Schematic Diagram

Figure 6-7 shows the schematic diagram for the barometric pressure sensor. The IC is connected to the input pin of the microcontroller. Pin 2 of the microcontroller is set to read analog voltage. The microcontroller uses the Analog to Digital converter to generate a digital value for all calculations. The output pin 4 of the pressure sensor also has a capacitor and a resistor that functions as a low bypass filter thereby avoiding any high frequency transients.

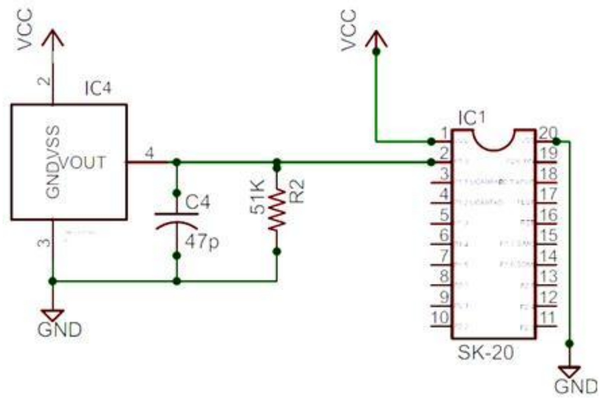

Figure 6-7. Pressure Sensor Schematic Diagram

6.2.1.7 Ambient Light Sensor Schematic Diagram

Figure 6-8 shows the schematic diagram for the ambient light detection. A PNP phototransistor is connected to the input pin of the microcontroller. The gain has been calculated by adjusting resistor R8 to 4.1KΩ. When no light is present, no current is flowing through the transistor which, keeps pin 5 of the microcontroller at a logical zero (low). When ambient light is present, current passes through the transistor, and once the voltage threshold of 2.2V is surpassed across resistor R8, a logical high is read by the microcontroller. It is important to note that pin 5 of the microcontroller must be configured as an input pin.

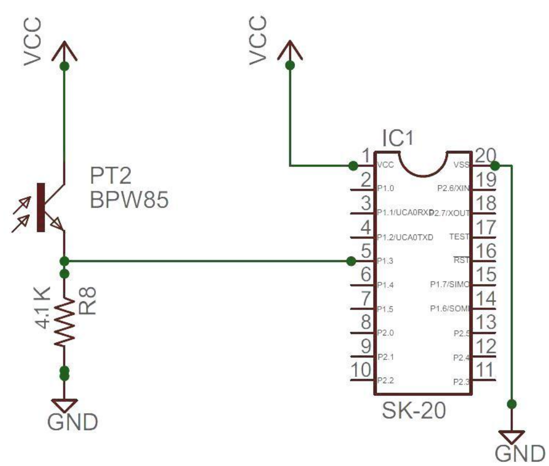

Figure 6-8. Phototransistor Schematic Diagram

6.3 Software Summary

The software used to control the Luggage Tracking Unit consists of multiple software modules carefully integrated to allow direct communication between the components. Because of the complexity of the software for the Luggage Tracking Unit, it can be better understood when broken down into specific modules. At the top level, the overall functionality of the LTU's software depends on the proper functionality of its underlying software modules.

The on-chip software is designed to directly control the communication between the microcontroller and the other LTU components. The on-chip software receives and process data from the GPS module. The GPS coordinates are then processed into a text message and transmitted to the user via the GSM module. Thus, the on-chip software's main objective is to successfully transmit information concerning the LTU's current position while also maintaining communication lines between the multiple components within the LTU.

To better understand the high-level functionality of the software used to control the LTU, a class diagram is provided below:

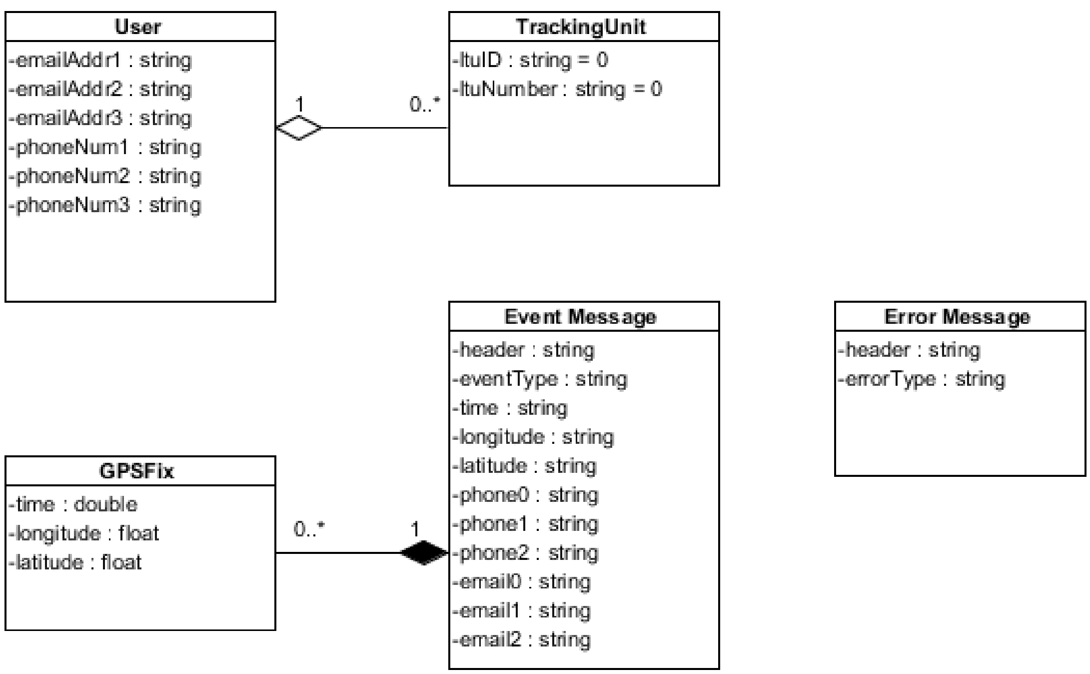

Figure 6-9. LTU Class Diagram

The LTU class diagram contains four basic classes: User, TrackingUnit, GPSFix, and StatusMessage. The User class is responsible for storing and verifying the user's information. Because each user can have many LTUs, there is a one-to-many relationship between the User class and the TrackingUnit class. The

TrackingUnit class is responsible for storing the identification numbers associated with each individual LTU. The GPSFix class contains the information associated with each LTU's current location, while the StatusMessage class is responsible for formatting each alert in the form of an email or SMS text message. There is a one-to-many relationship between the StatusMessage class and the GPSFix class because there can be multiple sets of GPS coordinates associated with each status message.

The User class is used to store the basic information of the user in a single User object. There are three fields for the user's email addresses and phone numbers. These fields allow the user to be verified in multiple ways, using numerous email addresses and phone numbers as notification destinations.

The TrackingUnit class assigns each LTU a unique set of identification numbers. An exclusive tid (tracking unit identification) will be assigned to each LTU and used to identify it.

The GPSFix class contains all the necessary information to correctly identify the location of an LTU at a specified time. The time parameter is used to log the time once a request is sent to determine the location of an LTU. The longitude and latitude fields are then be populated with the correct GPS coordinate information based on the LTU's current position. The objects created by this class are associated with the alert and status messages sent to the user to provide the current location(s) of their LTU(s).

The StatusMessage class will be used as a blueprint to format each alert message. Once the alert has been completely formatted, it will be processed and sent as an email and/or SMS text message to the destinations specified by the user, or sent to the GPS server for distribution. The header serves as a title field in text format. The time field contains the time at which the alert message was processed and sent. The longitude and latitude fields will contain the current GPS coordinates of the LTU at the time the alert message was sent. The body of the alert message will contain important information regarding the LTU in text format. The three emails and three phone numbers are the destination(s) of each alert message, while the tid (tracking unit identification) field notifies the user as to which LTU the alert message is referring.

Error messages will contain the type of error that occurred. It is important to note that, while error messages are stored in memory, not all error messages are attempted to be sent to the user. For example, if a communication test error occurs or a message send error occurs, the error message is not attempted to be sent to the user. If the error message was attempted to be sent, that attempt would naturally fail as well, so it is not necessary to even try to send a message in that case.

In addition, no error handling is performed if an error message fails to send. The reason for this is that an infinite loop could essentially be created because error handling would be performed on error handling. So, for example, if a GPS fix attempt failed to send, an error message would be sent to the user. If this error message failed to send, then yet another error message would be generated, and it too would be attempted to be sent to the user. If, for some reason, the GSM module was malfunctioning or some other issue was occurring which prevented the GSM module from sending messages, this error handling cycle would repeat itself indefinitely.

The following table lists all potential error events that may occur during operation of the LTU:

Table 6-1. LTU Error Summary Part 1

Error Type	Error Details
Configure Clocks	This error is reported if calibration data for the internal microcontroller clocks has been erased.
Communication Test	This error is reported if three consecutive communication tests fail. Communication tests are run at the beginning of every operation cycle to verify communication between the microcontroller and GM862. If this error is reported, there is no verifiable communication between the microcontroller and GM862, so no further operations are attempted until communication has been reestablished.
GPS Acquire	This error is reported if there are three consecutive failed attempts to acquire a GPS fix. When this error occurs, a notification is attempted to be sent to the user alerting them of this error.
GPS Low Power	This error is reported if there are three consecutive failed attempts to command the GPS module into a low power mode. If this error occurs, a notification is sent to the user alerting them of the error, but the LTU operation continues as usual. This error means that the LTU's battery life will be significantly reduced because the GPS module is constantly operating in full power mode.
GSM Notify	This error is reported if there are three consecutive failed attempts to send all messages waiting in the LTU's transmit buffer. If this error occurs, a notification is sent to the user alerting them of the error, but the LTU operation continues as normal.

Table 6-2. LTU Error Summary Part 2

Error Type	Error Details
GSM Low Power	This error is reported if there are three consecutive failed attempts to command the GSM module into a low power mode. If this error occurs, a notification is sent to the user alerting them of the error, but the LTU operation continues as usual. This error means that the LTU's battery life will be significantly reduced because the GSM module is constantly operating in full power mode.
GSM Off	This error is reported if there are three consecutive failed attempts to turn off the GM862 module. If this error occurs, a notification is sent to the user alerting them of the error, but the LTU operation continues as usual.
Message Save	This error is reported if there are three consecutive failed attempts to save an SMS message to the SIM card memory. If this error occurs, a notification is sent to the user alerting them of the error, but the LTU operation continues as usual. This error means that the message has been lost because it could not be saved to the SIM card for later transmission.
Message Send	This error is reported if there are three consecutive failed attempts to send an SMS message from the GSM module. If this error occurs, the error is logged in memory, but a notification is not sent to the user because it too would fail to send. Instead, the memory index of the message is saved in a transmit buffer to be sent during the next operation cycle. After the error has been logged, the LTU operation continues as usual.

7 Prototype Construction

7.1 Bill of Materials

Line #	Item	Qty	Description	Manufacturer	Vendor	Part Number	Unit Cost	Lead Time
1	IC1	1	MSP430 ultra-low-power 16-bit microcontroller	TI	Sample	MSP430G2553	-	In stock
2	IC2	1	GPS receiver and GPRS digital communication	Telit	Semiconductor Store.com	GM862-GPS V.00	$120.66	In stock
3	AN1	1	GSM ANTENNA, MMCX	Taoglas	Digi-Key	FXP07.09.0100A	$14.82	In stock
4	AN2	1	GPS ANTENNA, MMCX	EAD	Digi-Key	GPS3620	$16.96	In stock
5	IC3	1	Memory. Flash 4M 8 I/O Pin RapidS SPI 264B 2.5V 2.7V	Atmel	MOUSER	AT45DB041D-SU	$0.92	In stock
6	IC4	1	High Accuracy, Absolute Pressure Sensor IC	Freescale Semiconductor	MOUSER	MP3H6115A	$10.08	In stock
7	PT2	1	Silicon NPN Phototransistor	Vishay	MOUSER	BPW85A	$10.08	In stock
8	BQ24	1	Li-Ion charge management IC	TI	Sample	BQ24024	-	In stock
9	BATT	1	BATTERY Li-Ion, 3.7V, 900mAh	Ultralife	MOUSER	UBP002	$10.08	In stock
10	LED4	1	Green T-1 3mm LED	Lumex	MOUSER	SSL-LX3044GC	$0.15	In stock
11	LED5	1	Red, T-1 3mm LED	Lumex	MOUSER	SSL-LX3044IC	$0.15	In stock
12	LED6	1	Yellow T-1 3mm LED	Lumex	MOUSER	SSL-LX3044YC	$0.15	In stock
13	CON1	1	BOARD TO BOARD CONNECTOR 50 PIN	MOLEX	MOUSER	501920-5001	$6.64	In stock
14	SK20	1	20-pin IC Socket	Harwin	MOUSER	D2820-42	$1.50	In stock
15	USB	1	USB Micro-B RECEPTACLE	TE Connectivity	MOUSER	1981568	$1.50	In stock
16	HD1	1	2P HEATHER, 90 DEGREE BATTERY CONNECTOR	MOLEX	MOLEX	22-28-9020	$0.95	In stock
17	-	2	PRESSURE CRIMP TERMINAL	MOLEX	MOLEX	46999-0101	$0.08	In stock
18	-	1	CRIMP TERMINAL HOUSING	MOLEX	MOLEX	10-11-2023	$0.15	In stock
19	CASE	1	ENCLOSURE BOX	PacTec	MOUSER	71100-510-000	$3.84	In stock
20	C1	1	0.47µF Tantalum Capacitor, 35V	VISHAY	MOUSER	293D104X9035A2TE3	$0.67	In stock
21	C2	2	0.1µF Tantalum Capacitor, 10V	VISHAY	MOUSER	293D107X9010C2TE3	$1.06	In stock
22	C3	1	47pF Tantalum Capacitor, 10V	VISHAY	MOUSER	293D102X9010C2TE3	$0.54	In stock
23	D1	1	Zener Diode 500mW, 6.2V	DIODES INC.	MOUSER	PD3Z284C6V8-7	$0.50	In stock
24	R1, R2, R3	3	470Ω 5% Resistor, 1/4W	VISHAY/DALE	MOUSER	CRCW060347K0JNEAHP	$0.19	In stock
25	R8	1	4.1KΩ 1% Resistor, 1/4W	VISHAY/DALE	MOUSER	CRCW120341R5FKEA	$0.10	In stock
26	R9	1	51KΩ 1% Resistor, 1/4W	VISHAY/DALE	MOUSER	CRCW120351K5FKEA	$0.10	In stock
27	R21	1	10KΩ 5% Resistor, 1/4W	VISHAY/DALE	MOUSER	CRCW060310R0JNEAHP	$0.19	In stock
28	R19, R20	2	47KΩ 5% Resistor, 1/4W	VISHAY/DALE	MOUSER	CRCW0603470RJNEAHP	$0.19	In stock
29	T3	1	NPN BJT, GAINCB=200, 2ma, 5V	DIODES INC.	MOUSER	BC547F	$0.40	In stock
30	PCB	1	4”x4” PCB Board	Advanced Circuits	Advanced Circuits	Custom Order	$33	Lead 7 Days
31	EVAL KIT	1	GM862 Cellular Evaluation Board - USB	Telit	Sparkfun	RB-Spa-308	$84.71	In stock
32	DEV KIT	1	TI Development Kit for MSP430	TI	TI	MSP-EXP430G2	$4.30	In stock
33	-	1	Monthly Cellular Service	AT&T	AT&T		$16	In stock

7.2 PCB Design

Figure 7-1 shows the complete electrical schematic for the LTU. CadSoft EAGLE PCB software was used for the design of the PCB.

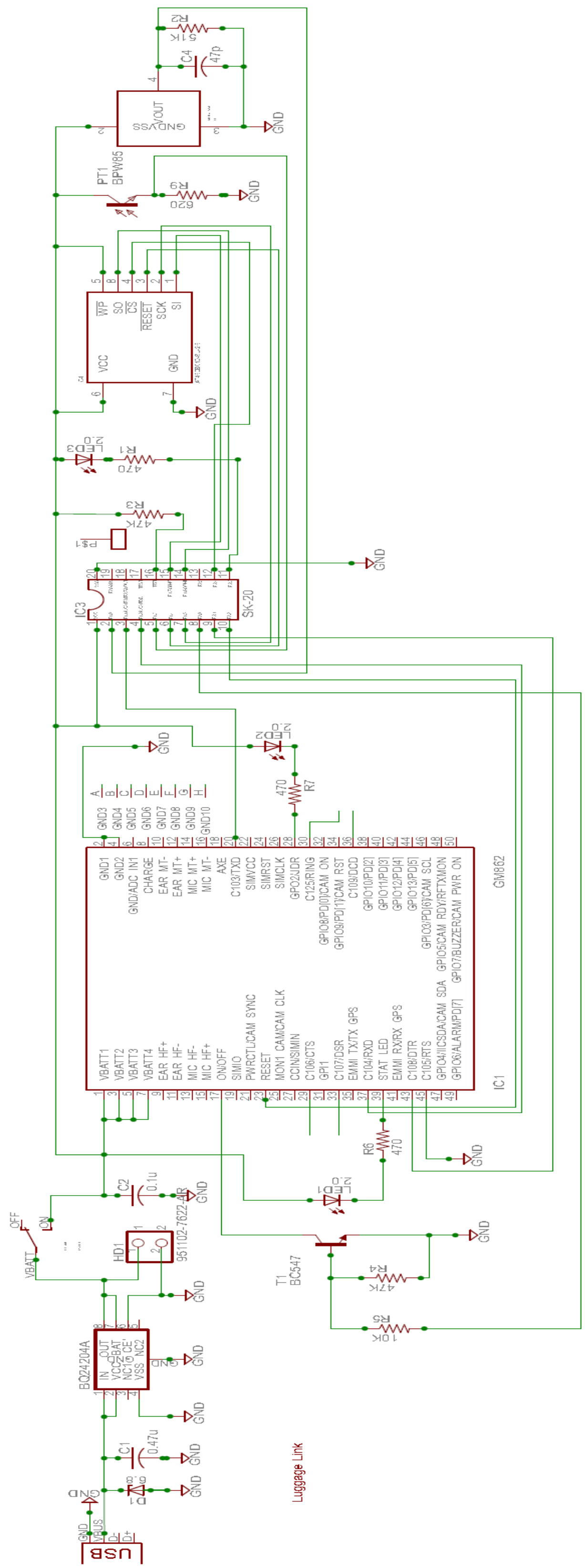

Figure 7-1. LTU Schematic in CadSoft EAGLE PCB software

Figure 7-2 shows a view of the layout generated by EagleCAD software. Red lines are the top traces, blue lines represent the bottom traces, vias are represented in green, and components dimensions and names (silk screen) are in grey.

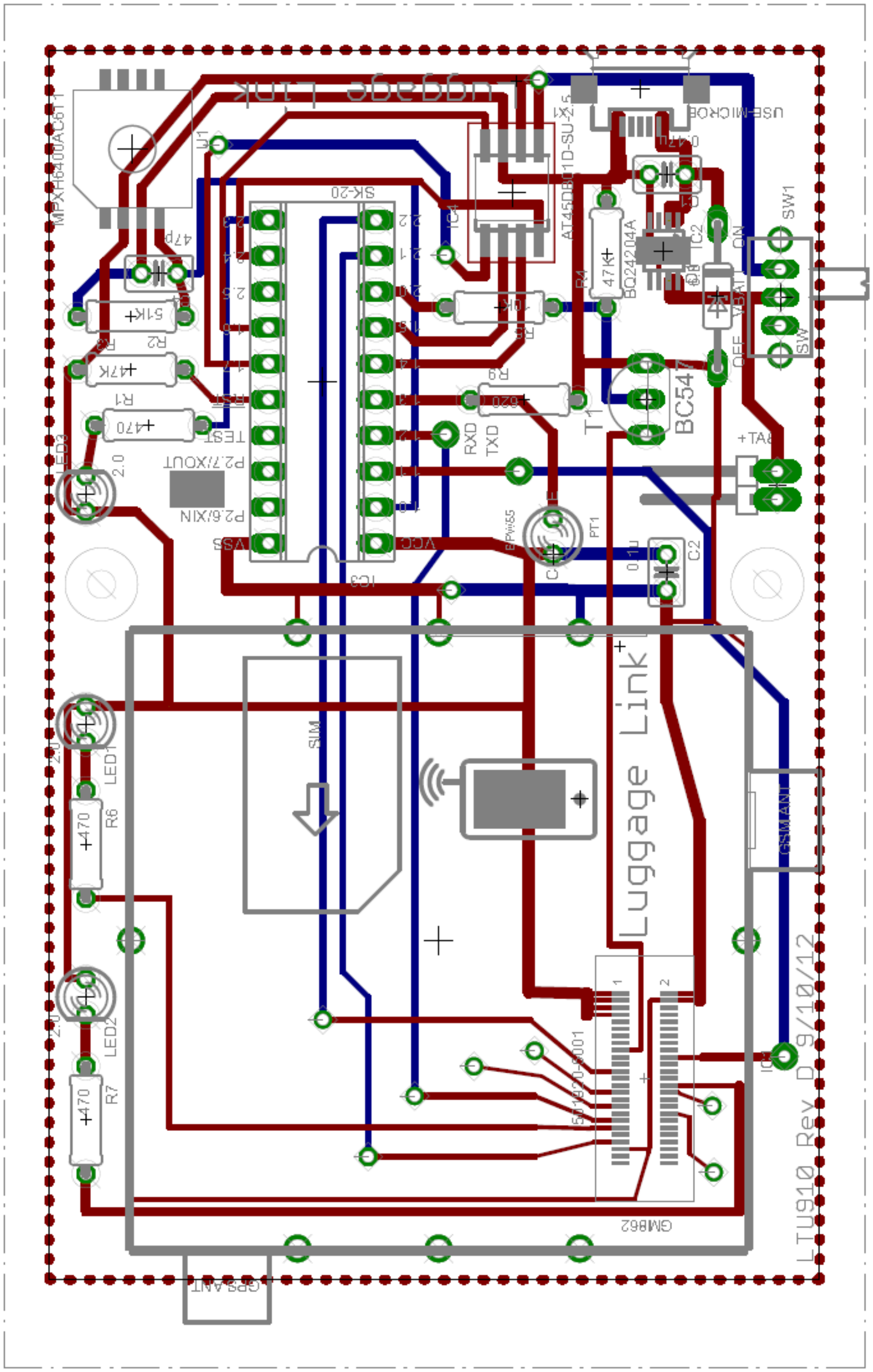

Figure 7-2. LTU Layout in CadSoft EAGLE PCB software

7.3 Procurement

7.3.1 Hardware

7.3.1.1 Prototype Components

The majority of the components for this project were purchased from vendors. Some of the vendors that we utilized are as follows: Mouser Electronics, Digi-Key, Motorola, Sparkfun, and Semiconductor Store. These vendors are distributors for the major manufacturers of electronic components.

7.3.1.2 PCB Manufacture

The PCB was ordered through a board house called Advanced Circuits located in Aurora, CO and Tempe, AZ. Since the budget has been set for two LTU, the first unit or PCB was ordered as soon as the design phase was complete. The second unit was ordered after basic testing was completed on the first PCB. This helped reveal any flaws that were not foreseen during the design phase. See the figure below for a picture of the PCB before any components were mounted on it.

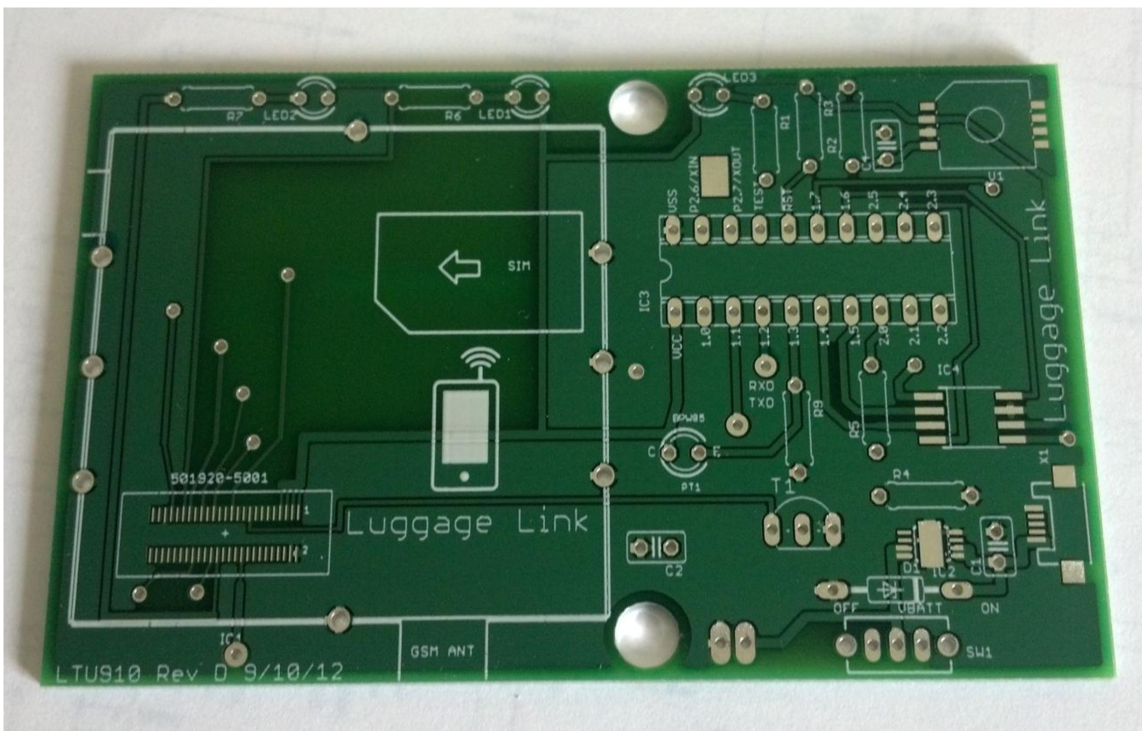

Figure 7-3. LTU PCB

The standard lead time for an order of a 2-layer design PCB with less than 60 sq. inches is about 7 days, not including shipping time. The total waiting time after uploading the GERBER files was 16 days.

7.3.2 Software

7.3.2.1 PCB Design Software

Two possible PCB layout software platforms were identified in order to generate the PCB Gerber files for our project. The first option was a free software package provided by the board house, Advanced Circuits. The PCB Artist software can be downloaded on any computer running Microsoft Windows® OS. This software package includes a library of over 270,000 parts and auto-routing tool, which selects the most effective route to interconnect terminals and components. These connections are made via tracks which are segments of a conductor. Free software provided by Advanced Circuits is called design file check (DFC). Once the Gerber files are generated, a DFC can be done in order to identify any potential problems related to physical connections of the components and files issues.

The second PCB layout software is called CadSoft EAGLE PCB. It is provided by CadSoft Computer. The EAGLE PCB software offers a schematic editor, a layout editor and autorouter. The software also provides a library editor that allows for creation or custom adjustment of components. Another feature is the ability to create a bill of material and check pricing and availability of components in addition to the automatic creation of a bill of material. Moreover, an order can be completed entirely through the selected board house, so there is no need to purchase items before sending them to the board house.

After considering the two PCB software options, it was decided that CadSoft EAGLE PCB would be used for this project due to its simple user interface and ease of ordering a PCB through the program itself.

7.3.2.2 On-Chip Software

The on-chip software was developed in the C programming language, which is supported by the MSP430 microcontroller module. Texas Instruments, who manufactures the MSP430, provides a free software developer environment called Code Composer Studio that is preconfigured to interface with the Launch Pad. The Launch Pad is the evaluation board for the MSP430 product line and was used extensively throughout this project to program the MSP430 and test its functionality. The Code Composer Studio software was obtained at no cost through the Texas Instruments website.

7.4 Prototype Assembly

Our initial budget did not include assembly cost, so we decided to assemble the PCB ourselves. Soldering and point-to-point testing for a single LTU took about 10 hours. The most difficult items to solder were the SMD chips and the 50-pin board-to-board connector. The charger manager IC and the flash memory IC were soldered first in order to have better access for the tip of the soldering iron to the pins. The 50-pin connector has the pitch of the pins very close together and, moreover, they were short. This made it very difficult for the tip of the soldering iron to heat individual pins, and many times the solder would cross over and fill other pins as well. Removing this solder also proved to be a challenge.

Another issue was found due to removal of the GM862. Wear and tear weakened the connectors and did not allowed the GM862 50-pin male to fully connect to the 50-pin male. Also, the forces involved in removing the GM862 usually broke the tiny solder points between the pin and the PCB. Figure 7-4 shows all the components soldered to the PCB. The 20 pin socket and the 50-pin connector can be seen since the MSP430 and the GM862 are removed.

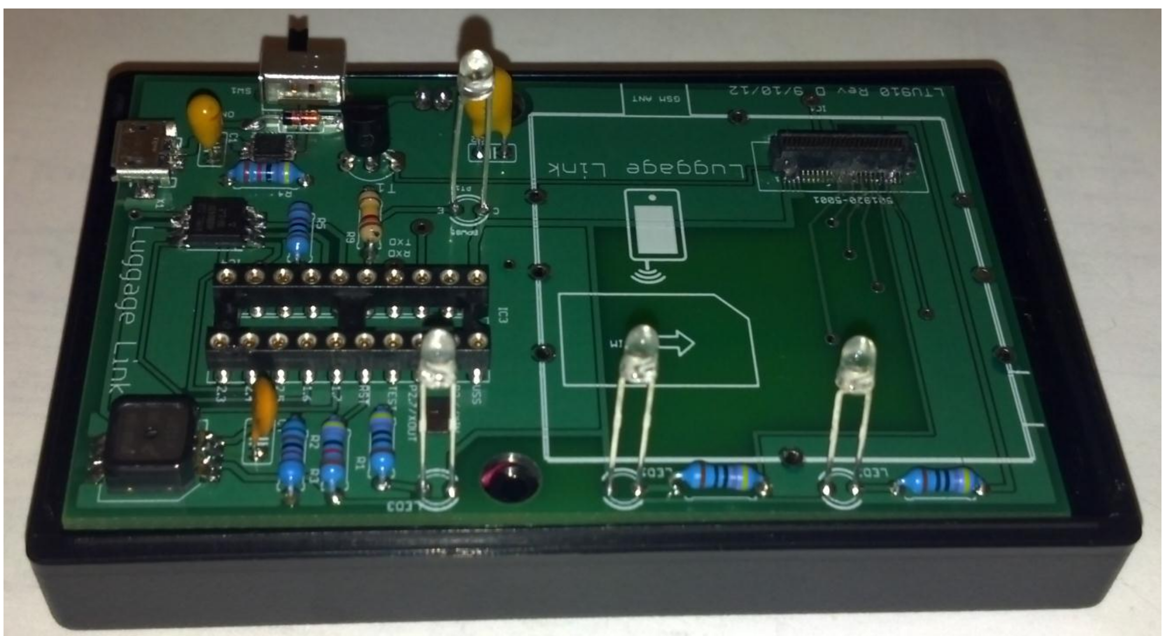

Figure 7-4. PCB Partial Assembly

Figure 7-5 shows the PCB with the GM862 mounted to the board. The white mini coax cable on the top right of the picture is the GSM antenna. This antenna can be folded over the components just before covering the PCB with the top half of the enclosure.

Figure 7-5. PCB including GM862

The last items to be installed were the GPS antenna, the GSM antenna and the battery. Since the GSM antenna was thin, it remained on the top. Figure 7-6 shows how the black mini coax cable was routed towards the bottom. In order to do this, the PCB was designed shorter in length, leaving a gap between the end of the PCB and the side of the box.

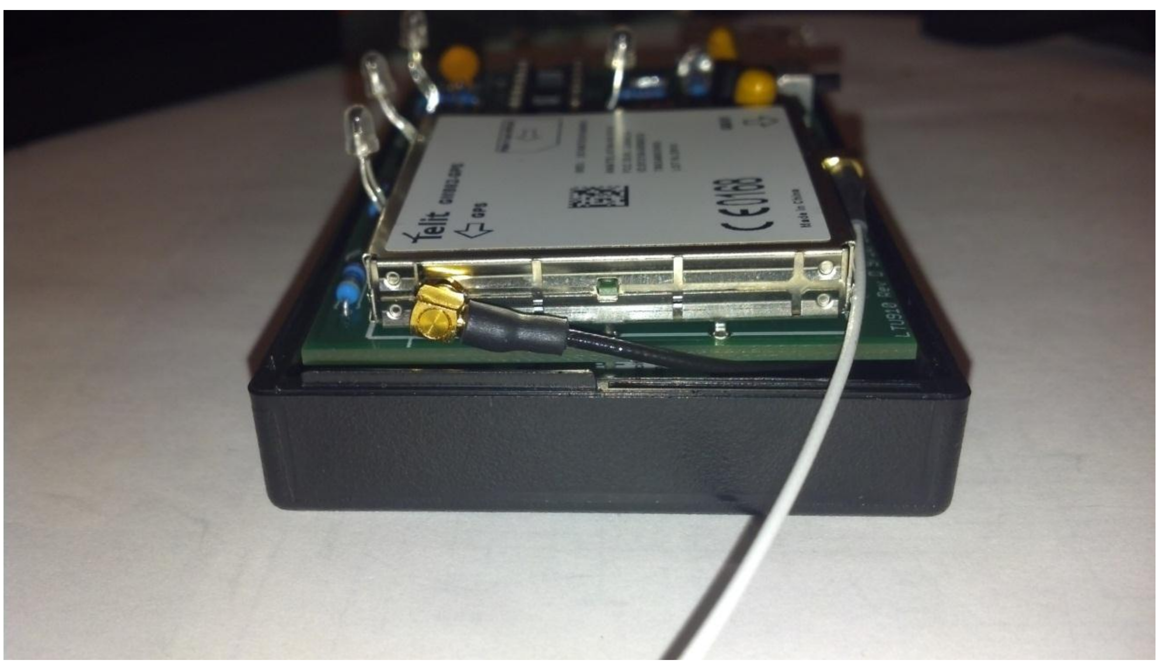

Figure 7-6. GSM, GPS antennae routing

The GPS antenna was routed under the PCB, so it will share the space with the battery. Figure 7-7 shows the bottom view of the PCB with the GPS antenna and the battery.

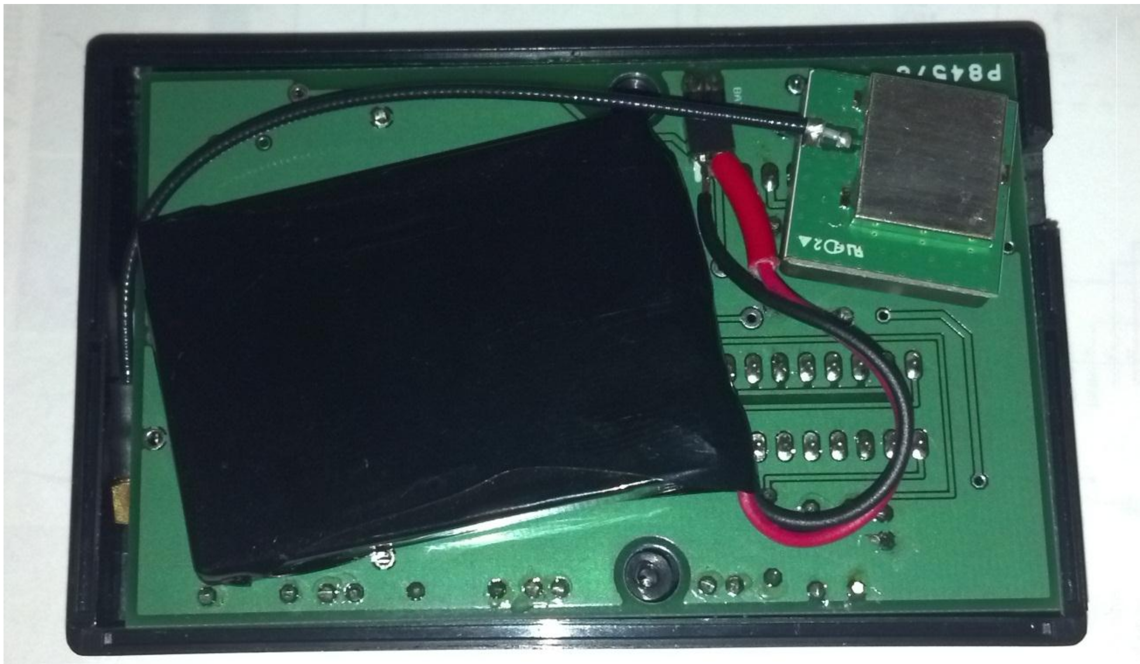

Figure 7-7. PCB bottom view, GPS antenna and battery

Additional time of about 3 hours was required to drill holes and perform modifications to the enclosure. These holes were for the LEDs, phototransistor, USB power connector, and On/Off switch tab. See Figure 7-8 for a detailed view.

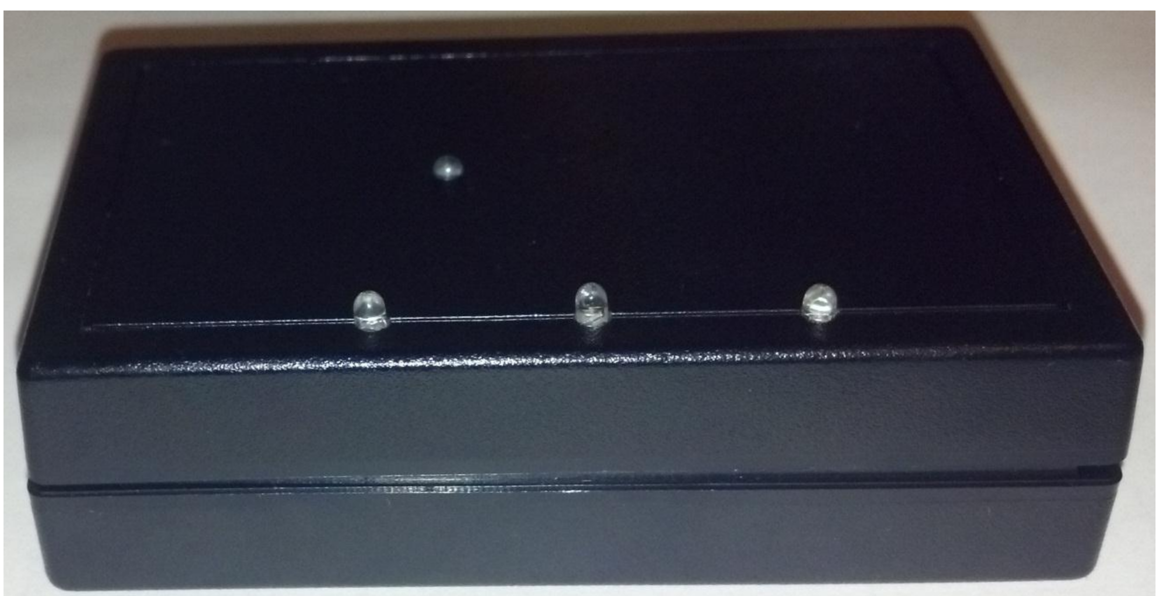

Figure 7-8. PCB inside enclosure box

8 Testing

8.1 Master Test Plan

The master test plan ensured that the system was functioning properly and efficiently, as originally designed. The ultimate goal of the testing was to eliminate all errors from the system. More specifically, errors that stemmed from the integration of the hardware and software components were exposed in the master test plan. From the perspective of low-level testing, the hardware and software test plans sought to expose the errors within their respective regions and the modules associated with those regions. The low-level functionality of the system was tested with more precise tests geared toward specific modules. However, the master test plan focused on the high-level functionality of the system, rather than specific, low-lying problems.

8.1.1 Simulation Test Environment

In order to avoid as many issues as possible in the build and prototype phase of development, we planned to simulate as many aspects of our design in a software simulation program before actually implementing them. Simulation provides the benefit of being able to experiment with an idea or design before spending the time, energy, and money to actually integrate that component into the product. This application of simulation applies to both hardware and software as individual components, and can even be applied to hardware and software simultaneously.

A complete hardware/software co-design and simulation was planned to be implemented using SystemC. SystemC is a C++ based software that provides the ability to model a system's hardware and software components, as well as their interfaces, all in one software program. This program was to rigorously test the general functionality of the system and verify the basic functionality of the unit, excluding the influence of environmental factors. The simulation test environment was to establish benchmarks to help direct and define the real-world test environment. Once the simulation testing had been completed, the physical prototype would undergo much of the same testing in a real-world environment. Due to time constraints, we were not able to implement any simulations in SystemC for this project.

8.1.2 Real-World Test Environment

In order to yield realistic outcomes when testing the LTU, the test environment was setup so that it resembled a real-world setting in which the system would operate. First, the basic functionality of the device was tested to ensure the device was working properly under ideal conditions. If the results from the real-world test environment resembled those found in the simulated test environment, further testing would commence to test the robustness of the device under non-ideal conditions.

For this project, there were no set limitations or constraints put in place regarding the potential destinations or locations of the unit. Therefore, it is assumed that the LTU will experience many environmental changes throughout its various journeys. The environmental changes can be, and will likely be, erratic. The device will continually experience rapid changes in altitude while in flight. The unit may also experience rapid changes in temperature and humidity when traveling internationally or to remote locations.

Since the environments in which the device will be taken are somewhat unpredictable, general environmental factors were taken into consideration when defining the test environments. That is, the device was tested by exposing it to significant changes in pressure, humidity, temperature, and many other environmental factors to determine whether any of these conditions negatively affect the operation of the unit. The test environments were also setup, to the best possible extent, to verify the environmental limits on the LTU. The environmental factors that can be controlled were the focus when defining test case environments, while the uncontrollable factors were considered but not altered. The defined testing environments sought to test each environmental factor individually as well as holistically. It should be noted that when forming the test environments, the specified limits of the modules were not deliberately exceeded in order to ensure the safety of the prototype.

8.1.2.1 Testing Environmental Factors Individually

When testing individual environmental factors, only one environmental variable was typically allowed to fluctuate. All other aspects were to be controlled in such a way that they experienced no significant or erratic changes. For example, the device was tested with varying amounts of air pressure while all other factors remained controlled and unchanged. Once the data was collected on the responsiveness and functionality of the system under different air pressures, the air pressure was fixed, along with all other controllable factors, and another individual factor was tested. This process of individually testing each environmental factor was continued until all controllable aspects had been thoroughly tested and unambiguous data had been collected.

8.1.2.2 Testing Environmental Factors Holistically

Once the individual environmental factors had been tested, a series of holistic tests were initiated. This testing environment was to test multiple factors in one, universal test environment. This testing environment was to encompass varying changes in a majority of the environmental factors in such a way that it resembled a real-world scenario. This environmental testing helped to form a conclusion on whether the device would successfully operate in extreme conditions with erratic changes in the environment.

8.1.3 Pressure Testing

The most important environmental factor to consider in our system was air pressure. Since the primary use of the Luggage Tracking Unit is on distant travels, the unit will likely spend a significant portion of its operational time on an aircraft. Thus, the system needed to be able to undergo erratic and significant changes in air pressure and still function properly. The system also needed to be able to handle the pressure changes that come with variations in altitude. Variations in pressure can cause the physical components of the unit to respond differently, resulting in invalid operation.

Testing variations in air pressure in a real-world scenario proved to be a difficult challenge. Most means of getting to a high altitude are considerably expensive and time consuming. Thus, being able to test the system at a typical operating altitude in order to simulate a decrease in air pressure was not feasible. To bypass this roadblock, a method of simulating changes in air pressure was developed and implemented. The barometric pressure sensor that was chosen for the LTU has a small, vertical nozzle on its top face. To simulate changes in air pressure, we attached a small plastic hose to the nozzle on the pressure sensor.

By blowing into the hose or sucking on the hose, we were able to easily simulate both increases and decreases in air pressure. More importantly, we were able to verify that the LTU was responding properly to these changes in air pressure. When air was sucked through the hose, this simulated a decrease in air pressure. Similarly, when air was blown into the hose, this simulated an increase in air pressure. The figure below shows the LTU with a hose connected to the pressure sensor.

Figure 8-1. Pressure Sensor Testing Setup

The algorithm we implemented in the microcontroller software constantly checked this air pressure, and if a certain number of changes in air pressure were detected within a specific period of time, an interrupt would be triggered and the LTU would respond accordingly. If a sufficient number of decreases in pressure were detected, the LTU would send a notification to the user alerting them that a pressure decrease had been detected and that their luggage was on an aircraft. After sending this notification to all of the user's specified phone numbers and email addresses, the microcontroller commanded the GSM module to disable all cellular communication. Again, this action was taken in order to comply with the Federal Aviation Administration's regulations against the use of cellular devices while on an aircraft.

Although software simulation could have been used to test the system in general ways, some form of physically testing variations in altitude needed to be performed to properly test the robustness and functionality of the device. Testing and reconfiguring the system in this manner was continued until the system was proven to successfully and correctly respond to variations in air pressure.

8.1.4 SIM Card Assessment

In order to holistically test and demonstrate the Luggage Tracking Unit's full functionality, a valid SIM card was procured to send SMS text messages. An AT&T SIM card was used for this portion of the testing. Before testing, the SIM card was verified to have a valid phone and data plan. This ensured that SMS text messages could be reliably sent using the Luggage Tracking Unit's GSM module. Because multiple LTUs can be used simultaneously to track multiple pieces of luggage, the system was also tested with multiple LTUs, each containing their own SIM card.

8.1.4.1 Verification of Single SIM Card

To ensure overall functionality of the communication aspect of the system, a single LTU, containing an AT&T SIM card, was tested. First, the basic functionality of the system was tested to ensure the GSM module was properly sending and receiving data via the SIM card. Individual test cases were designed to test the transmission of all necessary data required to efficiently communicate with both cellular phones and email addresses. Once these individual test cases had validated the proper functionality of the SIM card in conjunction with the GSM module, an overarching test was conducted to test and verify all aspects of the system that utilized the GSM module and SIM card to send and receive essential data.

8.1.4.2 Verification of Multiple SIM Cards

Once a single LTU was shown to successfully transmit data using an AT&T SIM card, multiple LTUs were tested simultaneously. It should be noted that the SIM cards each had their own cellular plan. That is, although the AT&T SIM cards were on the same cellular plan, the cards had their own unique contact number, and they were able to operate independently of each other. Each LTU contained its own SIM card, and the test cases were designed to specifically address the problem of operating multiple units simultaneously.

Specifically, the communication of each LTU with the user's cell phone and email address was examined. The GPS coordinates were sent to each of the LTUs simultaneously to ensure that communication with each LTU was stable, despite noise contributed by the other. Once each LTU processed the coordinates, the SMS text messages, sent by each of the GSM modules, were verified to contain the correct information regarding the LTU's current GPS coordinates. Once the test had shown that two LTUs worked properly while operating simultaneously, it was assumed that operating more than two LTUs, each with their own individual SIM cards, could also operate simultaneously.

8.1.5 Stopping Criteria

Throughout the testing process, errors were inevitably encountered. A predetermined system for handling those errors was put in place prior to testing. A plan to know when to stop testing and either announce success or go back to development was established. However, it should be noted that the stopping criteria varied across test cases. Thus, the specific test case was evaluated in depth before a stopping criteria was agreed upon by the individuals running the test.

8.1.5.1 Error Discovery

There were two possible actions that could have been taken when errors were encountered during testing:

1. Immediately stop the testing when the error was detected and attempt to fix that problem before continuing.
2. Continue testing and record all of the encountered errors in a sequential manner. If a fatal error was encountered when practicing this method, stop testing and attempt to solve the errors encountered up to that point.

The method used when testing was discussed amongst the group members prior to executing the specified test. Depending on the components and modules being tested, the appropriate testing procedure was chosen. Once the majority of the errors had been detected, a new testing procedure was to be introduced. This testing procedure consisted primarily of testing the LTU for a specified amount of time to ensure proper functionality of all systems under extended periods of time.

When performing time-specific testing, a group meeting was held prior to testing to determine the amount of time to test the basic functionality of the system. Throughout testing, errors were recorded, much like the method described in (2) above. Once the time limit had been reached, testing was halted and a team meeting was held to determine how to address the encountered errors. If the group decided to address the errors immediately before testing, the system was taken out of the testing environment, and diagnoses of the errors was begun. After the errors had been dealt with, testing commenced in the same manner. However, if the testers decided that the errors encountered were not fatal, an additional testing time limit was determined and testing was continued until the new time limit was reached.

8.1.5.2 Error Handling

Once the errors were detected in the system, a solution to properly eliminate the errors was discussed and agreed upon by the group. The approach varied depending on the location and severity of the errors. If the error was found to be in any of the hardware modules, the system was disassembled as necessary and diagnoses were begun to pinpoint the error. An attempt to immediately eliminate the error then ensued. Conversely, if the errors were found to be within any of the software components of the system, the error was dealt with immediately to effectively eliminate it while in the testing environment. If a solution to the error could not be found while in the testing environment, analysis of the suspected software component(s) was continued until the issue was discovered and resolved. After the errors were resolved, testing was continued in order to expose additional errors in the system.

8.1.5.3 Adequate for Delivery

Once the identified errors had been resolved, the system underwent the same testing process once more to ensure no additional errors existed. This verification testing process served to expose the potential errors that may have appeared while attempting to resolve other errors. Throughout the verification testing, detected errors were resolved in the same way as mentioned in the previous error handling section. However, unlike the other testing methods, once an error was resolved in the verification test, the entire testing procedure would restart. This process ensured that while eliminating detected errors, other errors were not appearing, and if they did, they would be detected and eliminated immediately.

Once the LTU was put through the verification testing completely with no detected errors, the system underwent one last test. One final, overarching test was conducted to ensure the full functionality of the basic features as well as the additional, nonessential features. The system was analyzed once more, ensuring no errors had occurred that would jeopardize the functionality of the system under any conditions. Once the system had been verified to successfully meet all of these expectations, it was said to be adequate for delivery. Because there was no actual “delivery” for the purposes of this project, the “delivery” of the product refers to the final presentation and demonstration of the project.

8.2 Hardware Test Plan

8.2.1 Overall Objective of Hardware Testing

In order to alleviate troubleshooting and verify functionality of the LTU, several simpler tests and verifications were performed for each of the individual hardware components. These tests and verifications were executed following procedures designed to ensure that all components were isolated from each other and that potential external interference was minimized.

8.2.2 Testing Individual Hardware Components

Each component was tested and verified for proper functionality. In order to accomplish proper isolation of key components, evaluation and development kits were utilized. The biggest challenge was found when testing those components that use serial communication like the GM862, which utilizes serial communication protocol, and the flash memory, which utilizes SPI serial communications protocol. The following sections define the approach for functional testing and verification.

8.2.2.1 GM862-GPS Functional Test Plan

Verification of correct functionality of the GPS was performed by powering the GM862 while mounted to the evaluation kit. With the use of the AT commands, the values of the coordinates in NMEA format were retrieved from the GPS module. The unit was powered on through the evaluation kit, which was connected to either a wall outlet via power adapter or laptop via USB. The coordinates and timestamp were recorded and compared to the unit's actual location to check for accuracy and functionality.

Since the GM862 is in charge of all RF communications via GSM and GPS, it is the largest consumer of power. A test was developed in order to find the best power saving mode. A fully charged battery was used and the microcontroller was programmed to indicate the GM862 to go into trickle power mode. The lapses of time until the battery discharged were recorded. The same procedure was done but in this case the microprocessor was set for push-to-fix mode. tricklepower mode was used in the final LTU product since the battery lasted longer.

8.2.2.2 MSP430 Functional Test Plan

The microcontroller required extensive testing. All of these tests were performed on the MSP430 microcontroller. Most of the tests required the use of the red LED to indicate which parts of the program were functioning correctly and which functions were not. Also, the correct voltages levels for outputs were verified to meet the 2.1 V minimum for high signal level and the 0.5 V maximum for low signal level.

Inputs were simulated using the onboard power provided by the power supply (charger) and the battery. Verification of each digital input was done by checking values in the microcontroller registers while placing direct jumpers into each pin. The usage of internal power guaranteed that voltage and current levels were met when the LTU was in operation.

8.2.2.3 Flash Memory Test Plan

Since the flash memory was connected to the microprocessor via serial communication SPI, a test was developed in order to send and retrieve data utilizing the program in the microcontroller. A series of test algorithms were developed, and with the use of the red LED, we were able to determine if the data was properly saved into memory and properly retrieved.

8.2.2.4 Battery Test Plan

The battery functionality and performance was tested in several areas. Voltage level, current level, charge performance, temperature performance were evaluated. The first verification was to check for voltage output after it was fully charged. A minimum of 3.7 V and a maximum of 4.2 V were found to guarantee correct functionality of all components.

A second test was performed in which the unit was powered on, and the required time for the battery to discharge completely was observed. This was accomplished by activating or powering all of the components including LEDs, barometric pressure sensor, phototransistor, GM862, and MSP430, and observing how long the battery lasted. This test was partially done due to the increased cost of text messages, so the test was performed while testing functionality and debugging the program. At the end, we found that the batteries required to be recharged every 3 weeks, which translates to an average of 126 hours between charges.

A third test was to write a short program that forced all systems functions to work simultaneously (GPS, GSM, memory data logging and retrieval, activation of LEDs and collection of data from barometric sensor and light sensor) and check if there were any failures in the functionality of each component while observing and measuring any temperature change on the battery. This last test was done partially while programing and debugging, and found that none of the components in the LTU were getting too hot to the touch. We concluded that the probability of the battery overheating and igniting was very low.

8.2.2.5 On/Off Switch Test Plan

The switch was tested by using a multi-meter and checking for continuity when moved from the On position to the Off position. This test was performed when power was not present. A second test was performed while the PCB was being supplied with power. With the switch in the On position, we verified that proper voltage was present at each respective Vcc pin on all the components.

8.2.2.6 Barometric Pressure Sensor Test Plan

Since the barometric pressure sensor provides an analog signal in reference to the absolute pressure, the data collected by the microcontroller was used to verify the functionality of the sensor. A small algorithm with a preset threshold was programmed into the microcontroller, and a simulation was performed by blowing through a tube connected to the pressure sensor. When air was sucked from the tube, it a decrease in pressure, and the algorithm activated the red LED, signifying an increase in altitude. When air was blown into the tube, simulating a pressure increase, the algorithm activated the red LED again, signaling a decrease in altitude.

8.2.2.7 Ambient Light Sensor Test Plan

In order to test the ambient light sensor feature, a test was implemented in which the light source was removed from the LTU. Since the phototransistor was directly connected to an input pin in the microcontroller, a small algorithm was developed to activate the red LED if ambient light was removed, triggering a hi/lo interrupt.

8.3 Hardware Test Procedure

The test procedures described below are based on the hardware test plans described in section 8.2. Each subsection focuses on the test procedure for a particular hardware module. The table within each subsection provides the step-by-step procedure that was taken when testing the specified module. A detailed description, along with the conditions under which the test took place, is provided for each step. The expected results were replaced with the observed results for each test.

It should be noted that some test cases closely resembled others. Although some steps within the testing procedures may seem repetitive, the somewhat redundant steps were necessary to properly test specific variables. For example, after performing straightforward tests to ensure basic functionality, the same test was conducted with a variation in conditions. Again, this may seem redundant, but it was important that this factor be tested independently of other variable changes. This form of testing made it easier to pinpoint the errors that arose throughout the testing of that specific module.

8.3.1 GM862-GPS Test Procedure

A description of the step-by-step procedure for testing the GM862-GPS is provided in the table below. The procedure is based on the GM862-GPS test plan described previously.

Table 8-1. GM862-GPS Test Procedure

Step #	Description	Conditions	Result
1	Basic functionality of the GPS was tested using sample inputs to achieve corresponding outputs. This served as a proof-of-concept test.	The test was conducted with the GM862 mounted on the evaluation kit.	The GM862 provided the correct output for each of corresponding inputs on the GPS.
2	The module was powered on at different, random locations to ensure the location was properly identified.	The test was conducted with the GM862 mounted on the evaluation kit.	The module correctly identified its current location.
3	The module was powered on at different, random locations and the current location and timestamp was logged in the data log.	The test was conducted with the GM862 mounted on the evaluation kit.	The module correctly identified its current location and logged the GPS coordinates and timestamp into the data log.
4	The module was powered on at different, random locations to ensure the location was properly identified.	The test was conducted with the GM862 mounted onto a Luggage Tracking Unit.	The module correctly identified its current location.
5	The module was powered on at different, random locations and the current location and timestamp was logged in the data log.	The test was conducted with the GM862 mounted onto a Luggage Tracking Unit.	The module correctly identified its current location and logged the GPS coordinates and timestamp into the data log.

8.3.2 MSP430 Test Procedure

A description of the step-by-step procedure for testing the MSP430 is provided in the table below. The procedure is based on the MSP430 test plan described previously.

Table 8-2. MSP430 Test Procedure

Step #	Description	Conditions	Result
1	The frequency of the MSP430's master clock was tested with an oscilloscope.	The test used an oscilloscope connected to the MSP430 microcontroller.	The frequency of the MSP430's master clock was between 50MHz and 60MHz.
2	The correct output voltage of the MSP430 was analyzed.	The test used a voltmeter connected to the MSP430 microcontroller.	The 2.1 VDC minimum for high signal level was met.
3	The correct output voltage of the MSP430 was analyzed.	The test used a voltmeter connected to the MSP430 microcontroller.	The 0.5 VDC maximum for low signal level was met.
4	The flash memory was analyzed to ensure proper functionality. Inputs was simulated using the onboard power provided by the power supply and the battery.	The test was conducted on the MSP430 microcontroller.	Direct jumpers were placed into each pin of the microcontroller's registers. The inputted values were verified.
5	The serial communication SPI, which connects the microprocessor and the flash memory, was tested by sending and retrieving data from the memory.	The test was conducted on the MSP430 microcontroller.	The microcontroller successfully transmitted data to and from the flash memory.

8.3.3 Flash Memory Test Procedure

A description of the step-by-step procedure for testing the flash memory is provided in the table below. The procedure is based on the flash memory test plan described previously.

Table 8-3. Flash Memory Test Procedure

Step #	Description	Conditions	Result
1	The basic functionality of the flash memory was tested by sending and retrieving data via the serial communication SPI.	The microcontroller and its developing software was used to write and read from the memory module.	The data was successfully written to and read from the memory module.
2	The basic functionality of the flash memory was tested by uploading a series of data values, structured similar to the NMEA protocol, to the flash memory.	The microcontroller and its developing software was used to write and read from the memory module. Two short programs were used: one to write the data to the module and one to read the data from the module.	The data written to the memory module matched the data read from it.
3	The basic functionality of the flash memory was tested by uploading a series of data resembling the log data format used in this system.	The memory module was integrated and mounted to the LTU. The log data was written to and read from the module resembling typical usage.	The data written to the memory module matched the data read from it.

8.3.4 Battery Test Procedure

A description of the step-by-step procedure for testing the battery used in each LTU is provided in the table below. The procedure is based on the battery test plan described previously.

Table 8-4. Battery Test Procedure

Step #	Description	Conditions	Result
1	After the battery was fully charged, the output voltage was checked.	The test was conducted under basic operating conditions of the battery.	The voltage output was 4.2V which meet the requirements of the LTU.
2	After the battery was fully charged, the LTU was left powered on to determine the battery life of the system.	The test was conducted with the battery fully mounted to the LTU and a program was written to continually flash the LTU's LEDs.	The battery power the system for the minimum of five hours. This ensured that without a power-saver mode, the LTU remained on through the duration of an average flight.
3	All of the system's individual components remained at full load to determine the maximum temperature of the battery. This served as a safety test.	The test was conducted with the battery fully mounted to the LTU and in a safe environment in case of extreme temperatures.	The battery did not exceed a safe operating temperature.

8.3.5 On/Off Switch Test Procedure

A description of the step-by-step procedure for testing the On/Off switch is provided in the table below. The procedure is based on the On/Off Switch test plan described previously.

Table 8-5. Push Buttons Test Procedure

Step #	Description	Conditions	Result
1	The switch was activated by sliding to the on position. This test ensured proper physical functionality of switch.	The test was conducted under basic operating conditions of the switch.	The switch stayed in the on position.
2	The switch's continuity was tested.	The test was conducted under basic operating conditions of the switch.	Using a multimeter, the switch's continuity was verified.

8.3.6 Barometric Pressure Sensor Test Procedure

A description of the step-by-step procedure for testing the barometric sensor is provided in the table below. The procedure is based on the barometric sensor test plan described previously.

Table 8-6. Barometric Sensor Test Procedure

Step #	Description	Conditions	Expected Result
1	The basic functionality of the barometric sensor was tested by simulating a decrease in pressure, and therefore an increase in altitude.	The pressure sensor was tested by continually sucking air from the tube connected to it.	The decrement in pressure was analyzed by the algorithm and an indication was made by the red LED.
2	The basic functionality of the barometric sensor was tested by simulating an increase in pressure, and therefore a decrease in altitude.	The pressure sensor was tested by continually blowing air into the tube connected to it.	The increment in pressure was analyzed by the algorithm and an indication was made by the red LED.

8.3.7 Light Sensor Test Procedure

A description of the step-by-step procedure for testing the light sensor is provided in the table below. The procedure is based on the light sensor test plan described previously.

Table 8-7. Light Sensor Test Procedure

Step #	Description	Conditions	Expected Result
1	The light sensor's basic functionality was tested by simply exposing it to a light source and then removing the light.	The test was conducted under controlled conditions to ensure the light sensor is either directly exposed to light or there is no light exposure to the sensor.	The presence of light was successfully detected when it was being exposed to a light source. The red LED was activated.
2	The light sensor's basic functionality was tested by simply exposing it to a light source and then removing the light.	The light sensor was integrated and mounted on the LTU. The test was conducted under controlled conditions to ensure the light sensor is either directly exposed to light or there is no light exposure to the sensor.	The presence of light was successfully detected when it was being exposed to a light source. The red LED was activated.
3	The light sensor's basic functionality was tested by simply exposing it to a light source and then removing the light.	The light sensor was integrated and mounted on the LTU. The test was conducted under controlled conditions to ensure the light sensor was either directly exposed to light or there was no light exposure to the sensor.	The data collected by the microcontroller was sent via SMS, and it was verified that the light sensor was exposed to a light source.

8.4 Software Test Plans

8.4.1 Overall Objective of Software Testing

The purpose of the overall software test effort was to ensure that all software components within the system worked properly and efficiently. The software testing was to effectively expose all errors within the system. This included any previously identified errors and potential errors that could have stemmed from those previously identified errors. Along with exposing high-level problems, the software testing was to also expose low-level problems within specific modules.

For this system, high-level problems were defined as problems that cause the overall process of the system to fail. This included problems with the integration of the modules and communication issues between individual modules. Since a high level of integration was required to develop the LTU, all interfacing was rigorously tested to ensure proper communication had been established between each individual module.

Low-level problems were defined as the errors that occur within the module itself. Both high-level and low-level problems were identified and resolved in a similar manner. However, in order to identify the source of the problem, different tests were executed on both the high-level and low-level functionality of the system in order to pinpoint the specific error(s). Overall, rigorous testing was to eliminate all errors within the system, allowing the system to run flawlessly and as it was initially designed.

8.4.2 Testing Individual Software Components

In order to pinpoint errors within this complex system, individual test cases were to be setup to test specific modules independently. From a software perspective, the on-chip software was to be tested first in a simulated environment to ensure there was no potential damage done to the hardware due to defective software.

8.4.2.1 On-Chip Software Test Plan

In order to avoid dependence upon the availability of hardware, the on-chip software was to be simulated entirely in a software test environment before being implemented on hardware. This approach also served to protect against potential damage to the hardware due to faulty software. The simulation testing was continued until it had successfully passed all tests with no discovered errors.

The on-chip software was then tested in collaboration with the hardware components. It should be noted that the testing procedures used when interfacing the on-chip software and hardware were vigilant as to not damage any of the hardware components. The tests incrementally analyzed the functionality of individual modules, accumulating additional modules as the test successfully progressed. Throughout testing, if communication issues between modules, specifically the GPS and GSM modules, were discovered, the components were individually analyzed to reveal low-lying errors.

8.5 Software Test Procedure

The test procedures described below were based on the software test plans described in section 8.2. Each subsection focused on the test procedure for a particular software module. The table within each subsection provides the step-by-step procedure that was to be taken when testing the specified module. A detailed description, along with the conditions under which the test took place, is provided for each step. The expected results have been replaced with the observed results.

It should be noted that some test cases closely resemble others. Although some steps within the testing procedures may seem repetitive, the somewhat redundant steps were necessary to properly test specific variables. For example, after performing straightforward tests to ensure basic functionality, the same test was conducted with a variation in air pressure. Again, this may initially seem to be redundant, but it was important that this factor be tested independently of other variable changes. This form of testing made it easier to pinpoint the errors that appeared throughout the testing of each individual module.

8.5.1 On-Chip Software Test Procedure

A description of the step-by-step procedure for testing the on-chip software is provided in the table below. The procedure is based on the on-chip software test plan described in section 8.4.2.1. The results of each test are recorded in the far right column for reference purposes.

Table 8-8. On-Chip Software Test Procedure Part 1

Step #	Description	Conditions	Results
1	Basic functionality of the on-chip software was tested using sample inputs to achieve corresponding outputs. This served as a proof-of-concept test.	The test was conducted under basic simulated conditions.	The on-chip software provided the correct output for each of its corresponding inputs.
2	The communication between the GPS and GSM modules was tested by processing a sample text message containing coordinate information.	The test was conducted under basic simulated conditions.	The on-chip software properly processed the GPS coordinates into a text message and sent that information to the GSM module for transmission.
3	Basic functionality of the on-chip software was tested using inputs resembling real-world scenarios (i.e. GPS coordinates, user settings, etc.) to achieve corresponding outputs.	The test was conducted under basic simulated conditions.	The on-chip software correctly received and processed data from the GPS module. The GPS coordinates were processed into a text message and then transmitted via the GSM module.
4	Basic functionality of the on-chip software was tested using inputs resembling real-world scenarios (i.e. GPS coordinates, user settings, etc.) to achieve corresponding outputs.	The test was conducted under a simulated high altitude scenario, with other varying environmental factors.	The on-chip software correctly received and processed data from the GPS module. The GPS coordinates were processed into a text message and then transmitted via the GSM module.
5	Basic functionality of the on-chip software was tested using inputs resembling real-world scenarios (i.e. GPS coordinates, user settings, etc.) to achieve corresponding outputs.	The test was conducted under a simulated environment with varying air pressure to simulate a typical flight path.	The on-chip software correctly received and processed data from the GPS module. The GPS coordinates were processed into a text message and then transmitted via the GSM module.

Table 8-9. On-Chip Software Test Procedure Part 2

Step #	Description	Conditions	Results
1	Full functionality of the on-chip software was tested using extensive, real-world cases, which utilized all its features.	The test was conducted under a simulated environment with varying air pressure to simulate a typical flight path.	The on-chip software correctly received and processed data from the GPS module. The GPS coordinates were processed into a text message and transmitted via GSM.
2	Basic functionality of the phototransistor light sensor was tested using various light inputs to trigger the appropriate interrupts and achieve corresponding outputs.	The test was conducted under basic simulated conditions.	The on-chip software correctly received and processed the input from the light sensor and initiated the appropriate interrupts to notify the user that their luggage was opened.
3	Basic functionality of the on-chip software was tested using sample inputs to achieve corresponding outputs while interfacing with the hardware modules.	The on-chip software was integrated and interfaced with the hardware components and no simulation was executed.	The on-chip software correctly received and processed data from the GPS module. The GPS coordinates were processed into a text message and then transmitted via GSM.
4	Full functionality of the on-chip software was tested using extensive, real-world cases, which utilized all of its features.	The on-chip software was integrated and interfaced with the hardware components and no simulation was executed.	The on-chip software correctly received and processed data from the GPS module. The GPS coordinates were processed into a text message and then transmitted via the GSM.
5	Full functionality of the on-chip software was tested using extensive, real-world cases, which utilized all of its features.	The on-chip software was integrated and interfaced with the hardware components, and a flight path with varying environmental factors was simulated.	The on-chip software correctly received and processed data from the GPS module. The GPS coordinates were processed into a text message and then transmitted via the GSM module.

9 Administrative Content

9.1 Milestones

The following are key milestones that were to be met in order to keep the project moving ahead at a desirable pace. Although the "planned" dates provided were not completely fixed and could fluctuate slightly, they served as a general timeline to hold our group accountable for the work necessary to complete the project. It should be noted that a number of the exact completion dates for these milestones are unknown. In those cases, an estimated date has been provided.

Table 9-1. Project Milestones

Planned	Completed	Milestone
01/20/12	01/20/12	Team members selected
02/03/12	02/03/12	Project selected
03/24/12	04/07/12	Senior Design I documentation 50% complete
04/06/12	05/15/12	GSM/GPS module selected
04/23/12	05/15/12	Battery selected
04/25/12	04/25/12	Senior Design I documentation complete
04/27/12	10/29/12	GPS server setup begun
04/30/12	05/30/12	PCB design begun
05/04/12	04/20/12	Microprocessor selected
05/07/12	09/03/12	PCB design completed
05/14/12	09/12/12	All parts selected and ordered for Phase A
05/28/12	09/20/12	Phase A prototype assembled and testing begun
07/06/12	09/23/12	All parts received for Phase B prototype
07/20/12	09/25/12	Phase B prototype assembled and testing begun
07/28/12	09/12/12	MSP430 to GM862 communication achieved
08/10/12	10/26/12	Phase A software complete
08/17/12	11/20/12	Phase A prototype complete
08/15/12	11/15/12	LTU-A full functionality demonstrated
10/15/12	09/25/12	Phase B prototype complete
11/20/12	11/21/12	Final Testing Complete

In order to ensure the project maintained a smooth and continuous process, group meetings were regularly held, even while classes were not in session. Continuous communication over the summer and into the fall semester allowed any potential problems to be resolved in a timely manner. This assisted in alleviating some of the potential last-minute, unforeseen problems that inevitably materialize when developing such a system.

As seen in the chart above, the assembly and testing of the prototypes was intended to take place over the summer, but due to delays, was not achieved until midway through the fall semester. This timing still ensured that at that a working prototype was ready well in advance of the final demonstration at the end of Senior Design II. This strategy provided our group with ample time to continually expand the system already in place. In addition, the scalability of the system allowed for the addition of other features to further enhance the system's capabilities.

Before adding additional features to the system, a group meeting was held to discuss the monetary cost and amount of time necessary to incorporate these features. A consensus was reached to include these features, and hence the timeline and milestones charts were updated accordingly. As the final deadline approached, the timeline and milestones were continually assessed and reconfigured to include more specific, steadfast dates that needed to be met in order to ensure the completion of the system.

9.2 Budget and Financing

While no major corporations were able to be secured to cover the full cost of the project, we were able to receive a donation from one sponsor. Ms. Sandy Cook, President of Decimal Point Accounting Services in Jacksonville, Florida, generously agreed to contribute $500.00 to help with the funding of the project. This donation covered a significant portion of the overall cost of the project, which can be seen in the table below.

Although there is no direct relationship between Decimal Point Accounting Services and the need of a Luggage Tracking Unit, Mrs. Cook was still interested in the process used to develop such a device and the overall functionality of the device. Her interest was sparked by her own personal experiences of losing luggage while on travel for business. After the idea of using a small device to continually track the luggage throughout its journey was pitched, she agreed to make a donation.

When discussing the system with Ms. Cook, a general, overarching description was used. The discussion took place at the beginning of the project development. Therefore, no specific details or specifications were known. However, as a project sponsor, Ms. Cook has been continually kept up-to-date with new information concerning the overall process and design of the system. Documents are regularly sent to her via email to provide her with updates on how her contribution is being distributed within the system.

The estimated budget for this project is shown in the table below:

Table 9-2. Project Budget

Part	Estimate	Qty	Estimated Total	Actual Total
GM862 Module	120.00	2	240.00	241.29
GM862 Evaluation Board	100.00	1	100.00	84.71
GSM Antenna	20.00	2	40.00	29.64
GPS Antenna	15.00	2	30.00	33.91
Mini USB port	5.00	2	10.00	9.00
Battery	45.00	2	90.00	20.16
Microcontroller	5.00	2	10.00	10.50
Memory	10.00	2	20.00	2.00
Printed Circuit Board	35.00	4	140.00	150.45
50-pin connector	10.00	2	20.00	21.78
Enclosure	15.00	2	30.00	7.68
20P socket	1.00	2	2.00	2.44
Pressure Sensor	10.00	2	20.00	21.32
Phototransistor	2.00	2	4.00	1.84
Capacitors, Resistors, Diodes, Leds	10.00	1	10.00	9.77
NPN transistor	0.50	2	1.00	0.62
AT&T 06/19/2012 - 06/20/2012	100.00	1	100.00	96.96
AT&T 06/21/2012 - 07/20/2012	15.00	1	15.00	15.95
AT&T 07/21/2012 - 08/20/2012	15.00	1	15.00	15.25
AT&T 08/21/2012 - 09/20/2012	15.00	1	15.00	19.38
AT&T 09/21/2012 - 10/20/2012	20.00	1	20.00	30.13
AT&T projected	20.00	1	20.00	70.00
Total			**$ 952.00**	**$ 875.28**

A project briefing was initially planned to be developed to present to potential sponsors. Potential sponsors include those directly or indirectly associated with the buying, selling, or use of luggage.

It was agreed upon by the team that the remaining cost of the project would be divided evenly amongst the team members or adjusted to compensate for which team members kept which pieces of hardware. Unanticipated costs will inevitably be encountered throughout the development of the system. Unexpected expenses may be encountered if additional features are added to the system after all key components have been successfully developed. They may also come about if certain modules are damaged in the process of building either of the prototypes. Nonetheless, each group member has agreed to evenly distribute any unexpected expenses to ensure a working system can be developed.

10 Personnel and Bibliography

10.1 David Farrell

David Farrell is a senior Computer Engineering student at the University of Central Florida. He has worked as a student intern at Lockheed Martin Missiles and Fire Control in Orlando for the past three years. After graduating in December of 2012, David will begin full time employment with Lockheed Martin as a Systems Engineer. In this role, he will be supporting development of the Modernized Target Acquisition/Designation Sight and Pilot Night Vision Sensor for the AH-64 Apache Helicopter. Outside the classroom, David leads the college ministry at his church and enjoys reading, playing sports, and spending time with family.

10.2 Evan Husk

Evan Husk is a senior Computer Engineering student at the University of Central Florida. He is currently conducting research and writing an undergraduate thesis under the supervision of Dr. Avelino Gonzalez. The research and thesis is on *Imitating Individualized Facial Expressions in a Human-Like Avatar through Machine Learning*. After completing his thesis and graduating in December of 2012, he plans on furthering his education by pursuing a master’s degree in systems engineering at the University of Florida. He wishes to work as a systems engineer in the defense industry or in the field of communications. In his free time, Evan likes to play sports, fish, and spend time with his family.

10.3 Jose Mousadi

Jose Mousadi is a senior Electrical Engineering student at the University of Central Florida. For the last 5 years he has been working as a Project Engineer and Electrical Designer at Gas Turbine Efficiency, Inc. in Orlando, Fl. Previously, he worked as a Panel Builder at G&T Conveyor, Inc. in Tavares, Fl. This hand-on approach has aid him acquired extensive knowledge in industrial control systems and instrumentation geared towards the power generation and energy sector. Lately he has designed, developed and managed auxiliary systems used in Gas and Steam Turbines as well as other OEM products. In his free time, Jose likes to travel, spend time with family, and play basketball.

11 Facilities and Equipment

The primary facility for development and testing of our system was originally the Senior Design Laboratory at the University of Central Florida (UCF). Because our team does not reside in the same general area, it was necessary to choose a designated location at which we may meet to work on the development of the project. The UCF campus seemed to be the logical choice for this need as it provided ready access to both engineering equipment in the senior design lab as well as engineering knowledge in the form of instructors and fellow classmates. The Senior Design Lab is equipped with lockers in which equipment and other supporting components and devices may be stored. Due to the small size of our project, we were able to store all of our hardware and other equipment in these lockers.

The lab is also equipped with an ample amount of engineering equipment such as computers, oscilloscopes, power supplies, function generators, and bread boards. If any testing needed to be performed that was not supported by our evaluation board or PCB, we intended to use the equipment in the lab to perform the desired test activities. In general, it was anticipated that the evaluation board purchased by our team would accommodate the majority of our testing needs. The board was equipped with a serial communication interface, power port, and auxiliary I/O ports for interfacing with other devices and components. In other words, the evaluation board served as the hub around which all other supporting components revolved and operated in the testing and development phases of this project.

As the project progressed and steps were made toward completing the first prototype, it was decided that UCF was no longer the best place for us to meet as a group. All three members of the team reside a significant distance from the main UCF campus and were rarely on campus for class, especially during the summer and fall semesters. In addition, we were able to perform all assembly and testing of the LTU and its various components using tools and equipment already in our possession, so there was no need to use the equipment available in the Senior Design Lab.

For the duration of the fall semester, the majority of team meetings were held in the south Orlando area, either at a Panera Bread central to our location or at Evan Husk’s condominium. Both of these locations provided sufficient space and resources to conduct any testing and other problem solving. Additionally, a significant amount of work was accomplished on the project by the team members at their individual places of residence. The figures below show an early test setup at one team member’s residence:

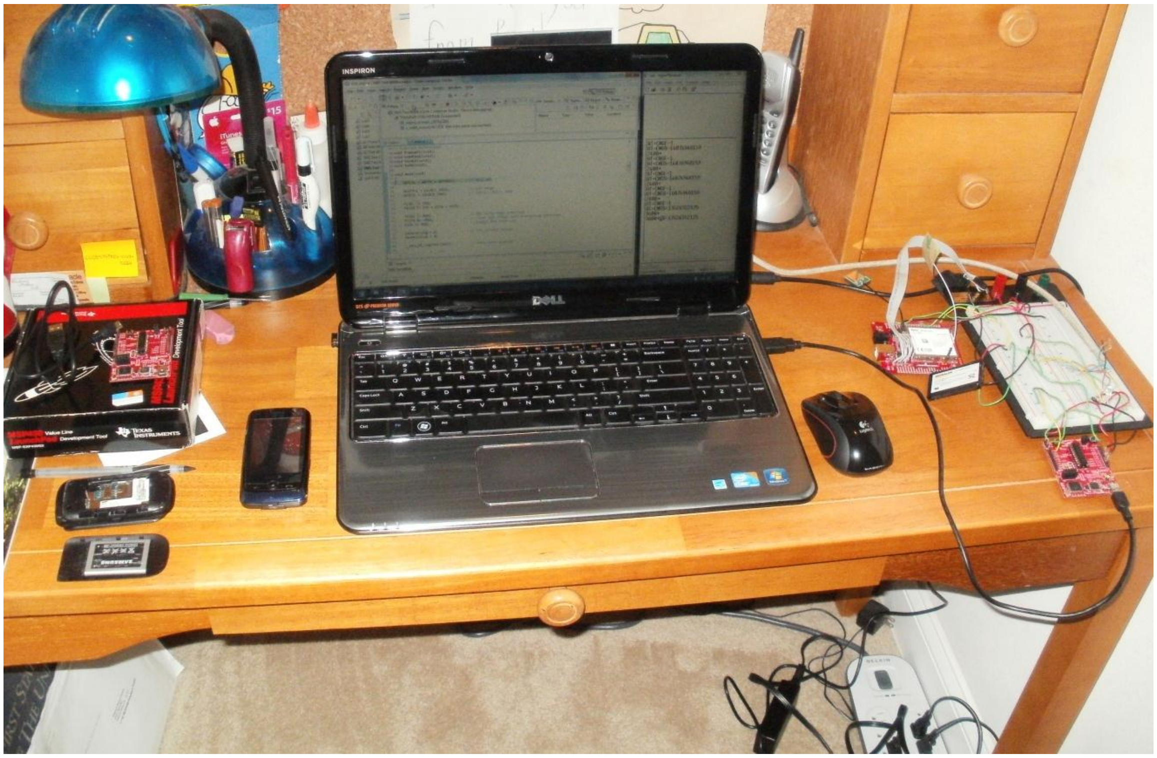

Figure 11-1. Initial Testing Setup

The figure shown below is one of the initial testing setups for verifying the MSP430 to GM862 communication. The GM862 evaluation board can be seen toward the bottom left corner. It is connected to the MSP430 LaunchPad via the bread board in the center of the image.

One key item to note is the black battery pack immediately to the right of the GM862 evaluation board. Using this battery to power the GM862 evaluation board was the corrective action taken to resolve one of the major issues we encountered early in the development process. It was discovered that the 5V power adapter purchased for use with the GM862 evaluation board did not provide enough power to the GM862 module by itself. Enough power was supplied to the GM862 module only if both the 5V power adapter and USB cable were plugged into a wall outlet and computer, respectively. If only the 5V power adapter was plugged into the evaluation board, the GM862 would shut down several seconds after being powered on by the user. This proved to be a difficult problem to trace the root cause of because it seemed certain the cause lay elsewhere, such as being due to the lack of a USB connection.

The logical resolution to this issue was initially to leave both the 5V power adapter and USB cable connected. However, it was determined through trial and error that if both the 5V power adapter and USB cable were connected, communication could not be received by the GM862 through the external pins on its evaluation board. This was due to the fact that signals were being received by the GM862 from two sources, the USB port and the external pins on the evaluation board. These signals conflicted with one another, and the USB cable appeared to override the other input signals coming from the LaunchPad.

After a significant amount of additional trial and error, and after consulting with Sparkfun technical support representatives, the root cause was determined. The issue was able to be resolved by supplying power to the GM862 through the battery via the external power pin on the GM862 evaluation board. This battery was the same battery used later on in the final product and therefore was assured to provide sufficient power to the GM862 module. The breadboard was again used to accomplish this, as it provided a readily configurable and flexible interface for hardwiring components together.

The MSP430 mounted on its LaunchPad can be seeing in the lower right hand corner of the image. The main pins connected between the LaunchPad and the GM862 evaluation board were the TX and RX pins for the UART serial communication.

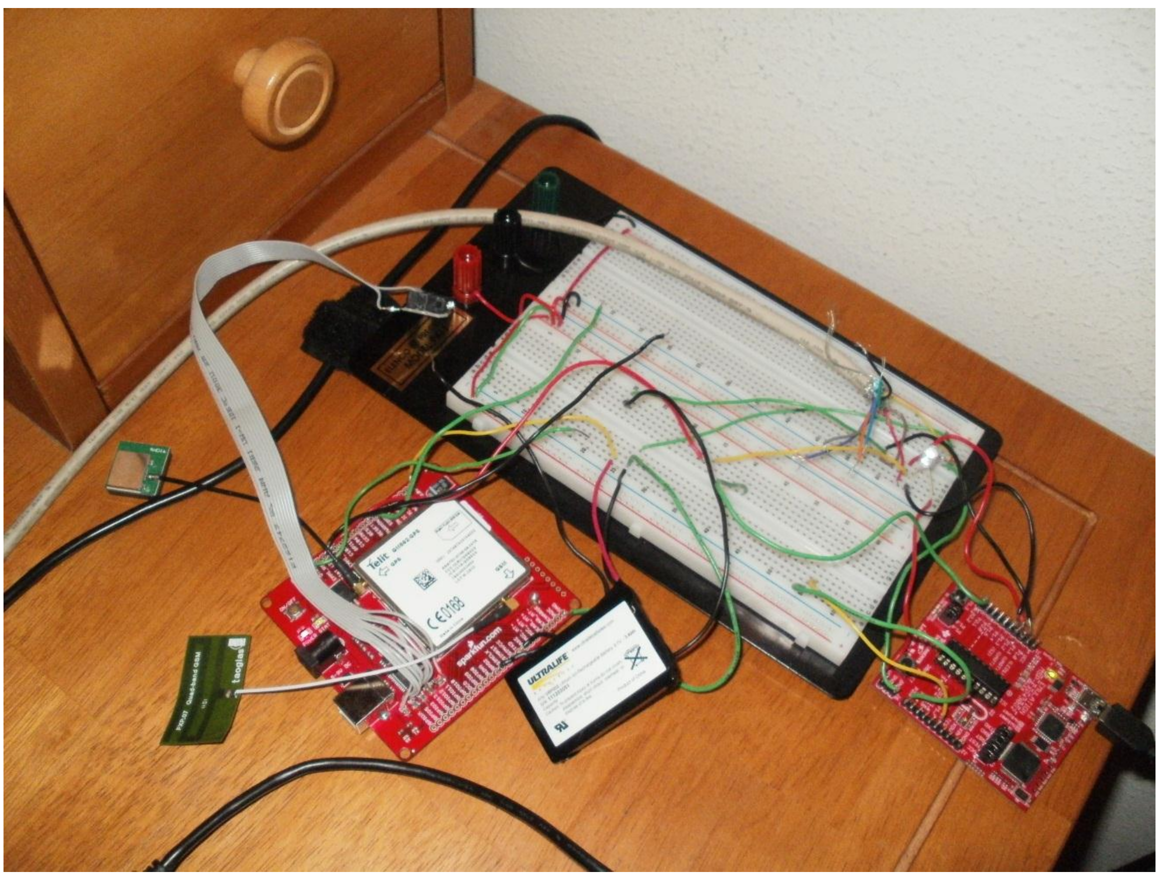

Figure 11-2. Initial MSP430 to GM862 Test Setup

12 Suppliers and Service Providers

Most of the suppliers will be contacted via email or through orders placed online through their websites. The following table lists the suppliers and their websites:

Table 12-1. Vendors, Suppliers and Service Providers

Supplier Name	Website
Mouser Electronics	www.Mouser.com
Digi-Key	www.digikey.com
Sparkfun Electronics	www.sparkfun.com
Semiconductor Store	www.semiconductorstore.com
Advance Circuits	www.4pcb.com
Motorola	www.motorola.com
Newark	www.newark.com
AT&T	www.att.com

The services of Advanced Circuits located in Aurora, CO and Tempe, AZ were required in order to fabricate the PCB. The services of AT&T Mobility were required in order to provide wireless cellular network access. The SIM card was purchased along with a prepaid payment plan. The service plan that fit best for the LTU was the 250 megabyte Data Plan, which cost $16 per month.

As a result of changes to the LTU's primary method of communication, the data provided in our plan was not utilized. Due to unforeseen issues with the GM862, the primary mode of data transmission became SMS messages sent via GSM, not GPRS. This prevented us from using the data we had purchased, and in the end, proved to be somewhat costly. Because no text messages were allocated for this plan, each text message sent cost $0.20, an expense that accumulated rapidly. For future implementations, it is recommended to purchase a plan with unlimited text messages.

13 Appendices

13.1 Acronyms

Abbreviation	Description
AT	Attention
COG	Course Over Ground
COTS	Commercial Off The Shelf
CS	Chip Select
DFC	Design File Check
GPIO	General Purpose Input/Output
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HDOP	Horizontal Dilution Of Precision
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
iOS	(Apple) Operating System
IP	Internet Protocol
LED	Light Emitting Diode
LTU	Luggage Tracking Unit
NMEA	National Marine Electronics Association
NVM	Non Volatile Memory
PC	Personal Computer
PCB	Printed Circuit Board
PDU	Packet Data Unit
RAM	Random Access Memory
SIM	Subscriber Identity Module
SiRF	Silicon Radio Frequency
SMS	Short Message Service
SPI	Serial Peripheral Interface
UCF	University of Central Florida
USB	Universal Serial Bus
UTC	Coordinated Universal Time

13.2 Works Cited

“Air Travel Consumer Report.” *Aviation Consumer Protection and Enforcement*. March 2012. U.S. Department of Transportation. 25 Mar. 2012 <http://airconsumer.dot.gov/reports/2012/March/2012MarchATCR.pdf>.

The OpenGTS Project. 2011. Geo Telematic Solutions Incorporated. 25 Mar. 2012 <http://opengts.sourceforge.net/index.html>.

“GM, GE/GL Families GPS Solutions User Guide.” 12 Dec. 2011. Telit® Wireless Solutions. 30 Mar. 2012 <http://www.telit.com/module/infopool/download.php?id=4099>.

“Telit Modules Software User Guide.” 26 Mar. 2012. Telit® Wireless Solutions. 30 Mar. 2012 <http://www.telit.com/module/infopool/download.php?id=522>.

“AT Commands Reference Guide.” 20 Mar. 2012. Telit® Wireless Solutions. 30 Mar. 2012 <http://www.telit.com/module/infopool/download.php?id=542>.

“GM862 Family Hardware User Guide.” 15 Sept. 2011. Telit® Wireless Solutions. 30 Mar. 2012 <http://www.telit.com/module/infopool/download.php?id=537>.

“Easy Script in Python.” 3 Dec. 2010. Telit® Wireless Solutions. 30 Mar. 2012 <http://www.telit.com/module/infopool/download.php?id=617>.

13.3 Copyright Permissions

13.3.1 Lost Luggage Graph (Figure 2-1)

Charles Tran charles@creditdonkey.com 4:11 PM (30 minutes ago)
to me

Hi David,

Thanks for contacting us.

Yes, you may use the "Infographics: Lost Baggage" image from our website. The image is licensed under: http://creativecommons.org/licenses/by-nd/3.0/us/

Good luck with your Senior Design project! Feel free to contact me if you need anything.

...

Charles

On Sun, Mar 25, 2012 at 12:30 PM, David Farrell <farrelldc@gmail.com> wrote:

> Hello,
>
> My name is David Farrell, and I am a student at the University of Central Florida. I would like to ask your permission to use the "Infographics: Lost Baggage" image from your website. A link to this image is provided below:
>
> http://www.creditdonkey.com/lost-luggage.html
>
> The image will be used in a Senior Design project document, and you will be credited and cited appropriately. This is purely for academic use and will not be used for commercial purposes. Please let me know if this is permissible.
>
> Thank you,
>
> David Farrell
> farrelldc@gmail.com

13.3.2 PocketFinder Image (Figure 4-1)

Greg Gaines greg.gaines@pocketfinder.com 2:29 PM (2 hours ago)
to me, service

David,
Thanks for your message and your interest in Location Based Technologies' PocketFinder family of products.
You have our permission and good luck with your project.

It would be great if I could get a copy of your work, as well.
Thanks and best regards,
Greg

Gregory Gaines | Location Based Technologies | Tel (888) 600-1044 Ext 8 | Cell (949) 572 7700 | Fax (714) 200-0287 | www.PocketFinder.com

Follow the Historic 911 Resolve and Remember Journey from Carmel NY to Carmel CA
http://www.pocketfinder.com/carmel911memorial/

CONFIDENTIAL: The information contained in this e-mail message may contain privileged or confidential information and is intended solely for the use of the individual or entity named above. If you are not the intended recipient or the person responsible to deliver the foregoing to the intended recipient, you are hereby notified that any use, dissemination or duplication of the foregoing is strictly prohibited. If you have received this message in error, kindly notify the sender immediately and delete this e-mail from your system.

From: David Farrell [mailto:farrelldc@gmail.com]
Sent: Sunday, March 25, 2012 1:00 PM
To: service
Subject: Copyright Permission

...

Hello,

My name is David Farrell, and I am a student at the University of Central Florida. I would like to ask your permission to use an image of your PocketFinder product and an image of your mobile app "Zone Alerts" screen. Links to these images are provided below:

PocketFinder product image: http://www.boston.com/partners/greader/prfmkt/images/12techlabplus_PocketFinder.jpg

Mobile app (Zone Alerts) image: http://www.pocketfinder.com/gps-locator/cell-phone-gps-locator/

These images will be used in a Senior Design project document, and you will be credited and cited appropriately. This is purely for academic use and will not be used for commercial purposes. Please let me know if this is permissible.

Thank you,

David Farrell
farrelldc@gmail.com

13.3.3 OpenGTS Figures

Martin D. Flynn mflynn@opengts.com 8:19 PM (17 hours ago)
to me, devstaff

Hello David,

Thank you for your request for permission to use these image.
Yes, you may use these images in your Senior Design project documentation.

Best Regards,
- Martin

...

On 4/20/12 5:00 PM, David Farrell wrote:
> Hello,
>
> My name is David Farrell, and I am a student at the University of Central Florida. I would like to ask your
> permission to use two images from the OpenGTS website and demo page. A link to the first image is provided
> below, and the second image is attached, as it is within the system demo site and could not be directly linked.
>
> Track Map image: http://www.opengts.org/TrackMap.jpeg
>
> Geozone image: (attached)
>
> The images will be used in a Senior Design project document, and you will be credited and cited
> appropriately. This is purely for academic use and will not be used for commercial purposes. Please let me
> know if this is permissible.
>
> Thank you,
>
> David Farrell
> farrelldc@gmail.com <mailto:farrelldc@gmail.com>

13.3.4 OpenGTS System Architecture Diagram (Figure 5-17)

Martin D. Flynn mflynn@opengts.com 6:53 PM (2 minutes ago)
to me, devstaff

Hello David,

Yes, you may use the System Architecture diagram. We only ask that you include the following,
or similar, in your reference section with regard to the System Architeture diagram:
"Copyright 2012 GeoTelematic Solutions, Inc, used by permission, all rights reserved."

Thank you very much.
Best Regards,
- Martin

On 3/26/12 3:00 PM, David Farrell wrote:
> Hello,
>
> My name is David Farrell, and I am a student at the University of Central Florida. I would like to ask your
> permission to use the OpenGTS System Architecture diagram found in the OpenGTS Installation and Configuration
> Manual. A link to the referenced document is provided below:
>
> http://www.opengts.org/OpenGTS_Config.pdf _
> _
>
> The diagram will be used in a Senior Design project document, and you will be credited and cited
> appropriately. This is purely for academic use and will not be used for commercial purposes. Please let me
> know if this is permissible.
>
> Thank you,
>
> David Farrell
> farrelldc@gmail.com <mailto:farrelldc@gmail.com>

13.3.5 Nearson 2.4GHz Fixed Mount Swivel Antenna (Figure 5-7)

Martin Wan Add to contacts
To josemousadi@hotmail.com
Reply

Dear Jose,

Thanks for checking with us for the authorization to use of our product images. It is permissible for you and your project related person to use our products images purely for your academic use and they shall not be used for any commercial purposes.

Please feel free to contact us for any future issues. Good luck for your project works.

Martin Wan

NEARSON

Nearson, Inc. • 3775-C Pickett Road • Fairfax • VA • 22031-3603
Tel: +1-703-913-5552x8301 • Fax: +1-703-913-5553

-----Original Message-----
From: Jose Mousadi <josemousadi@hotmail.com>
To: <sales_support@nearson.com>
Date: Tue, 17 Apr 2012 10:25:31 -0400
Subject: FW: Copyright Permission request

Hi,
My name is Jose Mousadi and I'm a senior Electrical Engineer at the University of Central Florida. At this moment, we are working on our Senior Design Project, and one of the components requires the use of the "ANTENNA RUBB DUCK 2.4GHZ 5"CABLE". We would like to ask permittion to use some of the figures that appear on the specification page for the Nearson part number S181FL-5-RMM-2450S antenna.

These images will be used in the Senior Design project documentation, and you will be credited and cited appropriately. This is purely for academic use and will not be used for any commercial purposes. Please advise if this is permissible.

Thank you,
Jose Mousadi

13.3.6 Molex Figures (Figure 5-9 and Figure 5-10)

Hernandez, Hans Add to contacts 6:08 PM
Reply

Good afternoon,

Thank you for the request. This should be ok.

Best Regards,
Hans

Hans Hernandez
Associate Sales Engineer
Americas Region
2222 Wellington Ct.
Lisle, IL 60532
Ph: 1-800-786-6539 ext. 555-2122
FAX: 630-813-2122
Hans.Hernandez@molex.com

-----Original Message-----
From: feedback@molex.com [mailto:feedback@molex.com]
Sent: Wednesday, April 11, 2012 7:01 AM
To: amerinfo
Subject: Copyright Permission request
INTERNET ADDRESS:
SENDER:Jose Mousadi

Sender's Contact Information

Name:
Company: University of Central Florida
Business Phone:

Hi,
My name is Jose Mousadi and I'm a senior Electrical Engineer at the University of Central Florida. At this moment, we are working on our Senior Design Project, and one of the components requires the use of the Molex 501920-5001 connector. We would like to ask permittion to use some of the figures that appear on the specification page for the Molex part number 52991-0508 receptacle and the Molex part number 501920-5001 plug.

These images will be used in the Senior Design project documentation, and you will be credited and cited appropriately. This is purely for academic use and will not be used for any commercial purposes. Please advise if this is permissible.

Thank you,
Jose Mousadi
josemousadi@hotmail.com