

NextGen Asset Tracking (NAT) Device

Group 1 – NAT

Team Members

Brianna Thomason	Computer Engineer
Brittney Fry	Computer Engineer, Project Manager
Lucas Dickinson	Electrical Engineer, Computer Engineer
Ralph Baird	Computer Engineer
Wayne Marshall	Electrical Engineer

Sponsor(s): *Young Engineering Services, LLC*

Under the direction of Michael F. Young, *UCF CS Adjunct Professor*

December 4th, 2017

Table of Contents

1.0 Executive Summary	1
2.0 Project Description.....	2
2.1 Project Motivation.....	2
2.2 Goals and Objectives	4
2.3 Requirement Specifications	5
2.4 Initial Project Block Diagrams.....	6
2.5 Quality of House Analysis	8
2.5.1 Overview.....	8
2.5.2 Understanding/Reading the House of Quality	8
2.5.3 Engineering/Technical Requirement Comparisons.....	9
2.5.4 Final Analysis	14
2.6 Project Operation Manual	15
2.7 Project Use Cases.....	18
2.7.1 General Simple User	18
2.7.2 Company User	18
3.0 Research Related to Project Definition.....	20
3.1 Existing Similar Projects and Products.....	20
3.1.1 Similar Projects.....	20
3.1.2 Similar Products.....	23
3.2 Relevant Technologies.....	27
3.2.1 Radio Frequency Wireless Communication Technology	27
3.2.2 Bluetooth Technology.....	38
3.2.3 RFID Technology	38
3.3 Strategic Components and Part Selections.....	42
3.3.1 Embedded Computing Hardware.....	43
3.3.2 The Microcontroller Modules	44
3.3.3 GPS Module.....	45
3.3.4 IMU.....	55
3.3.5 HID	73
3.3.6 Power Components	81
3.3.7 RPMA Modules	84
3.3.8 Cellular Modules.....	85
3.4 Possible Architectures and Related Diagrams	85
3.5 Parts Selection Summary	90
3.6 Bill of Materials	95
3.7 Printed Circuit Board Technology	96
3.7.1 Composition of a PCB	96

3.7.2 PCB Design Software	97
3.7.3 PCB Design Constraints.....	97
3.7.4 Final PCB.....	98
4.0 Related Standards and Realistic Design Constraints	99
4.1 Standards.....	99
4.1.1 Product Relevant standards.....	99
4.1.2 Design Impact of Relevant Standards	105
4.2 Realistic Design Constraints	106
4.2.1 Economic and Time Constraints	106
4.2.2 Environmental, Social, and Political Constraints.....	106
4.2.3 Ethical, Health, and Safety Constraints.....	106
4.2.4 Manufacturability and Sustainability Constraints	107
5.0 Project Hardware and Software Design Details.....	108
5.1 Initial Design Architectures and Related Diagrams	108
5.1.1 Electrical Architecture	108
5.1.2 Power State of Components	109
5.1.3 Software Architecture	110
5.2 DC/DC Converter.....	111
5.3 Voltage Reference.....	112
5.4 Microcontroller	113
5.5 Software Design: Modules.....	115
5.5.1 GPS	116
5.5.2 Configuration GUI.....	116
5.5.3 UART.....	132
5.5.4 MPU.....	133
5.5.5 Telecommunications	133
5.5.6 Web GUI Model-View-Controller.....	134
5.5.7 IMU.....	134
5.5.8 HID / USB.....	134
5.5.9 INS Skeleton	134
5.6 Flow Charts.....	134
5.7 Summary of Design	135
6.0 Project Prototype Construction and Coding.....	137
6.1 Integrated Schematics	137
6.1.1 OrCAD.....	137
6.1.2 The Schematic.....	137
6.2 PCB Vendor and Assembly	141
6.3 Software Prototype.....	141
6.4 Final Coding Plan.....	142

7.0	Project Prototype Testing Plan.....	145
7.1	Hardware Test Environment	145
7.1.1	GPS Module.....	145
7.1.2	Microcontroller Development board.....	145
7.1.3	Wireless Communication Modules	145
7.2	Hardware Specific Testing.....	146
7.2.1	GPS	146
7.2.2	Microcontroller	148
7.2.3	MPU.....	148
7.2.4	RPMA	148
7.2.5	LTE Module.....	150
7.2.6	DC-DC Converter	153
7.2.7	Voltage Reference Generator.....	154
7.3	Software Test Environment.....	155
7.4	Software Specific Testing	156
8.0	Administrative Content.....	159
8.1	Milestone Discussion	159
8.2	Budget and Finance Discussion	161
8.3	The Winter Park Laboratory	163
9.0	Appendixes	165
9.1	References.....	165
9.2	Copyright Permissions	170
9.3	Additional Testing Images	183

Table of Figures

Figure 1 - Asset Tracking viewed on Map.....	2
Figure 2 - Shipping Container Yard.....	3
Figure 3 - Existing Tracking Technologies Comparison	3
Figure 4 - Inertial Navigation System (INS).....	4
Figure 5 - Network System Diagram using Cellular Technology.....	6
Figure 6 - Network System Diagram using RPMA Technology	6
Figure 7 - Initial NAT System Block Diagram.....	7
Figure 8 - NAT House of Quality	9
Figure 9 - NAT Web GUI Login Screen.....	15
Figure 10 - NAT Web GUI Home Page	16
Figure 11 - NAT Web GUI Add New Device	16
Figure 12 - NAT Web GUI Map.....	17
Figure 13 - NAT Web GUI Table of Locations.....	17
Figure 14 - Helmet Tracking System.....	21
Figure 15 - XT-2000	23
Figure 16 - STI_Bolt.....	24
Figure 17 - GX350.....	25
Figure 18 - STI GL300	26
Figure 19 - Star Topology [P].....	30
Figure 20 - Lora Topology [Q]	31
Figure 21 - Deployment scenarios of NB-IoT.....	32
Figure 22 - RPMA US Coverage Map [DDDD].....	33
Figure 23 - LoRa US Coverage Map [S]	34
Figure 24 - LPWAN Comparison Table [J].....	34
Figure 25 - Power Profile for LTE-M1 Power Saving Mode.	35
Figure 26 - Power profile of an LTE-M1 UE in eDRX Mode.....	36
Figure 27 - Comparison of three LTE technologies.....	37
Figure 28 - RFID Communication [TTT]	39
Figure 29- Active vs. Passive Tags [UUU].....	41
Figure 30 - RFID Frequency Band Comparison [QQQ].....	42
Figure 31 - Microcontroller Comparison Table.....	45
Figure 32 - GPS Module Comparison Table	53
Figure 33 - MPU 9250	55
Figure 34 - MPU-9250 Block Diagram [PPP]	58
Figure 35 - MPU 9250: I2C operation, SPI operation (respectively) [PPP]	59
Figure 36 - MPU-9250: The User-Accessible Power Modes [PPP]	62
Figure 37 - I/O Levels and Connections [PPP].....	64
Figure 38 - Orientation of Axes of Sensitivity and Polarity of Rotation	64
Figure 39 - Orientation of Axes of Sensitivity for Compass	65
Figure 40 - ADXL335 [VVV]	67
Figure 41 - ADXL345 [WWW].....	67
Figure 42 - LIS331 [XXX]	68
Figure 43 - L3G4200D [YYY]	70
Figure 44 - ITG3200 / ADXL345 [ZZZ] Figure 45 - ITG3200 / ADXL345 [ZZZ].....	71

Figure 46 - MPU 6050 [AAAA]	71
Figure 47 - IMU Comparison Chart.....	72
Figure 48 - HID Device and Host Communication [BBBB]	73
Figure 49 - USB Interface between a PC and an Embedded System.....	74
Figure 50 - Voltage Regulator Comparison Table	83
Figure 51 - Voltage Regulator Comparison Table	84
Figure 52 - Hardware Prototype Architecture: Modular Design	86
Figure 53 - Hardware Prototype Architecture: Integrated Design	86
Figure 54 - Hardware Prototype Possible Block Diagram.....	88
Figure 55 – (Left) Testing Setup for our Development Board, (Right) Ingenu RPMA Module	91
Figure 56 – NAT Dev Board: Front View Figure 57 - Nat Dev Board: Back View	91
Figure 58 - All Components	91
Figure 59 - Bill of Material (BOM) Table	95
Figure 60 - Basic PCB composition.....	96
Figure 61 - Final NAT PCB Layout.....	98
Figure 62 - Electrical Block Diagram	108
Figure 63 - Power State of Components Table	109
Figure 64 - MVC Pattern Permission to use granted under Public Domain	110
Figure 65 - Buck Regulator Operating Cycle. On State (Left) Off State (Right)	111
Figure 66 - TPS62200 Application Circuit	111
Figure 67 - (left) Diode Voltage Reference Implemented by a BJT (right) a Zener Avalanche Voltage Reference [E]	112
Figure 68 - Basic Bandgap Voltage Reference [E].....	113
Figure 69 - ISL2108 Application Circuit.....	113
Figure 70 - MOSFET Switch.....	114
Figure 71 - Crystal Oscillator Equivalent Circuit	115
Figure 72 - Crystal Oscillators used with a Microcontroller [I].....	115
Figure 73 - Configuration GUI - Configuration and Settings Tab.....	118
Figure 74 - Configuration GUI - Data Display Tab.....	119
Figure 75 - Configuration GUI - Testing Modules Tab.....	119
Figure 76 - nat.conf File Contents	124
Figure 77 - GUI Connected.....	126
Figure 78 - General Use	127
Figure 79 - Program Device Button Pressed.....	128
Figure 80 - Hardware Status Button Pushed.....	129
Figure 81 - Sync Device Data and Time Button Pushed	130
Figure 82 - Retrieve Button Pushed.....	131
Figure 83 - SD Card Flowcharts (Left) Clear, (Center) Save All, (Right) View Contents	132
Figure 84 - Main Device Software Flow Chart.....	135
Figure 85 - Integrated Schematic-Semester 1	138
Figure 86 - Final Schematic	140
Figure 87 - Software Prototype Flowchart.....	141
Figure 88 - Software Prototype Terminal Output	142
Figure 89 - GPS Acquisition times test	147
Figure 90 - GPS Acquisition Times	147
Figure 91 - GPS power supply currents	148
Figure 92 - RPMA Oscillator Calibration State.....	149
Figure 93 - RPMA current consumption during various states.....	150

Figure 94 - XBTU Configuration Settings for LTE Module	151
Figure 95 - XBTU Sending SMS Messages	152
Figure 96 - SMS Messages from LTE Module on Engineer's Phone	152
Figure 97 - LTE Module Power Supply Current Measurements	153
Figure 98 - Voltage Reference Generator Dropout voltage	155
Figure 99 - Initial NAT Project Gantt Chart	159
Figure 100 - Expanded NAT Project Gantt Chart.....	160
Figure 101 - Initial NAT Project General Budget.....	161
Figure 102 - Detailed NAT Project Budget	162

1.0 Executive Summary

Everybody has items that they have lost. Whether that be their keys, their wallet, their phone, or a myriad of other items that are important to them, yet they still manage to unintentionally lose. That's why many products have been developed to find such lost items. Find my iPhone is a feature that Apple implemented in their cellular and mobile devices because their users indicated to them that their phones get lost or stolen on a fairly frequently basis. Samsung has implemented a similar feature called Find my Mobile for the same reasons. Cars get stolen enough that OnStar has been developed so that it can find it using GPS technology embedded in the vehicle. For frequently lost cars there are modules that can be placed in the vehicle so the owner can find it themselves through an app on their phone. Car keys have dongles, wallets have trackers, and so on. People lose things, it's a fact of life. However, in addition to ordinary individuals, companies also lose things that they then need to find. Hospitals lose carts within the many hallways and rooms that they need to find, shipping containers in a yard may get lost within the masses, and it may just be overall easier for a company to plant a tracker on an item than endlessly manually track where all of their products are. This is where our device really comes into action.

This project is a next generation asset tracking device, the NextGen Asset Tracker or NAT.

Our project will utilize GPS and, eventually, INS to track the location of anything our device is attached to and relay this information to the client through an interface, such as a Windows form based GUI or a Web based GUI. This information will be transferred to said interface through communication over a wireless network. Our device will have 2 options from which to communicate its data to the user, either through RPMA or Cellular communication networks.

The difference between our device and a normal tracker will be that it will eventually utilize the connected IMU to run INS, Inertial Navigation System, when the GPS loses signal. The GPS may lose signal for any number of reasons, but if/when it does the INS algorithm will utilize the last known GPS signal and the values being read from the IMU to calculate the new location.

INS is a technology that is already being utilized in airplanes and submarines. Essentially what it does is it uses information to calculate the devices new location. Airplanes will get a GPS fix on where it is when it starts its flight and then utilize an on-board Inertial Measurement Unit which tells the plane information like acceleration and rotational movement so that the plane can calculate based on the last known location where it is now. This technology has not yet been created for a device so small.

The device will be designed with a PCB layout, the PCB will be printed, the components stuffed, and then the corresponding firmware and GUIs will be designed, coded, and implemented.

This report will document the process our team has gone through and will go through during the development of the NAT. It will begin with a more in depth look at the motivations and objective of our device. Then, moving forward onto the research that was done into the design, including the design constraints and standards that we have worked with and around. Finally, it will document the actual selection of the modules and components of the device, the design around those modules, and the testing of those modules proving they meet our requirements. It will also document some of the administrative aspects, including budgeting of both time and finances.

2.0 Project Description

This section will give more details to our product, yet still an overview that the rest of this document will go into further details about. It will start with the overall motivations and goals, followed with the objective of the device, then further on to the product specifications that our team will work to meet in our design, and then finally a discussion over the house of quality.

2.1 Project Motivation

This project is about the designing, developing, and testing of a low cost, low-power, next-generation asset tracking device.



Figure 1 - Asset Tracking viewed on Map

*Permission Requested [P3]

An article in InCorp indicated that in the U.S. alone there is an average estimated range of \$20 to \$50 billion dollars in stolen assets or equipment by employee "making it one of the most costly and widespread challenges faced in today's business world" (InCorp, 2017). Why continue to take the risk associated with manual tracking of your assets or equipment? In today's advanced world where Internet access is at everyone's fingertips, it should be just as easy to locate your equipment by tracking it in real time. A 24-hour real time tracking device for equipment comes with many benefits. One will have peace of mind knowing where their equipment is at all times. With our NextGen Asset Tracking (NAT) device, you will have the ability to track your equipment whenever you need to. If your equipment gets lost or stolen, you could easily locate it. This would save you not just the money you would need to replace it but also the time you would spend looking. You will also be able to track

With the NAT device, if something gets lost or stolen, one will have the ability to deploy objects that can be located easily. This is ideal for things like construction equipment, shipping containers, airport equipment, military equipment, oil field equipment, computer communication equipment, and especially objects with no power themselves. Individuals and companies can easily lose these items, or take many hours diligently documenting their locations. This is a waste of time that could be used to complete beneficial work for the company, and instead place a GPS or similar tracking device to the item and spend less time making sure you can find it the next time even if someone

happens to move it. Speaking only of the shipping yard shown below in Figure 2, a misstep in documenting where each one of these are could be disastrous in locating it again.



Figure 2 - Shipping Container Yard

*Permission Requested [P4]

The device will also be useful in tracking packages that are being shipped, not just the vehicle it travels in, and in tracking people for use in law enforcement and private investigations. Obviously, GPS is not new technology, and neither are GPS tracking devices. Current trackers exclusively use GPS, very expensive hardware, and use LTE to send messages. This is very expensive and has a very negative effect on battery life. Some cell phones have GPS tracking built in but we are not competing with cell phones. Cell phones can cost anywhere from \$700 to \$800 and therefore would not be a very cost effective tracking device. The table below in Figure 3 highlights some of these devices and their features:

Device	Real-Time	Battery life	Price	Monthly Fee
STI_GL300 Real-Time GPS Tracker	✓	2 weeks	\$70	--
GX350 Real-Time GPS Tracker	✓	2.5 weeks	\$100	--
XT-2000 OBD Real-Time GPS Vehicle Tracker	✓	Powered by vehicle battery	\$120	\$25
STI_Bolt Asset Tracker	X	9 months	\$130	\$14.95
Tracking Key GPS Logger 2	X	20 hours	\$170	N/A
STI_GL300 Real-Time Tracker w/ 6 Month Battery & Case	✓	6 months	\$230	--

Figure 3 - Existing Tracking Technologies Comparison

These types of products that our device may compete with will be discussed later in the paper in the existing projects section.

One way the NAT device differs from other devices is that it will have INS as a backup when GPS signal is lost. While others only use cellular networks to report data, we will also use a new wireless technology from Ingenu called Random Phase Multiple Access (RPMA). RPMA is a technology communication system that uses direct-sequence spread spectrum (DSSS) with multiple access. It utilizes the globally available and free 2.4 GHz band. In addition to RPMA, the NAT device will use a cellular communication technology module. RPMA is a Low Power Wide Area (LPWA) technologies which will contribute to our device having low power consumption, giving us longer a battery life. The module we will be designing will also be able to take analog or digital inputs, or be able to produce either analog or digital outputs.

2.2 Goals and Objectives

The objective of our project is to design and build a low-cost IoT module that can be attached to anything and report back its location to the appropriate party. It shall primarily do this with GPS, however it will have a secondary position based on Inertial Navigation System (INS) when GPS signal is not available, or is suddenly lost.

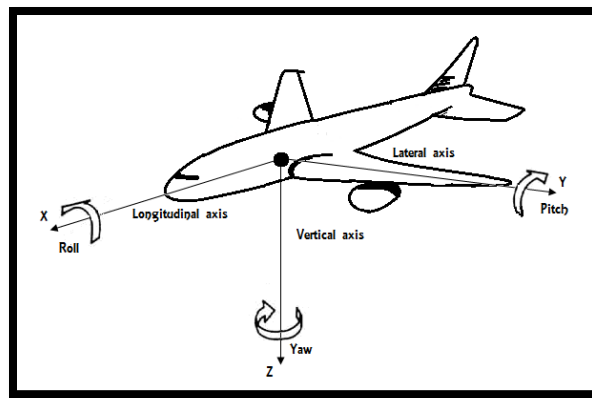


Figure 4 - Inertial Navigation System (INS)

*Permission Granted by Citation [EE]

The additional objective, potentially for future revisions past the scope of this project, is to have the hardware inputs allow for a variety of additional sensors that can monitor and report back on a myriad of conditions.

Some general goals of this project include that the device is small and portable. Keeping in mind that other, more important requirements may change this goal, the device will be designed to be smaller than the size of a credit card and not very much thicker than 1/2 to 1 in. Another goal will be that the device consumes as little power as possible, so that the battery life is expanded as long as possible. This is due to the fact that our users want this device to stay working properly for as long as their product remains in the field. They would not want to have to go change out batteries or full devices every few months. Having a battery that lasts at least 3 months, but hopefully up to years is a goal of this project. Accuracy is also critical for this project, as a tracker that is not

accurate is not of very much use. All of these goals are detailed more in the next section, requirements specifications.

2.3 Requirement Specifications

General Requirements

- Shall produce 1 PCB as well as supplemental software
- Shall produce a product of size smaller than 5.370 in ×4.125 in and less than ½ in thick
- Shall operate on a single 3.7V battery
- Option to operate on a 10V to 30V external DC power source
- Utilizes RPMA Wireless data link
- Utilizes Cellular wireless data link
- Input taken as motion, GPS, power
- Output data to the radio modules
- Interface to the Ingenu wireless module
- Interface to a licensed cellular radio device
- TTL serial data interface for design and debugging
- USB interface to configuring the device
- Interface to the PIC processor to program it and for debugging
- Optional I/O support inputs from a variety of external sensors using an I2C, SPI, contact closures and/or analog inputs/outputs

Device Hardware Components

- Low power 16-bit MCU
- GPS receiver
- 9-Axis motion sensing device
- RPMA radio module
- Cellular radio module
- Micro-SD module
- Ability to record data
- Implement an antenna
- Interface to external devices using contact closures, I2C, and SPI
- Able to switch DC power to external devices using software.
- Able to run for at minimum some minimum time frame based on the client's specifications

Software Modules and Programs

- Able to configure the device through HID interface
- Option to be able to configure client specific settings through a web browser for cellular
- Option to be able to configure client specific settings through a window based GUI
- Windows based GUI to configure the device
- Able to read and decode the motion sensing hardware data
- I2C interface with GPS
- SPI interface with RPMA Radio module

- Interface to Cellular radio module
- Software to create the data packet to be sent on the Ingenu network
- Software to encrypt and decrypt the packets to and from the Ingenu network
- Windows GUI for retrieving data from the Ingenu network servers
- Windows based positioning software that tells user where selected devices are located
- Optional INS system algorithm that takes over when GPS system is lost
- Optional Unique INS system software
- Software interface from cellular module to cellular carrier

2.4 Initial Project Block Diagrams

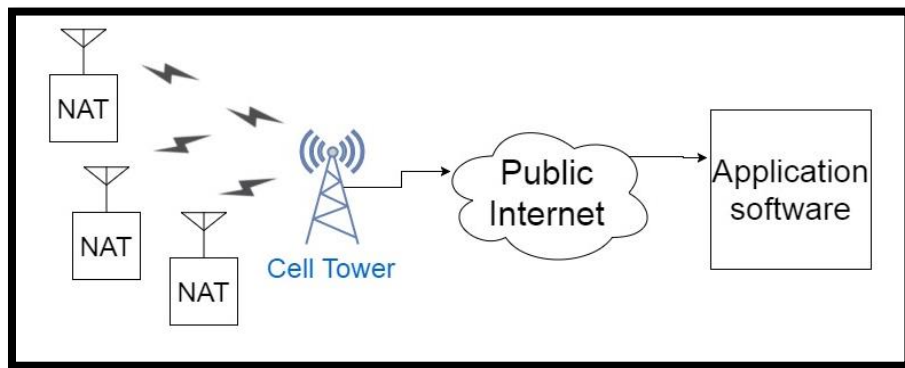


Figure 5 - Network System Diagram using Cellular Technology

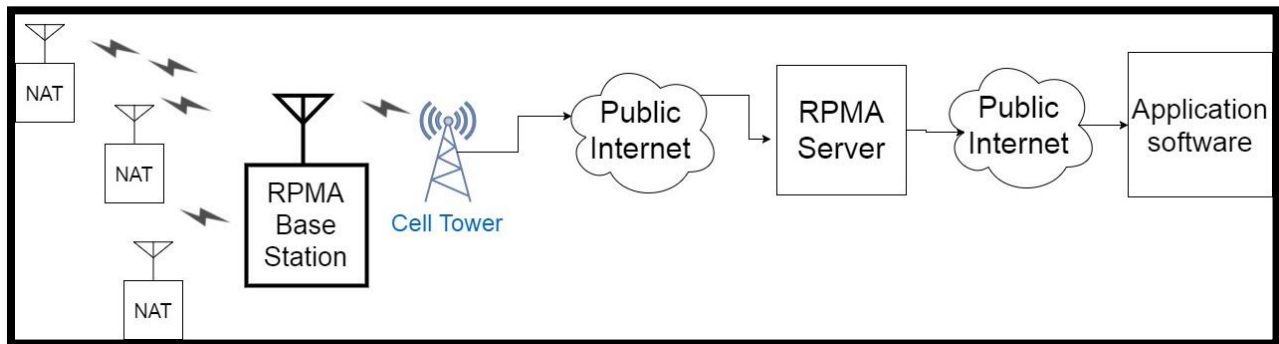


Figure 6 - Network System Diagram using RPMA Technology

Figure 5 and Figure 6 show the system communication diagram, which depicts how our system will connect to other systems via communication networks. Two options we have posed are communicating through cellular networks, or through RPMA (Random phase multiple access), which is a low power wide area network that was made to connect machines to machines through the IoT, shown in Figure 5 and 6 respectively.

Figure 5 depicts that with a cellular communication module, the instances of our device (NAT) will communicate directly with the cell towers. This option is simpler, and yet still a plausible

implementation. We plan to use cellular technology since our data throughput requirement is so low that the cost requirement may not be a limiting factor.

Figure 6 shows multiple instances of our system (NAT) which have their own built in antenna, speaking to the RPMA base station which then connects further to the cell towers. Research is currently being done into both systems for our application, and additionally we would like to research and test the option of implementing both options.

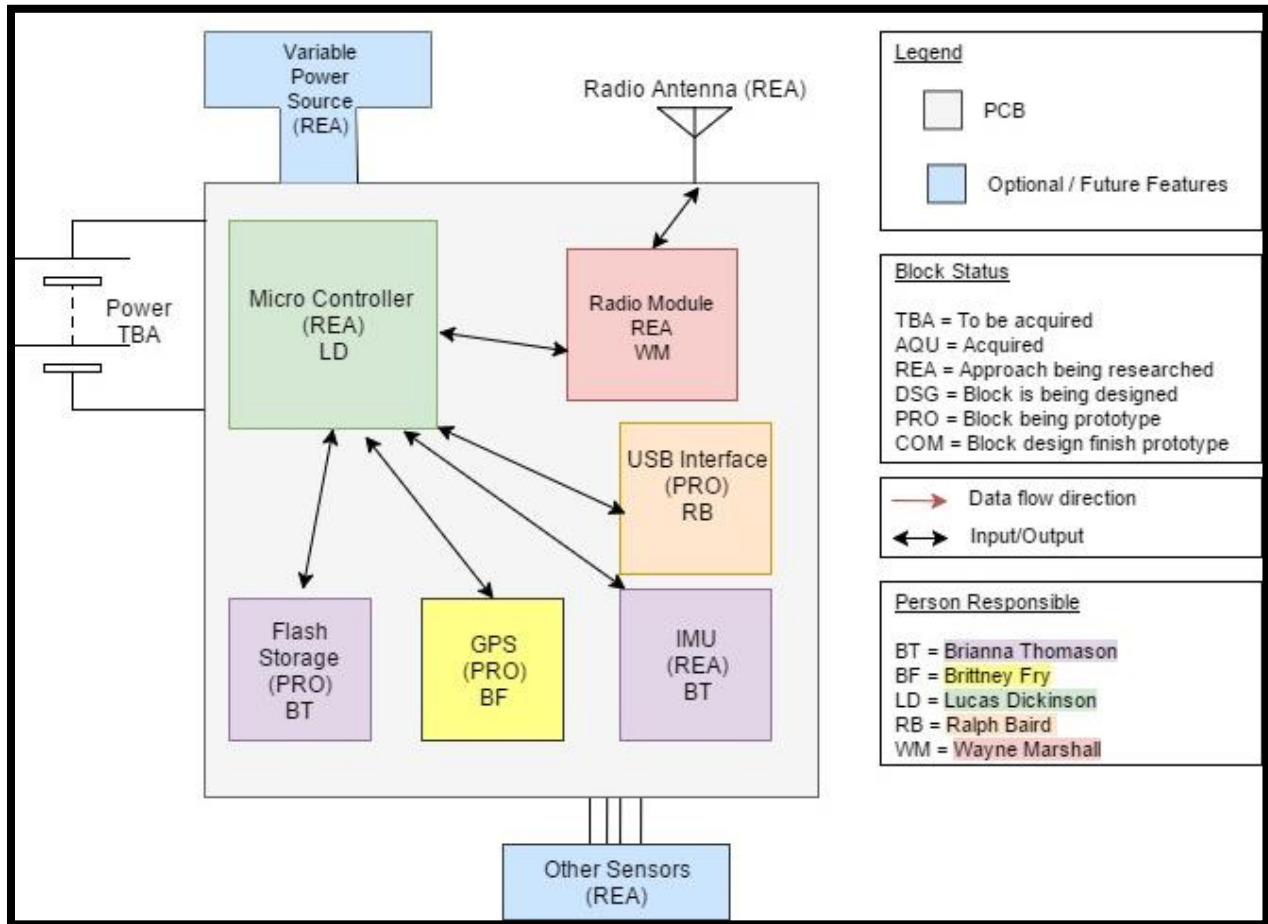


Figure 7 - Initial NAT System Block Diagram

Figure 7 shows the NAT system block diagram, which depicts a basic look at the hardware modules that will be implemented. We will be utilizing some microcontroller, which will be able to support the different communication interfaces as well as consume the lowest amount of power. This microcontroller will take input and give output to all of the components in the system. It also will be where the software that the CpE (Computer Engineering) team writes will be stored and ran, called the firmware from this point in the document. The radio module, which may provide RPMA communication, cellular communication, or both. The IMU (Inertial Measurement Unit) that will output values of or related to acceleration, magnetic field, and orientation. These values will be sent to the microcontroller, so that the INS (inertial navigation system) algorithm can be ran on the values. The algorithm is not included in this specific project requirement, but the CpE

team will provide a base or skeleton where possible to allow the project to be continued after the scope of this projects timeline. The GPS module will read the current location of the device, using its internal hardware. It takes only input from the satellites that its antenna reads. The flash storage will store the value of every location the GPS module reads. All modules only input and output internal to the PCB via the microcontroller. All other data that needs to be exchanged to modules other than the microcontroller due so via the microcontroller. Such as the IMU needs the values that the flash storage has stored, and gets such via a request to the microcontroller. The other sensors module will be an optional expansion bus, which may read any variety of values that the user could want and output those alongside the location values.

2.5 Quality of House Analysis

2.5.1 Overview

The House of Quality (HOQ) is intended to compare and contrast, and then further analyze, different important engineering and marketing requirements of the NAT device. The HOQ is part of a product development process, widely used in industry. It is a sort of conceptual map that allows inter-functional planning and communications, meaning it allows for the different teams to work together to determine which types of goals are important, and to achieve them what other goals would be effected.

2.5.2 Understanding/Reading the House of Quality

In our HOQ, we have 5 major marketing requirements (shown down the left side of the chart), and 7 engineering requirements (shown across the top). These are compared against each other, and in the case of the engineering requirements against itself. These comparisons, shown in the center part of the chart, compare what happens to the marketing requirement if the engineering requirement were to either increase (shown by a '+' underneath the title) or decrease (shown by an '-'). The comparisons of the engineering requirements against the other engineering requirements are shown below this middle section of the chart, in a sort of triangular figure. These comparisons are indicated by arrows.

These arrows shown in the chart indicate the result of giving the best available to the technical/engineering requirement in regards to the marketing requirement, for example what would happen to the marketing requirement 'small size' should the engineering group decide to increase the engineering/technical requirement 'Battery Life' to the largest possible option. In this scenario, the '↓' indicates that by increasing the battery life to as large as possible, the engineering group will have to compromise on the marketing requirement small size. Therefore, the '↓' indicates that the two requirements have an inverse, or opposite, effect on each other. Making one better will make the other worse. Otherwise, a '↑' indicates that the two requirements have a direct effect on each other, making the one the best will also improve the other.

There are two different indications for the arrows, a single arrow and a double arrow. The single arrow indicates that the effect these engineering/technical requirements have on the marketing requirements are small, thus moving one to a better or lower quality will not affect the other requirement significantly. In our example, this means that increasing the battery life with, for arguments sake, a larger battery will not affect the size significantly but it will affect it. However,

with something like accuracy that sees the double arrow ‘↓↓’ when compared to battery life marketing requirement means that to increase the accuracy will largely and significantly affect the battery life. Hence, when the engineering group looks to see what can afford to be compromised on, they should try and stay within the single arrows.

If the house of quality shows no arrow inside of the comparison box, this indicates that the two requirements pose no real effect on each other.

House of Quality - NextGen Asset Tracker								
		Technical Requirements						
Battery Life		↓	↓	↓	↓	↓	↓	↓
Cost		↓	↓	↓	↓	↓	↓	↓
Accuracy				↑	↑	↑	↑	↑
Reliable & Error Free Data link RPMA								
Reliable & Error Free Data Link Cell Network								
Rotational Vertical & horizontal Position								
Compass								
		Battery life	Cost	Accuracy	Reliable & Error Free Data link RPMA	Reliable & Error Free Data Link Cell Network	Rotational Vertical & Horizontal Position	Compass
		+	-	+	+	+	+	+
Marketing Requirement	Low Cost	-	↓	↑	↓	↓↓	↓↓	↓↓
	Long Battery Life	+	↑	↓	↓↓	↓↓	↓↓	↓↓
	Small Size	-	↓	↓				
	Acceptable Accuracy	+	↓↓	↓	↑	↑	↑	↑
	Real Time	+	↓↓	↓	↓			
		> 3 Months	<\$40	<3 Meters	Pass/Fail	Pass/Fail	<10 degrees	<10 degrees

Figure 8 - NAT House of Quality

2.5.3 Engineering/Technical Requirement Comparisons

The sections below detail the full details on what Figure 8 is stating regarding the NAT project, as well as some of the reasoning behind the comparisons. The analysis is organized by looking at

each engineering/technical requirement comparison against the rest of the house of quality requirements. After each individual section is discussed and analyzed, the full overall analysis of the house of quality will then be discussed as a whole, as well as what the engineering group should take from it in regards to the NAT project research, design, and overall production.

2.5.3.1 Engineering/Technical Requirement: Battery Life

Discussing battery life against the marketing requirements first, it will be analyzed what will happen if we get the battery life to be as long as possible. If we increase battery life, theoretically the cost will have to increase, as the stronger, longer lasting batteries are naturally more expensive. Ignoring the comparison against itself, which occurs because long battery life is both an engineering and marketing requirement, with an increase in battery life would cause an increase in size. Larger batteries require more space, and our device needs to be small. Thus, we have to give up some battery life for a smaller size, the issue is going to be finding the balance between a good battery and a size of the PCB. Moving on to accuracy, more accuracy requires stronger components or more samples per time frame and both of those drain more battery to run. Therefore, to have a longer battery life you would have to reduce some of the accuracy of the module. The final marketing requirement is that the system should be real time, meaning that our system reads GPS and IMU data and reports it to the user as it happens in “real-time”. However, real-time reporting costs many things, including power. To report things as they happen requires a lot of power, especially because of the addition to powering the communications devices.

Comparing battery life against the other engineering requirements, except the ones that are also marketing requirements. With a longer battery life, every other engineering requirement has to decrease. This is because anything that performs to the highest level, such as RPMA communication, Cell Network, or highest accuracy vertical and horizontal position and compass from the IMU, drains the battery level. Therefore, to have a longer battery, we have to reduce the performance of these qualities.

Battery life will also be discussed in all of the other sections, as it is a very important feature of the device. Therefore, continue through this analysis of the House of Quality to see more discussion of how the battery life effects other requirements.

Overall, battery life is affected greatly by the other requirements. Generally, to have a longer battery life all of the other requirements would have to be compromised on. Therefore, to have a slightly lower battery life will allow for many of the other requirements to see benefits.

2.5.3.2 Engineering/Technical Requirement: Cost of Product

The next engineering/technical requirement is the cost of the product. Our requirements show that we want the product to be less than 40 dollars per unit. Achieving this requirement, we would need to find cheaper components for some of the modules that we are implementing, as well as eventually buying the components in bulk will reduce the price of some of the components overall. Therefore, because of the possibility of saving cost by buying less powerful components, yet still components that work for our design, most of the marketing requirements would be lowered. At a lower cost, the device would not have as long of a battery life. Better batteries with longer life cost more would possibly be a larger size because the better modules which have smaller sizes, would

be more expensive. The same argument is why the acceptable accuracy and real time also have to lose performance for the cost to be reduced. However, it is also seen that these are not of a particularly significant effect on the cost. While all see an inverse effect, they are also all single arrows. Meaning that compromising on the low cost will not necessarily benefit our product enough to be where the greatest change could be made, overall.

The discussion then moves on to comparing a lower cost against the other engineering/technical requirements. A lower cost will likely significantly reduce the battery life. First, the engineering team would have to compromise on a larger battery. A larger battery would extend battery life due to simply just increasing how much power that could be stored. Larger batteries, however, cost more money. Therefore, a smaller battery would be utilized to reduce costs. Also, the components that would possibly have low power consumption are likely to be more expensive than the ones without, as features cost money. Thus, we would get cheaper components by compromising on the battery life of the device.

Next, continuing on through comparison between the engineering/technical requirements, when it comes to the requirement of accuracy a lower cost would greatly reduce the accuracy of the device for a couple of reasons. One being that with a lower cost the team, again, has to compromise on the better, more accurate components as they likely cost more than less accurate, possibly older components. This means that for things like GPS or IMU modules, where one may give <1 m of accuracy, which would be a great feature for our NAT device, there could be an older, cheaper one that gives us <3 m of accuracy. Since the older, cheaper one still meets our devices requirements, we could compromise on accuracy for cost. Another component of accuracy is that a more accurate reading to the user means that the device has to send more updates on location more frequently, thus increasing cost because of requiring devices that can read more samples per second, as well as having to send more messages over the radio modules increasing cost through paying for those messages (such as through cellular data plans). Thus, compromising on accuracy that way by not sending as frequent of location updates would reduce our overall cost as well.

Having a reliable and error free data link for both the RPMA and the Cellular networks requires more expensive radio modules, as they can guarantee a more reliable connection. Therefore, to save on costs to bring the overall cost per unit down, the group would have to compromise on the radio modules that guarantee a more reliable and error free connection, which means that getting a lower overall cost has an inverse effect on both the RPMA and the Cellular network connections.

Having a cheaper overall product, again will have to make compromises on the components in the device, including the accuracy of the rotational vertical and horizontal position of the 9-axis IMU as well the compass reading of the 9-axis IMU. This is simply because better, more accurate IMU devices (such as ones that give <2 degrees of accuracy instead of <10) will cost more money, as they required more technology and are generally newer than the ones with less fine-tuned accuracy. Thus, to reduce cost, the inverse effect on the rotational vertical and horizontal position as well as on the compass readings.

Overall, having a lower cost will at least minorly decrease every other requirement that this product has. Thus, when looking for places to compromise, as far as the team can handle in their budget, it is recommended that this not be the requirement that others compromise for.

2.5.3.3 Engineering/Technical Requirement: Accuracy

Accuracy is the next engineering/technical requirement that the House of Quality investigates against the marketing requirements. Accuracy is effected in a few different ways. The first being that the modules that provide location data do not have to do so continuously. Instead, they can report back on the location of the device only occasionally. Be that once every minute or once every year, our device can be programmed to accommodate numerous options that the user can specify exactly the type of accuracy they want/need. Another way that accuracy of the device is effected is by implementing a module that guarantees more accurate location data. These modules are expected to be generally more expensive, both in time and in power consumption of the module. The project has a significant specification that the device has to have high accuracy, thus to achieve the highest accuracy it will affect the marketing requirements either inversely or directly. Starting with the cost of the device, we have already specified that to get a module that has greater accuracy we expect it to also be more expensive, therefore having an inverse effect on overall cost of the module. This will also be due to the fact that to be more accurate, more messages are going to have to be sent over the radio modules which costs more money. An increase in accuracy will also cause a decrease in battery life for similar reasons. To have a more accurate reading the module will have to run more often, thus higher power consumption, as well as have to utilize the radio modules more frequently. This will cause the battery to drain faster, and therefore have a lower battery life. The accuracy of the module should truly have no real effect on the overall size of the device. Ignoring the comparison with its equivalent marketing requirement, an increase in accuracy may decrease the ability to be real time.

Continuing the comparison of the engineering / technical requirement accuracy against the other engineering / technical requirements that do not have equivalent marketing requirements. When attempting to get greater accuracy, a direct effect on a reliable and error free data link for both RPMA and cellular communication technologies is assumed to be possible, since more accuracy would cause the radio communication modules to move to low power mode less frequently, thus retaining a strong working connection. When working with INS based location services, we need more accurate IMU module readings. Therefore, with more overall accuracy we need more accurate IMU modules, allowing for more accurate compass and rotational positions.

Overall, more accuracy benefits the technical/engineering requirements that benefit from having better modules, however to have more accuracy there needs to be a higher cost with a lower battery life.

2.5.3.4 Engineering/Technical Requirement: Reliable and Error Free Data Link Connection over RPMA and Cellular

The next requirements in the House of Quality are the reliable and error free data link connection over RPMA and cellular radio modules. Both of these requirements will be analyzed together, since both effect the other requirements in similar ways for similar reasonings.

The initial analyzation will be between the engineering/technical requirements reliable and error free data link connection for both RPMA and Cellular technologies and the various marking requirements, the first being low cost. As discussed in other parts of this analysis, to achieve a more reliable and error free communication connection, better more reliable radio modules are

going to have to be used. These better modules cost more money than their less reliable counterparts, and therefore to achieve a more reliable connection, a higher cost will have to be incurred. This is why, in the House of Quality, the reliable and error free data link connections have such a significant inverse effect on the low-cost marketing requirement of the NAT device.

The next comparison that needs to be made is with the long battery life marketing requirement. To achieve a longer battery life, components will both have to be both low power consumption modules, as well as allow for significant periods of time where they are basically powered down, consuming less power. However, these are likely to be more prone to less reliability, as they are constantly moving from high power states to low power states while messages are moving between high bandwidth and low to no bandwidth usage, as well as more errors, as we are moving between a solid connection and no connection. Also, more reliable connections are harder in areas where the modules are struggling to even send messages. This means that a stronger, more power consuming module would have to be utilized to achieve a constant reliable and error free data link connection, and therefore this has a significantly inverse effect on the battery life of the unit.

Moving onto the size of the unit, there is unlikely why a reliable and error free data connection should affect the size of the overall module at all. Therefore, these components have no effect on each other and the section indicating this comparison is blank.

Acceptable accuracy is the next marketing requirement that gets compared to the reliable and error free data link connection engineering/technical requirement. With a more reliable and error free connection, the accuracy of the data coming through will be stronger, and therefore they have a positive, direct correlation on each other. Therefore, by obtaining and utilizing modules that give the device a more reliable and error free data link connection, the team will also likely improve the overall accuracy of the device, thus positively impacting two product requirements.

For the same reasoning behind the comparison between reliable and error free data link connection and small size, there is also a blank comparison between reliable and error free data link connection and real time. The device must be real-time, meaning that a user can open some application and see immediately where their devices are. A more reliable connection does not make the device any more or less real time. Therefore, the comparison is blank on the house of quality.

Moving onto the top of the house of quality where the reliable and error free data link connection is compared to the other engineering/technical requirements. The first look is at battery life and cost, both of which have already been discussed due to also being marketing requirements, so these will not be analyzed again. The same goes for accuracy. Thus, the only comparison that needs to be analyzed and discussed is that between reliable and error free data link connection and the IMU values, rotational vertical and horizontal position as well as the compass values. However, by looking at the house of quality it is shown that those values do not have any effect on each other. Improving the radio modules does not have any effect on the quality of the IMU module. Therefore, the comparison here is blank.

Overall, by improving and ensuring that the radio modules have a reliable and error free data link connection the team will have to compromise on cost and battery life significantly, however otherwise it will help the accuracy of the device in the process.

2.5.3.5 Engineering/Technical Requirement: Rotational Vertical & Horizontal Position and Compass

The rotational vertical and horizontal position accuracy comes from having an accurate IMU device. The same effect is related to having compass accuracy. The IMU controls both of these values. The team needs to get an IMU with great accuracy on these values, so that they output the proper value within 10 degrees of the precise calculation. How this effects the other requirements will be discussed here.

Beginning with a comparison between the engineering/technical requirements of the IMU and the marketing requirement of low cost, the house of quality shows that this has a significant inverse effect on each other, meaning that increasing one will decrease the other. More accurate readings from the IMU means that the modules will likely have to be newer and, thus, more expensive. Thus, the more accurate the IMU device the higher the cost, comparatively an inverse effect.

The IMU accuracy's effect on the long battery life marketing requirement is very similar to the analysis above. To have a more accurate reading, the IMU will have to be a newer, more expensive, more power consuming model. Also, to get such accurate readings always the IMU will have to run with more samples per second, thus consuming more power overall compared to if the device did not need as accurate IMU readings. Thus, the significant inverse effect the two requirements have on each other.

The size of the device, as well as the device being real time, are not affected by more accurate IMU readings, therefore in the house of quality they are shown as blank spaces.

The final marketing requirement that the IMU accuracy values should be compared to is the acceptable accuracy. Logically, more accurate IMU readings will allow for more accuracy overall. Thus, the positive direct relationship depicted in the house of quality.

For the final time, moving to the top of the house of quality where the engineering/technical requirements should be compared to each other the analysis for the IMU values accuracy requirements are quick. The discussion related to the battery life, cost, and accuracy have already been discussed, and therefore do not need to be repeated here. Then, we again look at the comparison between accurate IMU reading and reliable and error free data link communication networks and see that they have no effect on each other and therefore are not comparable.

Overall, for similar reasons to the reliable and error free data link communication network requirements, the IMU accuracy values have significant effects on the low cost and long battery life requirements for both marketing and engineering/technical. However, they pose no effect on the size and real-time marketing requirements, nor the radio modules reliability engineering/technical requirement. They do, however, pose a slightly positive effect on the overall accuracy of the product.

2.5.4 Final Analysis

The final analysis will be a quick summary of the analysis that was discussed above, as well as a suggestion of how this should be interpreted in regards to the NAT project.

There will have to be a compromise on some of the engineering/technical requirements. From analysis of the House of Quality it is seen that to have a better product in terms of performance, the battery life will have to be lower and the device will have to be more expensive. However, if the team were willing to compromise on some of the devices components performance, the team could achieve a strong device, while still maintaining a lower cost and a longer battery life.

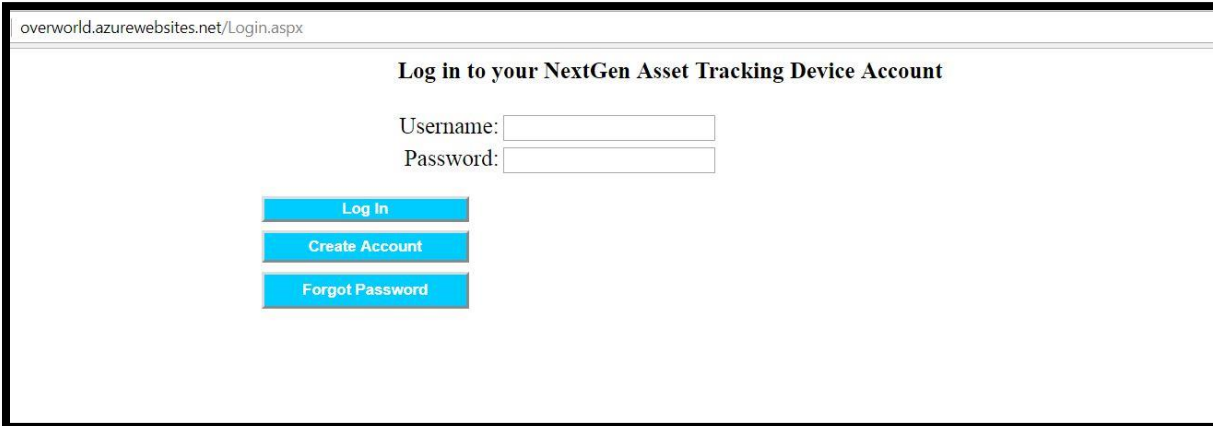
2.6 Project Operation Manual

The NAT device is a portable, small device that will attach to a myriad of objects and items that the user wants to be able to locate at their desire. There are no specific requirements detailing how this attachment should happen, however. At this moment, the final NAT product will be placed inside a simple enclosure. This enclosure can be designed various different ways fairly easily with the utilization of 3D printing. Thus, the actual connection to user devices is customizable to the users' needs. If they want to connect it to a metal box like a shipping container, an enclosure with a strong magnetic base can be designed. If, instead, it will be attached to some wire, a cord that wraps around the wire can be designed instead. This all depends on the users' needs.

The NAT device itself will be enclosed inside the enclosure so as to avoid any malfunction on the PCB, or any hard conditions that may come from exposed electronics, both to the device and to the user.

The device will be initially be set up by the manufacturer and distributor. At the manufacturing point, the department responsible for shipping the device(s) to the end consumer will also program the device according to the users' requirements. This will be done during the ordering process, they will specify on numerous customizable settings varying from what kind of indicator will trigger the GPS to relay the location to the user to which modules will save their values to the SD card. These various settings that the user can configure in the device are explained in detail in the section 5.6.1.2 Configuration GUI. The distributor will also set the devices up on the user's map allowing them to see all of their devices easily.

Steps to Operate System:



overworld.azurewebsites.net/Login.aspx

Log in to your NextGen Asset Tracking Device Account

Username:

Password:

[Log In](#)

[Create Account](#)

[Forgot Password](#)

Figure 9 - NAT Web GUI Login Screen

There are two major options for users, those whose devices get a GPS fix on a motion trigger and those whose devices get a GPS fix on a regular time interval. Both, however, start at the same step, logging into the Web based GPS fix system, as shown in Figure 9 above.

Web backend will verify that the username and password combination are valid. Then, if the login is proper, the user will be directed to their personal homepage where three options will be given, as shown in Figure 10 below. The web backend will store the ID's, Name's, and the current and history of locations of the devices that the user has in their fleet.



Figure 10 - NAT Web GUI Home Page

The client will be able to create a personal account which will contain the current location, along with the previous location history. Once logged in, the user will be prompted to select one of three options, which are to go to “My Profile,” “Add a Device” or “Track a Device.”

Under the profile page, the options to edit username or password are available and a list of the user’s current devices will be displayed. The option to add a device will prompt the user to enter the new device’s serial number, the one-time access code and to nickname the device. The list of current devices will be displayed on this page as well, as shown in Figure 11 below.

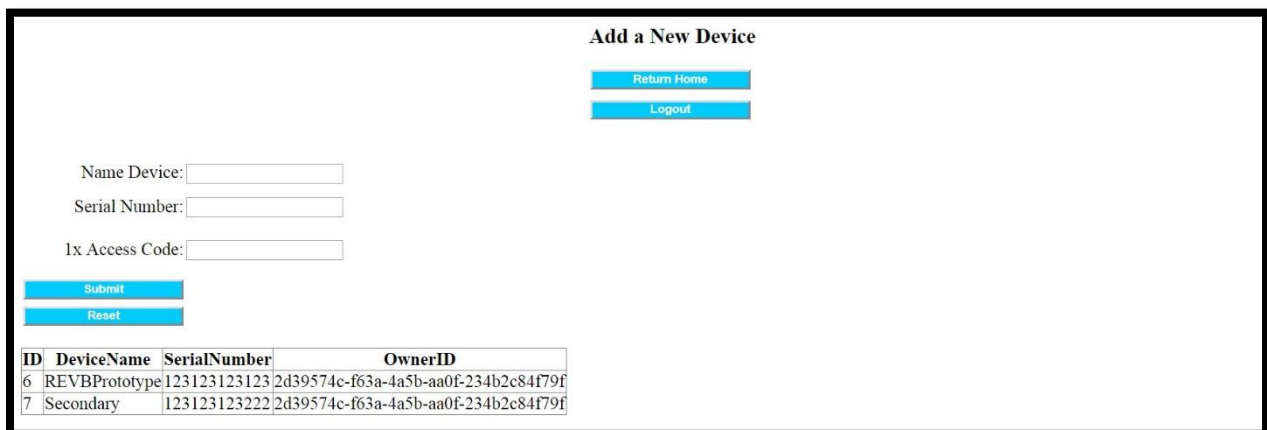


Figure 11 - NAT Web GUI Add New Device

Under the track a device page, as seen in the Figures 12 and 13 below, the client will see the list of current devices with the option to choose one to track. Once selected, that specified device will be shown on the map, along with the latitude and longitude coordinates of its current location and its timestamp. In addition, there will be a list of all the previous locations and timestamps of that device that can be viewed, with the first row being the device’s current location and the others being the previous locations in a time-descending order.

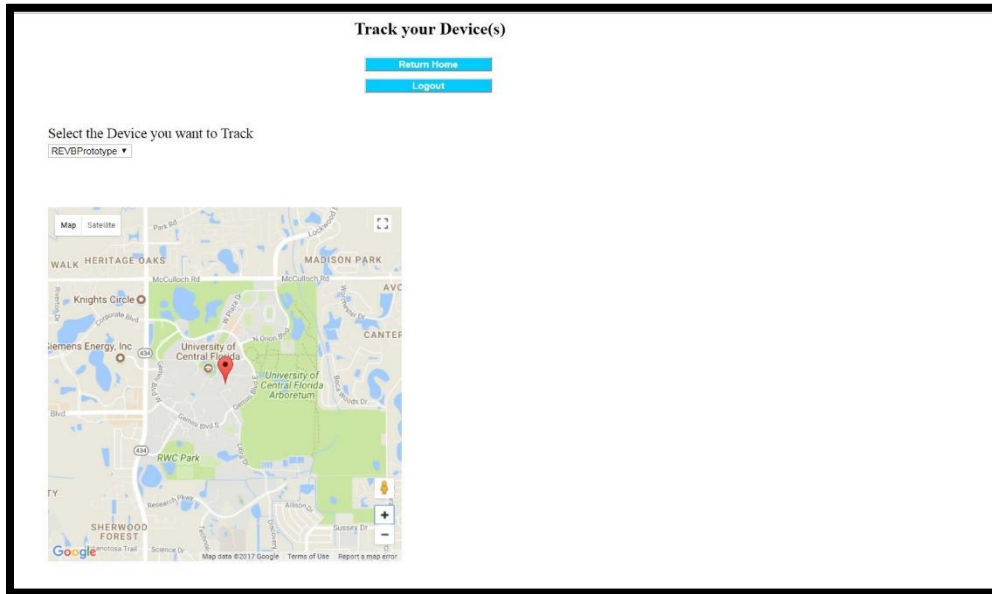


Figure 12 - NAT Web GUI Map

Current and Previous Locations:

	SerialNumber	Latitude	Longitude	Timestamp
Select	123123123123	28.600695	-81.198044	11/28/2017 10:41:48 PM
Select	123123123123	28.600340	-81.197624	11/28/2017 10:36:00 PM
Select	123123123123	28.600548	-81.198692	11/27/2017 5:44:32 AM
Select	123123123123	28.600697	-81.198120	11/27/2017 5:43:14 AM
Select	123123123123	28.602200	-81.316826	11/27/2017 5:14:07 AM
Select	123123123123	28.595335	-81.385361	11/27/2017 4:52:07 AM
Select	123123123123	28.595272	-81.385376	11/27/2017 4:41:39 AM
Select	123123123123	28.595280	-81.385376	11/27/2017 4:40:44 AM
Select	123123123123	28.595215	-81.385345	11/27/2017 4:35:34 AM
Select	123123123123	28.595215	-81.385345	11/27/2017 4:34:44 AM
Select	123123123123	28.595266	-81.385353	11/27/2017 4:28:23 AM
Select	123123123123	28.595266	-81.385353	11/27/2017 4:27:28 AM
Select	123123123123	28.595266	-81.385353	11/27/2017 4:26:36 AM
Select	123123123123	28.595266	-81.385353	11/27/2017 4:25:40 AM
Select	123123123123	28.595266	-81.385353	11/27/2017 4:24:46 AM
Select	123123123123	28.595266	-81.385353	11/27/2017 4:23:51 AM
Select	123123123123	28.595266	-81.385353	11/27/2017 4:22:59 AM
Select	123123123123	28.595173	-81.385254	11/27/2017 4:18:43 AM
Select	123123123123	28.595398	-81.385441	11/27/2017 3:43:55 AM

Figure 13 - NAT Web GUI Table of Locations

There is the main database for the web GUI which will hold the account information for the users, along with their registered devices. It will also have a separate table used for holding the latitude and longitude coordinates of each device. Once a location is found, the coordinates are added to the list of location history for that device that is viewable to the user on the tracking page when that device is selected.

2.7 Project Use Cases

This section will detail some use cases that some of our customers using our devices may see. A use case being a typical scenario of the use of our product.

2.7.1 General Simple User

A general simple user is a regular person that decides to purchase and use our device for their individual personal use. Cases that this user may see are detailed as follows.

2.7.1.1 Attached to Keys/Wallet/Keys/Wallet/Small Portable Object

The user can attach our small, lightweight device to their keyring, wallet, or any other small portable object that may be easily lost. The user goes about their day without really noticing that the device is with them, attached to said device. The user somehow misplaces/loses/has their device stolen without noticing immediately and not being able to simply pick their small portable object. The user logs in with their device credentials online. The user then sees where their device is and takes the appropriate actions.

2.7.1.1 Placed in a larger object such as a vehicle

The user can leave our small, lightweight object in something like their car, another smaller vehicle like a motorcycle or ATV, or something larger like a mobile home or RV. The user goes about their day. Option 1: they lose their vehicle somewhere like a parking lot. The user opens up their phone and logs into the website using their credentials. They see where their vehicle is within 3 meters and go to it. Option 2: their vehicle gets stolen. They go online with their credentials and see within 3 meters where their device is. They notify the authorities. Option 3: they are trying to locate a person, who's vehicle they have left/attached one of our devices, such as a child. They want to find said person so they open the website online, login with their credentials, and see within 3 meters where their person is.

2.7.2 Company User

A company user is an organization that decides to utilize our device within/on their product(s).

2.7.2.1 Company with Device in Frequent Motion

Devices in frequent motion can utilize our Motion Trigger location option. Applications, among many others, that may wish to utilize this feature are:

- Packages that have an item of significance in transit.
- Vehicles in their fleet
- Animals in a reserve

With applications such as these the use case may be as follows. The user has an item of which they may at a later time need to locate and would have a hard time either finding this particular object, or keeping track of the many objects in their fleet would be too labor intensive. Therefore, when they deploy a new object they attach our device to their application, as it is small and lightweight, and when they need to find/track this object they simply login to the website and see where all of their fleet currently is. They locate the one of interest and see exactly on a map where that item is within 3 meters.

2.7.2.2 Company with Device Mostly Stationary

There are also devices that a company or organization may wish to locate that are ultimately unmoving. However, the time and labor required to keep track of all of these items may be not worth the effort. The applications that may utilize the location frequency feature instead of the motion trigger feature are:

- Shipping containers in a shipping yard
- Electrical wires (where our future external sensors may be desirable)
- Airport/Hospital equipment
- Oil Field equipment
- Military equipment
- Computer Equipment

With applications such as these the use case may be as follows. The user has an item of which at a later time they may need to locate and would have a hard time either finding this particular object, or keeping track of the many objects in their fleet would be too labor intensive. Therefore, when they deploy a new object they would attach our device to their application, as it is small and lightweight, and when they need to find/track this object they simply login to the website and see where all of their fleet currently is. They locate one of interest, and see exactly where on the map that item is within 3 meters.

Items like airport or hospital equipment is always on the move, being shuffled around the hospital and airport where it is immediately needed and being discarded when the user's attention is diverted elsewhere. This causes many items like hospital and luggage carts to be misplaced all over the building(s) they are being used in. With our device, these items could be located more quickly.

Oil field equipment stays stationary in the field for so long that should someone need to locate that machine for maintenance or the like it would be very difficult to find without some intense manual mapping system of their own. With our device, it would be a simple login and search to see where the device is now.

3.0 Research Related to Project Definition

Research is a very important part in any engineering project. This includes researching similar products, and in our case similar Senior Design projects so as to compare what makes our project different from what already exists. In addition to which projects already exist, research has to be done into possible technologies to implement in our device. This includes communication technologies, such as LTE, NB-IoT, and RPMA among others. There is also research into what types of modules and components are appropriate to what the device needs to accomplish. This includes exploring different modules and determining which one will sufficiently meet our goals while remaining in our budget. The final section of research that must be done is investigating which types of architectures, both in our hardware and our software, that we may like to implement in our device. The research that we have done into these subjects are detailed in this chapter, which aided in the final selection of our parts described at the end.

3.1 Existing Similar Projects and Products

The first step in research for our device is detailed here, similar projects and products. This research should aid in determining many factors, such as what specifications our project should have, what can differentiate our project from others, and what technologies we should look into implementing in our device.

3.1.1 Similar Projects

In this section, we will discuss other projects that we have investigated and analyzed that are in a similar category to ours. These following projects are based around or include and significantly utilize a GPS or similar tracking system. Our fellow UCF classmates have paved the way with some good ideas and implementations in previous projects. A few of them will be highlighted in the following section.

3.1.1.1 TrackAlert

The Tracker alert is a previous senior design project intended for the safety of someone incapable of caring for themselves. The project incorporated four separate systems to notify a user of the whereabouts of the individual being tracked. The four systems are RFID, GPS, RF, and a web application used to relay critical information to the end user who can then take the necessary action to both find and help the distressed person. Communication with the end user is achieved with a web application utilizing Google's App Engine to enable any device capable of accessing the internet to access the location of the tracking device and the serial number of the device and name associated with it. If the GPS device has left a designated area, the RFID will alert the base station of his and the emergency tracking system will be initiated. While the RFID is alerting the base station of the breach of parameter, it is also alerting the end user through the web application. The GPS coordinates of the arm band are transmitted to the base station through the same RFID technology, and then sent to the web application through Ethernet, allowing the end user to keep receiving updates on the location of the device and the individual wearing it.

The RFID subsystem of the TrackAlert emergency system consists of an RFID reader, passive tag, stick antenna, and an RF transmitter. The microcontroller used for this system was an MSP430F2370 that was selected by the project group for its low power characteristics. The purpose of the RFID tag and reader is to determine when the GPS tracking device has left the parameter where the tag would be on the GPS and the parameter would have the reader to know when the tag left the area.

The GPS subsystem is made up of a transmitter module, a rechargeable battery, a liquid crystal display, a microcontroller, the X-Bee, and the GPS chip itself. In this system, the GPS determines the actual location of and sends this information to the microcontroller for processing. The X-Bee will then take this data and transfer it to the base station.

The role that the base station plays in this system is that it act as a central hub for information to be routed. The base station wirelessly transmits data between the RFID reader, the web app and the GPS system. The base station only receives data from the RFID reader then commands the GPS module to transmit coordinates. Once the location of the GPS is received by the base station, it is at this point it will send the necessary information to the web application for the end user to take action. The communication between the web application and the base station takes place using an ethernet cable.

The final piece to the puzzle of this system is the web application which consists of a user interface and the Google App Engine. The designers of this project wanted to go for database availability and portability which is why a web application was chosen to handle the task. They also considered security in their implementation and had the web app request a user ID and password for logging into the account.

3.1.1.2 Helmet Tracking System



Figure 14 - Helmet Tracking System

*Permission Requested [P15]

The Helmet Tracking System is a smart helmet system for the large community of motorcycle riders in the world. Its prototype can be seen in the image above. The designers involved in the creation of this project were focused on building something that will improve the experience the millions of people who ride motorcycles today. The design group identified a good amount of problems that a motorcycle rider may encounter on his or her daily ride, and implemented a collection of subsystem aimed at fixing the issues. One such subsystem is a Bluetooth module that will allow the rider to interact with a smartphone to gain access to phone calls and music all while riding the bike. Where the tracking system of the project comes into play is where an accelerometer is utilized. The accelerometer will behave as a type of impact sensor detecting sudden changes in velocity in the event of an accident. The module will send signals to the microcontroller, waking it from a low power mode, to retrieve the current latitude and longitude location from the GPS module. This location data will be sent to a user pre-selected emergency contact via SMS. Once the emergency contact receives the message alert of a possible accident, they will be able to try and contact the motorist and/or medical help after assessing the situation.

Accident as a whole can be dangerous and even deadly. However, considering the fact that motorcyclists are a little less protected on a bike than a driver is in a car, they may prove to be a little more dangerous and deadly. It is therefore important that if an accident does occur, the rider can get the medical response they need in time. To detect possible accidents, the Helmet Tracking System employs an accelerometer. This is a special sensor that can detect acceleration and deceleration in any axis. Usually in a crash, whether it be car or motorcycle, the impact will cause sudden high deceleration. Using this sensor in the safety system, initiates the first step of getting help for someone in need by first detecting that there is a problem.

After a potential accident is detected, the system will need to get the location of the rider to send to the emergency contact. Knowing the location will require a GPS system that can precisely acquire the coordinates of the device. For this project, the designers decided to keep track of the device's location at all times. The benefit of this is that, once the device gets an initial location, any other update will be considered a hot start. Hot start times for GPS modules are faster than cold start times for the obvious reason that the system already has previously known locational data as opposed to cold start having to acquire new data. This is good because it is critical that the individual or individuals in danger get the help they need in a timely fashion. The downside to this method is the power consumption. Generally, when a device is awake and processing data it will consume more power than when is in an idle state. Having the GPS updating the location all the time might take a toll on the battery, causing the device to have a shorter operation time.

Another key component of the Helmet Tracking System is its GSM module. This piece of hardware enables the system to have access to the mobile communication network. By the use of this module, the system can utilize the cellular network to send an emergency SMS to the contact of choice to get help. This module is similar to our cellular phones in the fact that it also needs a SIM card. SMS was the method of choice for relaying an emergency message because it uses standardized communication protocols that allow the exchange of short text messages between a fixed or mobile phone line. The module can send up to 150 characters which is sufficient for a message to alert someone.

The Bluetooth module implemented in the system is not as critical but can enhance the experience for the motorcycle rider. The module allows for synchronization to the two main types of cell

phones today in Android and iPhone. This can actually be thought of as a safety feature by considering the hands-free capabilities achieved by using the Bluetooth module to carry out a phone call. This way the rider will be able to put more focus into riding his bike rather than holding a cell phone. Not to mention listening to some of your favorite tunes via Bluetooth, which can bring enjoyability to any moment.

3.1.2 Similar Products

Following the similar projects, there are also products similar to ours that are already on the market. Within our research we have found 4 that are worth describing and analyzing below, which the comparison chart was already shown in the introduction of the paper (Figure 3). The benefit of viewing and documenting these products are that we can tell if our product needs other features or if there is a market that we are not meeting that we could include with some simple technology implementation.

3.1.2.1 XT-2000 OBD Real-Time GPS Vehicle Tracker



Figure 15 - XT-2000

*Permission Requested [P7]

If a vehicle is the only asset or object that needs to be tracked, the XT-2000 OBD Real-Time GPS Vehicle Tracker is a great little device. It provides instant vehicle tracking that can be observed from anywhere using a computer, smartphone, or tablet, through the platform provided by SpyTec. The XT-2000 relays accurate GPS data and route information that can be viewed on Google Maps or Google Earth.

Installation of the XT-2000 takes a few seconds. It is inserted directly into any modern vehicle's on-board diagnostic port (OBD) under the dashboard. This setup is very convenient because the OBD port has a connection to the car's battery, so the device should continue to work as long as the car battery is functional. Another convenient thing about the XT-2000 is that if there is no time to monitor the map but the user needs to stay informed, the device can generate alerts via text messaging or email. The alerts can notify someone about things such as if a certain speed is reached, if the car starts or stops moving, or when a certain location is reached. Geo-Fence technology can allow users to set up a perimeter anywhere on the map and get notification whenever the vehicle enters or leaves the area.

The XT-2000 is capable of tracking a little more than just GPS data. It also provides information on fuel efficiency, mileage, and maintenance needs so you can help extend the life of your vehicle. Another benefit is engine monitoring that lets you see when the engine is on, off or idling. If interested in if a car is being operated safely, the device measures braking force and acceleration. It will alert you if the car speeds up suddenly or stops suddenly. So, if someone other than you take the car for a drive, it is easy to see if they're being responsible drivers.

At only 25 dollars a month, the cost for subscription of the XT-2000 is very affordable. Also, because there are no contracts to sign and no cancellation or activation fees, you can start and stop using the tracker whenever the need arises.

3.1.2.2 STI_Bolt Asset Tracker



Figure 16 - STI_Bolt

*Permission Requested [P5]

The STI_Bolt Asset Tracker allows you track an asset over the internet. You can use this device to keep track of valuable pieces of equipment, special orders or shipments, or just your car. It is built for long-term tracking, with the battery lasting up to 9 months.

The device also comes with a magnetic mount and is small and stealthy, so you can use the STI_Bolt Asset Tracker to monitor the location of the things that are of importance to you without bringing too much attention to yourself.

The STI_Bolt Asset Tracker is easy to use and just takes downloading the app on your Apple iOS or Android device for you to have instant access to all of your tracking data. The tracking information may also be accessed on a computer or smartphone using the SpyTec GPS website. Notifications can also be sent to you on your smartphone or computer by SMS or email as soon as the device logs a new location. The device updates its position twice per day and you don't have to keep checking the app for updates. Just keep your phone or computer handy and the STI_Bolt Asset Tracker does the rest.

The STI_Bolt Asset Tracker can go virtually anywhere, so there are no limitations on how you use the device. It is less than 6 inches long, so it will only occupy a small amount of space and it includes a case which features a magnetic mount that easily attaches to the bottom of your car or a piece of equipment.

Since the battery on the device lasts up to 9 months, it is ideal for helping you track objects for long periods of time. The STI_Bolt Asset Tracker only need to be charged 2 times a year. The device is even equipped with something to alert you when the battery is running low, so action can be taken to charge the battery before it is too late.

The \$14.95 monthly subscription of the STI_Bolt Asset Tracker is lower than other GPS trackers. The people at SpyTec also assures that there are no hidden fees and you don't have to sign any contracts or pay an activation fees when you start using the device. You also won't be locked into a long-term commitment, so canceling the subscription can be done at any time.

Overall, this tracker is a solid device with many of the features that we hope to implement in our own project.

3.1.2.3 GX350 Real-Time GPS Tracker



Figure 17 - GX350

*Permission Requested [P1]

The GX350 Real-Time GPS Tracker lets you track people or objects over the internet in real-time. Being the successor of the STI_GL300 GPS Tracker, the GX350 features a longer battery life of up to two and a half weeks and according to SpyTec, better service coverage with less dead zones. Compared to the STI_GL300, the GX350 GPS Tracker also offers better service coverage and reliability. Certain areas on the map can be dead zones, in which finding an Internet connection can be almost impossible. Sometimes when crossing these area with a GPS you can lose connection, but with this device you don't have to worry about losing a connection. The device will update its location online even in hard-to-reach rural areas, so you don't have to worry about losing track of the device. This device can be used to keep an eye on the location of a company car, a loved one, or a valuable shipment.

All of the features you would expect to find on a quality GPS tracking device can be found on The GX350 GPS Tracker. You can access all of your tracking data online using a smartphone, tablet or computer just by logging onto the SpyTec GPS tracking website and entering your password. As long as there is an internet connection available, you'll be able to see where your target is in

real-time. The device can also be programmed to send you updates on your smartphone, tablet or computer when your target leaves a certain area thanks to geo-fencing technology. Geo-Fencing allows you to draw a boundary on the map and as soon as your target leaves that area, a message or an email will pop up on your smartphone, tablet, or computer. You can be sure that a notification will be sent to you as soon as your target reaches its destination or if your target wanders too far from a designated area. In addition to all of this, there is an ultra-low power sleep mode feature for this device. This is important because if the device is not sending or receiving data, or not attending to any important tasks, a sleep mode is perfect to help reverse some of the computing power and in turn save some battery power.

Like most of the GPS tracking devices, there is monthly fee to pay to use the service. There are no contracts to sign and no startup or cancellation fees for the subscription of the GPS services.

3.1.2.4 STI GL300 Real-Time GPS Tracker



Figure 18 - STI GL300

*Permission Requested [P6]

The STI GL300 Real-Time GPS Tracker monitors location with pinpoint accuracy and goes anywhere discreetly while giving you real-time updates. It is ideal for things such as guarding expensive items against theft, or monitoring your delivery drivers' routes. The GL300 is very compact measuring only 2 inches, so can fit easily into a pocket, purse or bag. The device also works great for tracking vehicles with the addition of a magnetic, waterproof case (sold separately) that can be placed underneath a car where it will not be visible.

Once the device is in place on the target, whether it be a person, vehicle or property, its location and movement can be tracked in real time from anywhere on a phone, tablet or computer. The SpyTec provides users with a web based GPS tracking application that lets you monitor the device with ease. The GPS website comes equipped with the feature of giving users the ability to set up alerts so they'll be notified by text or email when the tracker arrives at a certain location. This is possible through the tracker's geo-fence function that allows you to set up a perimeter on the map and get notified anytime the target enters or leaves the perimeter. In addition to this, there is a view route option on Google Maps or Google Earth. The accuracy of the location will be within 15 feet of where the tracker places the location.

Another feature of the STI GL300 Real-Time GPS Tracker is that it is capable of storing up to a year's worth of tracking data from its history and can even provide custom reports summarizing the data that has been gathered. The device can run on a single battery charge for up to two weeks, but with additional purchase, a 6-month extended battery pack is also available when the application requires the device to go much longer than two weeks on a single battery.

3.2 Relevant Technologies

3.2.1 Radio Frequency Wireless Communication Technology

The NAT requires communication between the device and the user via a radio module. There are many of those types of wireless communication technologies that are relevant to our project. The following sections discuss each of the technologies that was considered.

3.2.1.1 LPWA Wireless Communications

A Low Power Wide Area network or LPWAN is the ideal communication platform for embedded devices on the Internet of things (IoT). Especially for Machine-to-Machine (M2M) applications such as electronic metering or wireless data collection where the battery life of the device can determine how long it is deployed. As the name suggests LPWA communication is characterized by a wide range of effective communication, a low bit rate, and a scheme of periodic device pinging instead of constant connection to the network. The main advantage of having effective and reliable communication over a wide area is the reduction in the amount of infrastructure needed to facilitate the network. The cost of infrastructure matters to a consumer because it directly relates to the subscription cost of the network. Under different environmental conditions a LPWAN can reliably communicate over a range of 5 to 50 kilometers which significantly reduces the number of access points needed to cover a city or even a country. The Low Power part of LPWA comes in the low bit rate and network connection scheme. For the majority of IoT applications and M2M communications a very small amount of data needs to be transmitted very infrequently making throughput a non-issue in contrast to battery life which often determines the life cycle of a device. Sending a small amount of data, slowly, means that the device can spend the majority of its time in standby or low power mode. To further this end the LPWAN pings its devices very infrequently depending on the application a device may only transmit its data once a day, once a week or even less frequently. A device operating on a LPWAN can be deployed upwards of 10 to 20 years reducing maintenance costs [K], also devices that transmit very little data can have a simple architecture that is cheap to manufacture, and even cheaper to mass produce. In conclusion, communication over a LPWAN is ideal for IoT and M2M applications because battery life can be significantly increased and the costs of infrastructure, manufacturing, maintenance and transmission can be drastically reduced.

Our device will be used for large scale remote asset tracking requiring very few data transactions at even lower bit rates per day tracking largely stationary objects or objects where real-time tracking is unnecessary which makes it exactly what RPMA is designed for. Furthermore, our device will be deployed in areas where regular maintenance cost prohibitive because of the scale of deployment, the physical location of the device being hard to reach or a combination which will require it to have as long of a battery life as possible.

LPWA fits our project perfectly because we do not require the power of traditional wireless networks such as 3G or 4G cellular LTE. More powerful networks, like the ones used for mobile phones, require constant network connection, high data rates and significant infrastructure to maintain reliable communication, all of which are wasted with our kind of M2M transmissions. However, LPWA comes in many different forms such as Cat M1 LTE, NB-IoT, Sigfox, LoRa, and RPMA so the real challenge for this project is to choose which kind of LPWAN to use.

3.2.1.2 LTE CAT-M1

Most cell phones today operate on a 4G LTE Cat-6 or higher network. LTE Cat-M1 or LTE-M1 is a low-power wide-area (LPWA) air interface used in connecting Internet of Things (IoT) or machine-to-machine (M2M) devices with low to medium data rates. It features longer battery life and extended range inside buildings, compared to other cellular technologies such as 2G, 3G or LTE Cat 1. LTE-M1 is defined in the 3GPP release 13 standard that also defined Narrowband IoT (NB-IoT or LTE Cat NB1). Networks for LTE-M1 will co-exist with 2G, 3G, and 4G mobile networks and benefit from all the security and privacy mobile network features, such as support for user identity confidentiality, entity authentication, confidentiality, data integrity, and mobile equipment identification [ZZ].

3.2.1.2.1 Embedded module vs. modem

When deciding on a LTE-M1 solution, one thing to consider is whether to begin at the module level or at the modem level. The main difference between starting a design from the module level as opposed to the modem level, is that starting from the module level requires a lot more bandwidth in terms of time and money. Purchasing an embedded module will require one to budget for months of testing, and thousands of dollars that will need to be spent on the certification process that has to be gone through with carriers, and the federal government in the target geography in which the end-product will be deployed. Software updates to the end-product will need to go through maintenance releases. Maintenance releases occur whenever there needs to be a hardware or software change to the product. If you started your product at the module level and need to make firmware updates to it at manufacturing or remotely in the field, the certification process need to be repeated. Otherwise if your design began at the modem level, the certifications on the modem will be maintained by the provider while the product is allowed to continuously be modified.

The embedded module has the advantages of size and cost of the hardware over the embedded modem. While the modems are small circuit boards that can either be soldered to a design or be attached as a daughter board through connectors, embedded modules are just small chips that are soldered onto those modem. Also, eliminating the hassle of the certification process comes at a price when talking about the hardware itself. LTE-M1 modules can cost somewhere below \$10. On the other hand, a modem can cost in the \$50 to \$100 range.

During our research and creation of this document we noticed that there are only a few modules on the market right now for LTE-M1, but according to industry professionals that will change in the future as the internet of things becomes more and more popular. Sequans Communication's Monarch is a single-chip LTE Cat M1/NB1 solution designed specifically for narrowband IoT applications, including sensors, wearables, asset tracking, and other low data, low power M2M and IoT devices [FFF]. The product brief for this module highlights the Verizon wireless

certification, their support for narrowband IoT and Cat-M1, the small footprint size of 6.5mm x 8.5mm, the single antenna system architecture, and the reduced Tx power class option. Other products like Altair's ALT1250, Gemalto's EMS31, and u-blox's SARA-R4 series are available on the market with very similar features. All of these modules can be interfaces using UART, SPI, and I²C.

3.2.1.2.2 Suppliers

When it comes to embedded modems, the options become even smaller. There are only two main suppliers available for LTE-M1 modems, and they are Link Labs and Nimbelinek. Both suppliers offer a launch kit or a development kit which can cost anywhere from \$150 to \$1000. Link Labs' launch kit is available directly through their website and Nimbelinek's development board can be attained through their website but through different suppliers like Digi-key. Link Labs launch kit plus sensor board includes:

- LTE-M Communications Expansion Board
- GPS Sensor
- LTE Antenna
- Additional sensor board
- Lithium Battery
- Conductor™ Network Management Platform
- IP67 Enclosure
- Activation and 1 year of Verizon service

The development kit from Nimbelinek is called Skywire and is available from distributors like Digi-Key and Symmetry Electronics. The kit only includes the Skywire modem and a development board that the modem attaches directly on top of.

LTE-M1 is a good cellular solution for IoT and M2M communication. When choosing a module or modem it is important to keep in mind that modules are smaller and cheaper as far as hardware goes, but have to go through a costly and lengthy certification process with the cell carrier and the government. Also remember that LTE-M1 modules and modems can have a maximum Tx power up to 23 dBm which can source a large amount of current, so choosing the appropriate type of battery is very important.

3.2.1.3 Sigfox

Sigfox is a French company that builds LPWA networks focused on IoT and M2M applications using their semi-open standard. Sigfox is a semi-open standard because while a device manufacture can communicate using the protocol of their choosing they must use it on the Sigfox network. The Sigfox LPWAN makes use of the Industrial, Scientific and Medical ISM radio band, 868 MHz in Europe and 902 MHz in the US with a 159 dB link budget. A few features of the Sigfox LPWAN are a one-hop star topology, an ultra-narrow band signal, one of the lowest data rates available and even lower power consumption. The one-hop star topology makes integration and removal of devices from the network simple and protects the whole network from device failure. Since each device is connected to central hub when one device fails it will not bring other devices down with it and the fault is easily tracked which also makes repair easy. However, if the

central hub of the network malfunctions then the entire node fails, and the increased number of connections means that building up the infrastructure of the network is more expensive. On the other hand, the ultra-narrow band signal that is produced from Sigfox technologies easily passes through solid objects and penetrates underground at low power consumption which can reduce the amount of infrastructure required to build up the LPWAN. LPWA is characterized by a low amount of data transmission at low frequency, but how low is that for Sigfox? A subscriber to the Sigfox network is able to transmit 140 uplink messages and 4 downlink messages per day at twelve and eight bytes of payload respectively. For many IoT applications this is orders of magnitude greater than what is required. On average, an IoT service provider will require less than 15 kilobytes of data transmitted once a day [K]. Also, Sigfox offers low noise (aprox. -150 dBm) due to the nature of ultra-narrow band communications [N, O].

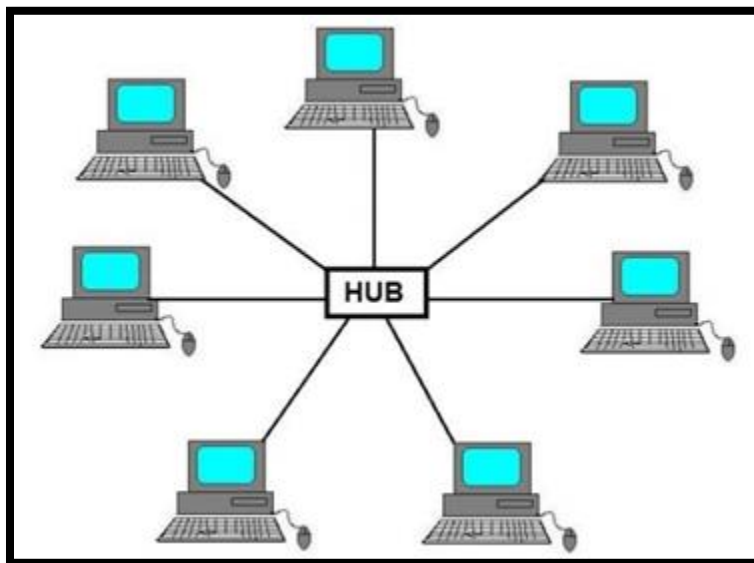


Figure 19 - Star Topology [P]

At the current time Sigfox boasts upwards of 90% coverage of France, Spain and The Netherlands with deployment planned in 9 other European countries. Sigfox is also partnered with high profile tech companies such as TI, Silicon Labs and ON Semiconductor making integration even simpler

In the end, though Sigfox meets all of our requirements for this project it is exclusively used in Europe and our interests are primarily in the US. That being said, world-wide connectivity is an attractive idea so the ideal LPWAN is one that offers the widest possible range of communication.

3.2.1.4 LoRa

LoRa is a **Long Range**, low power wireless platform that is, like all of the above, focused on the IoT and M2M communications. LoRa provides communication ranges comparable to that of cellular providers (15 – 20 km), and in fact LoRa antennas will be able to sit side-by-side, and sometimes be combined, with existing cellular antennas saving on infrastructure and other hardware costs. LoRa has primarily been developed by Semtech, but in an effort to make LoRa an industry standard the LoRa alliance was created at the 2015 Mobile World Congress. The founding

members of the LoRa alliance include Actility, Cisco, Eolane, IBM, Kerlink, IMST, MultiTech, Sagemcom, Semtech and Microchip Technology as well as some of the largest telecom providers Bouygues Telecom, KPN, SingTel, Proximus, Swisscom and FastNet. The LoRa alliance is an open, non-profit association created in an effort to standardize LPWA networks for the IoT and M2M communications. The main aim of LoRa is to integrate into smart city and industrial applications such as: smart cars, street lights, manufacturing equipment home appliances, wearables, etc. Like RPMA, LoRa makes use of the worldwide unlicensed frequency bands but LoRa uses much lower frequencies than RPMA (868 MHz in Europe, 915 MHz in North America and 430 MHz in Asia) in an effort to increase coverage, especially inside buildings. Also similar to RPMA, LoRa uses a spread spectrum modulation scheme based on linear frequency modulated pulses or chirps which helps keep LoRa transmissions very low noise. LoRa is laid out in a star-of-stars topology using gateways as transparent bridges to relay messages between end devices and the central network while the end devices themselves use a bi-directional single hop to communicate with the gateways. Due to LoRa's spread spectrum communication scheme messages at different data rates do not interfere with one another and create sets of virtual channels that increase gateway capacity. All of this is done while keeping data rates small (0.3 – 50 kbps) to maximize battery life and efficiency, and to that same end LoRa uses an adaptive data rate (ADR) scheme to manage the RF communications through the Lora WAN server [Q, R].

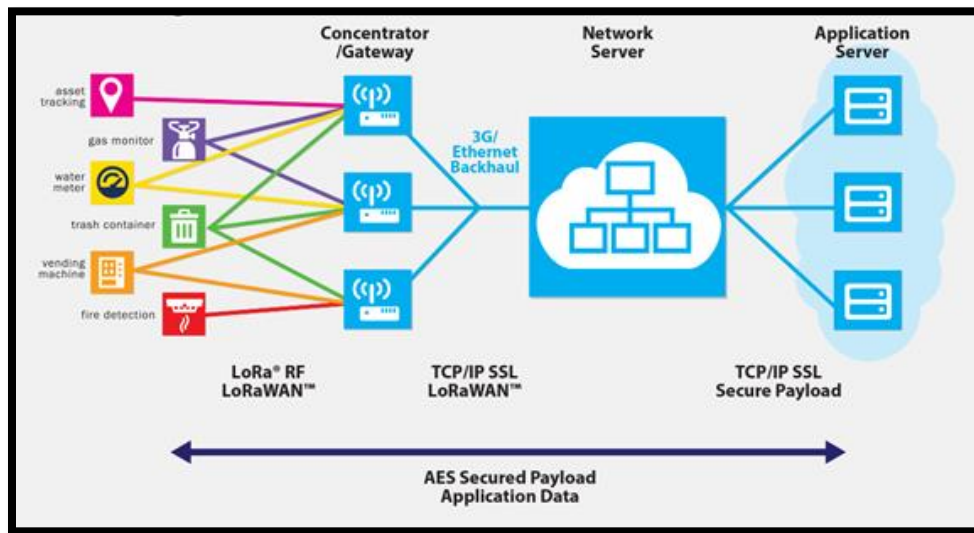


Figure 20 - Lora Topology [Q]

*Permission Requested [P21]

3.2.1.5 RPMA

Random Phase Multiple Access is a LPWA network used exclusively for machine to machine communication on the Internet of Things. RPMA runs on the unlicensed 2.4 GHz band through Direct-Sequence Spread Spectrum with multiple accesses and boasts a 172 dB link budget. Two entities are involved in LPWA communication the WAN carrier provider, a local cellular provider that owns, operates and maintains the infrastructure behind the wireless communication (tower rental, construction and HR), and the device manufacturer that ensures that their device runs at the lowest data rate possible and sends as little data as possible to maintain battery life. Ingenu is the

latter, they have built out an RPMA network that covers most of central Texas, with multiple companies using its services, and has plans for complete US coverage by the end of 2017 (Figure 22 below). Currently, Ingenu is to RPMA as Sigfox is to Sigfox; they are possibly the only company actively building an RPMA network. In addition, much like Sigfox, while a user can communicate over RPMA how they like under Ingenu they do have to use the network that Ingenu provides meaning there is a subscription fee [L, M].

3.2.1.6 NB-IoT

When sigfox began, it disrupted the 3GPP standardization group by claiming a market for devices that had little data to transmit, need to be inexpensive, low power, and long range. Sigfox was successful in this area for a while, but this eventually drove mobile operators with multibillion-dollar businesses to join the platform. Huawei partnered with Ericsson to develop a standard called Narrowband IoT (NB-IoT). NB-IoT sometimes referred to as LTE Cat NB1, is a LPWAN technology that is designed to work virtually anywhere. It connects device and operates independently or on existing cellular base stations that can allocate a resource block to NB-IoT. NB-IoT was also designed to handles small amounts of 2-way data transmitted infrequently, securely, and reliably.

3.2.1.6.1 Deployment Types

For NB-IoT, 3GPP offered three different deployment scenarios; Standalone, Guard Band, and In Band. Standalone mainly uses new bandwidth but can also use in some unused 200-KhZ bands that have previously been used for GSM. Guard band deployment is done by utilizing the bandwidth reserved in the guard band of the existing LTE network. In band however, makes use of the same resource block in an already existing LTE network. Standalone and Guard band deployment options tend to offer the best performance in terms of improved indoor coverage [III]. Figure 21 gives a physical representation of how the three deployment scenarios are implemented.

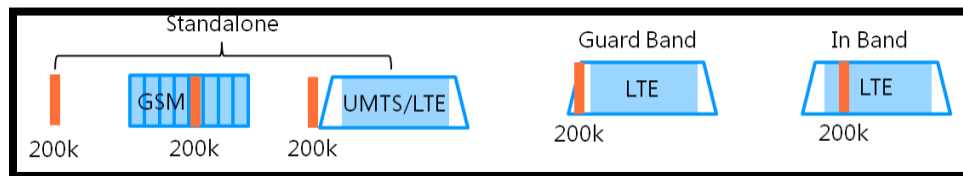


Figure 21 – Deployment scenarios of NB-IoT

*Permission Requested [P13]

3.2.1.6.2 NB-IoT Benefits

The main forces that drove the development of most LPWA technologies were low cost devices and deployments, low power consumption, and increasing the coverage while having large amounts of connections when compared with other wireless networks. LPWA devices are generally battery powered and NB-IoT promises battery life of up to 10 years on a single charge. To do this, it not only uses technologies that allow devices to enter low power modes, but NB-IoT is optimized to consume less power while the device is operating. Another point is that chips that only have support for NB-IoT will cost less because they will be less complexed in the creation

process. The 200 kHz bandwidth of NB-IoT is much simpler than the 1.4 MHz resource block of LTE-M.

3.2.1.7 Communication Technologies Comparison

RPMA makes a lot of sense for our project, the product form Ingenu meets all of the specifications of a LPWAN and boast some of the highest performance statistics as compared to their competitors (See Figure 24). Also, the price point makes RPMA easy to consider being sustainably lower than anything else on the market. In addition, because RPMA uses a worldwide, unlicensed 2.4 GHz spectrum its coverage can be built out to almost anywhere and used for no extra costs over Ingenus' services.

The only drawbacks to RPMA are the lack of competition leaving very few options for the device we choose and the fact that the band that Ingenu operates on is unlicensed, meaning there's nothing stopping others from using it as well so it may well be too busy for us to use, but we can decide upon that in testing.

The drawback, however, is unlikely to ever effect RPMA. The technology it uses to avoid collisions, as well as the technology that ever other communication technology uses to communicate over this license free band, has developed to the point where it can operate on a band with so much noise on the network and still guarantee a communication link with a low number of lost or dropped packets. The license free 2.4GHz band has so much noise because microwaves, and other ISM (Industrial, scientific, and medical) radiating devices populate this band. Therefore, the governing property dictated that this band be license-free.

RPMA has found a way to operate on this band with a low number of lost or dropped packets, which is great for us, the consumer. With operation on this band, the user does not have to pay for communication over the network, like they would over a cellular band. Without this added cost, many more people may utilize the NAT device so this technology would benefit our design and marketing the device to the consumer greatly.

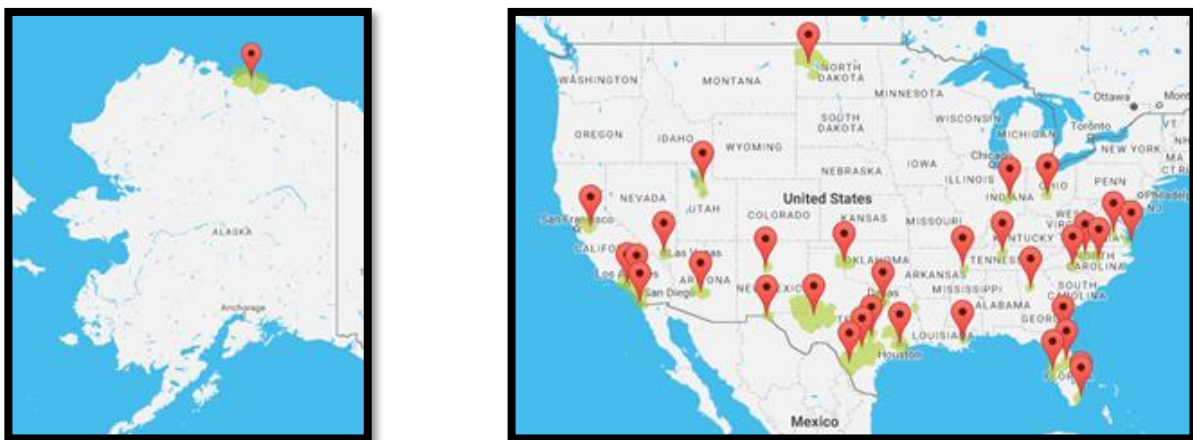


Figure 22 - RPMA US Coverage Map [DDDD]

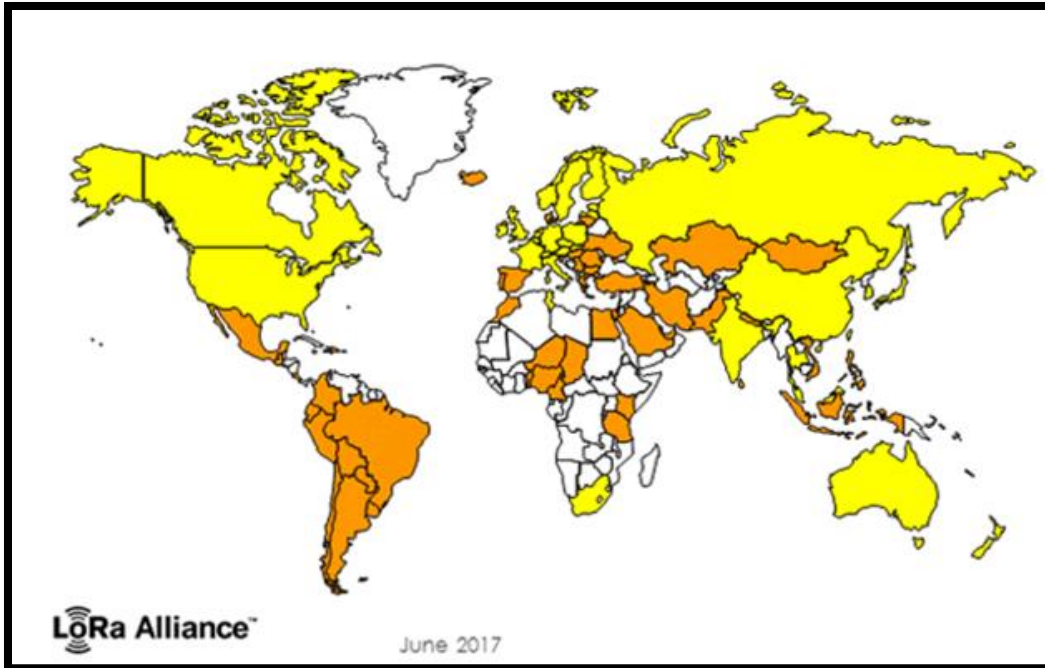


Figure 23 - LoRa US Coverage Map [S]

*Permission Requested [P21]

In many ways LoRa fits the needs and specifications of our project nicely, long range, long battery life and cheap, high density communication. LoRa also has the added benefit of the LoRa alliance backing it up which makes for good support, resources and well developed industry standards unlike RPMA. However, we feel that RPMA will have better coverage than LoRa and a higher quality of service which can be seen from Figure 24 below. Also, the team’s project sponsor has expressed an interest in RPMA over LoRa so for these reasons we will be choosing RPMA.

	Sigfox	LoRa	EC-GSM-IoT	NB-IoT	LTE Cat-M1	RPMA
Bandwidth	100kHz	125kHz	600kHz	180kHz	1.08MHz	1MHz
Coverage	149dB	157dB	164dB	164dB	160dB	177dB
Capacity	50,000/cell	40,000/cell	190,000/cell	200,000/cell	1M/cell	500,000/cell
Battery Life	10 years +	10 years +	10 years +	10 years +	10 years +	10 years +
Throughput	100bps	290bps - 50kbps	473kbps	250kbps	1Mbps	624kbps
2-Way Data	No	Class dependent	Yes	Yes	Yes	Yes
Security	16bit	32bit	3GPP (128-256bit)	3GPP (128-256bit)	3GPP (128-256bit)	AES 128bit
Scalability	Low	Medium	High	High	High	High
Mobility Support	No	Yes	Idle Mode	Idle Mode	Connected+Idle Mode	Yes
Location Support	No	Yes	Needs GPS	Needs GPS	Needs GPS	Needs GPS

Figure 24 - LPWAN Comparison Table [J]

*Permission Requested [P24]

Update: RPMA was removed from the final design due to problems within Ingenu.

3.2.1.7.1 Low Power Modes

As mentioned before NB-IoT consumes very low power as do most LPWA technologies. Almost all of these technologies consume low power when the devices are not operating, meaning they all consume similar amounts of power when in sleep mode. What separates NB-IoT from other low power claims is that when the device is operating and has to process data, processing simpler waveforms like NB-IoT will consume less power. LTE-M has low power modes that help it achieve power consumptions close to that of NB-IoT. The use of those low power modes however, fall on the developer to implement them efficiently.

Achieving power efficiency of battery life of up to 10 years on a single charge for LTE-M1 requires designers to utilize the two available modes for reducing power. These modes are Power Saving Mode (PSM) and Extended Discontinuous Reception (eDRX). PSM is a mode where the LTE device will tell the network it is going into a “deep sleep”. Based on some logic or timer, the LTE host device will wake up when it needs to transmit to the network. It will remain in RX mode for 4 idle frames so that the network can reach it if need be. As shown in Figure 25, devices in this mode will be dormant for majority of the time which will reduce power consumption. In a blog published by LinkLabs [YY], it is said that power consumption can be even lower than other LPWA technologies since the TX rate for LTE-M1 is much higher due to the lack of interference in the licensed spectrum.

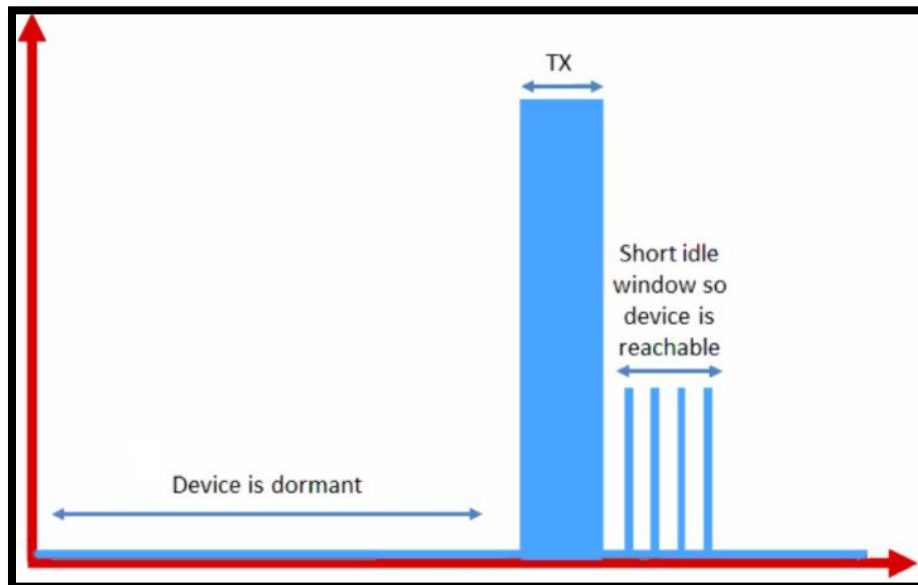


Figure 25 - Power Profile for LTE-M1 Power Saving Mode.

*Permission Requested [P10]

Extended DRX is bit different compared to PSM. LTE paging cycles are a period of time during which the network can contact a device if any data is queued for transmission. These paging cycles are normally 1.28s with 10.24s in between cycles. Taking advantage of the eDRX allows the LTE device to define how many “hyper frames” of 10.24s it will sleep before checking back in with the network. The mobile network operator can set the maximum number of hyper frames a device can sleep for, but it should be at least 40 minutes. According to experts at LinkLabs [YY], For a LTE-

M1 device that transmits data once per day, and wakes up every 60 hyper frames to check for commands (this would be about every 10 minutes), a life of 4.7 years is achievable on 2 AA batteries. In contrast with eDRX An LTE-M1 devices that transmits once per day in full PSM mode could last well over 10 years on 2 AA batteries. Figure 26 illustrates the basic idea of eDRX.

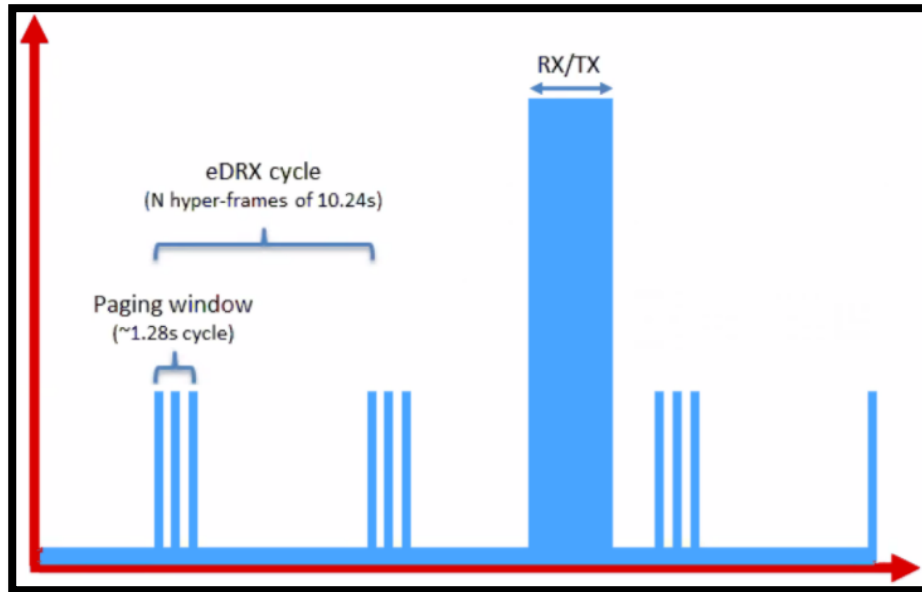


Figure 26 - Power profile of an LTE-M1 UE in eDRX Mode

*Permission Requested [P10]

The designers at Ingenu claim that one of the ways that RPMA meets the low power requirement to be considered LPWA technology, is by having their device run at extremely low data rates and transmit as little data as possible. Combing through the RPMA nanoNode datasheet we can also see the different low power modes that are available for this module. The device can either be in a “Deep sleep” or in “Idle” state. While in the deep sleep state, the nanoNode cuts power to all the power regulators except a few low quiescent LDO regulators. These regulators kept on keep the minimum amount of circuitry alive for keeping track of timers, and keeping a 32 kHz clock and some minor interface circuitry alive. The idle state refers to a state where the device is awake, the clock is on, however the receiver is in an off state.

Sigfox and LoRa take similar approaches to achieve power saving effects of the same magnitude. Sigfox has designed a lightweight protocol to handle small messages. This along with their small payload for uplink and downlink helps to conserve power consumption. Technologies using LoRa operate in sub-GHz ranges to conserve power and extend range. Low frequency operation is preferred in cases like this because although high-frequency options provide high data rates, they limit the range at acceptable power levels.

3.2.1.7.2 Small bandwidth

Another feature of LTE-M1 is its reduced bandwidth compared to normal LTE devices which operate with a bandwidth of 20 MHz. LTE-M1 reduced this and enabled devices to operate with a maximum channel bandwidth limited to 1.4 MHz. it operates a lower data rate connection within

one resource block in one time interval, which allows for a much simpler frontend that only has to digitize 1.4 MHz of spectrum versus a whole 20 MHz spectrum. It also allows the device to have only one antenna. Operating the device using half-duplex mode will mean that only half the processing power will be used because it is never transmitting and receiving at the same time. Most modules support both full and half-duplex and it is up to the designer to choose which mode will work best for the situation at hand.

As the name suggest, narrowband-IoT operates on a small bandwidth. Referencing Figure 27, we see that NB-IoT beats its other LTE counterparts with a user equipment receiver bandwidth of 200 kHz.

	LTE Cat-1	LTE Cat-M1 (Release-13)	NB-IoT (Release-13)
Peak data rate	DL: 10 Mbps UL: 5 Mbps	DL: 1 Mbps UL: 1 Mbps	DL: ~20 kbps UL: ~60 kbps
Bandwidth	20 MHz	1.4MHz	200 kHz
RX antenna	MIMO	Single Rx	Single Rx
Duplex mode	Full duplex FDD/TDD	Full/Half duplex FDD/TDD	Half duplex FDD only
Transmit power	23 dBm	20 dBm	20 dBm

Figure 27 – Comparison of three LTE technologies

3.2.1.7.3 Data Rate & Transmit Power

Another look at Figure 27, shows that the max throughput of LTE-M1 is 1 Mbps on both the uplink and downlink, as compared to about 20 Kbps and 60 Kbps with NB-IoT. The max throughput of 1Mbps of LTE-M1 is achieved when using full-duplex. In half-duplex mode, the device can get up to 300 Kbps for downlink and 375 Kbps on uplink. As mentioned before, LTE-M1 supports half-duplex which will cut processing power in half, while NB-IoT can only use half-duplex. We also notice that the transmitting power of LTE-M1 is about 20 dBm. This Tx power can get up to a maximum of 23 dBm for LTE-M1 and the RPMA module, which can be a problem for the device because it will have to be able to source around 500mA or more or peak current when required by the module. Having a supercapacitor bank in parallel to the power supply may be helpful, but this factor really comes down to choosing the right battery chemistry and capacity for the design.

3.2.1.7.4 Other Considerations

Being that NB-IoT is not completely a part of LTE, it either needs to be deployed in some unused GSM spectrum or it needs to operate in a sideband using different software than LTE. Operating in a sideband using different software can be expensive for carriers. Also it is unlikely that carriers are willing to reduce the number of resource blocks allocated to LTE handsets for there is a lot of money that is being made in that market. The complexity that comes along with the implementation of NB-IoT is a big question mark.

Often times there is not a vast amount or country wide 200 kHz spectrum from GSM available for NB-IoT to operate on. This could introduce unintended complexity in modem frontends and antennas. Also, there is the potential for licensing costs where big companies like Ericsson and Huawei may require licensing fees.

There is a good chance that by using LTE-M, you will have to pay IP licensing to companies like InterDigital and Qualcomm for access to innovating features like orthogonal frequency-division multiplexing (OFDM). The power efficiency of LTE-M1 is questionable because eDRX and PSM are being deployed for the first time, which makes their power efficiency sort of hypothetical. Without knowing if the networks will allow these low power modes to operate as they were designed or if specific features native to a carrier will minimize the power efficiency, we won't know exactly of efficient these modes operate.

In the United State, there are many large organizations that currently have billions of dollars invested into LTE networks. Outside the U.S. however, there are many areas around the globe that have far less LTE. Inside these regions that lack LTE there are usually larger deployments of GSM on which to seek unused band for NB-IoT use. NB-IoT would be ideal for innovators seeking affordable entry points in new global markets. Also, LoRa and Sigfox were developed out of the European Union. This means the popularity of modules for use in the United States may be an issue because the technology hasn't fully matured to all areas of the world yet. These considerations must be taken into account before making any decisions about any of these technologies.

3.2.2 Bluetooth Technology

Bluetooth is a low-power wireless technology that is used to stream audio, transfer data, and broadcast information between devices. This is an applicable technology for our application as it is low-power and allows the transmission of data. There are two types of Bluetooth technology, the basic/ Enhanced data rate (BR/EDR), and the Low Energy (LE). The BR/EDR enables continuous wireless connections and uses a point to point (P2P) network topology. This is ideal for headsets and wireless speakers, but not our application. The other technology is LE. This enables short burst wireless connections and uses multiple network topologies include P2P, broadcast, and mesh. Mesh is applicable for asset tracking, so it is applicable for our application. [FF].

While this technology is applicable to this project, the team decided not to investigate this option further as it would require every user to have a Bluetooth enabled device and for the device to be in range of a receiver at all times. Therefore, this technology was not contender.

3.2.3 RFID Technology

Description

Radio-frequency identification, RFID, is a technology that uses electromagnetic fields to detect and track tags. These tags could be attached to people, objects or information on a microchip, in the form of a serial number, which is attached to an antenna. The chip and the antenna attached

together is what the RFID tag is comprised of and that, combined with the reader with an antenna is what makes up the RFID system [RRR].

The antenna allows the chip to communicate to the reader, the identification data at hand. This works in such a way that radio waves reflected from the RFID tag, are translated by the reader, which then converts it into digital data. As shown in Figure 28 below, this information can then be passed on to the computer. The reader sends out the electromagnetic waves, and the tag antenna receives them. Radio waves travel through most non-metallic resources. This allows them to be enclosed in plastic that protects them from any sort of weather or another catastrophe. The tags contain electronically stored information. (More detailed information about tags can be found below). Passive tags collect energy from a nearby RFID reader's interrogating radio waves. They use this energy to then power the microchip's circuits. The chip then controls the waves that are sent back to the reader from the tag and the reader converts the new waves into digital data. Active tags have a local power source, for example a battery. These tags are able to function at hundreds of meters from the RFID reader. In fact, the tag does not need to be visible to the reader. It can actually be embedded in whatever is being tracked [RRR].

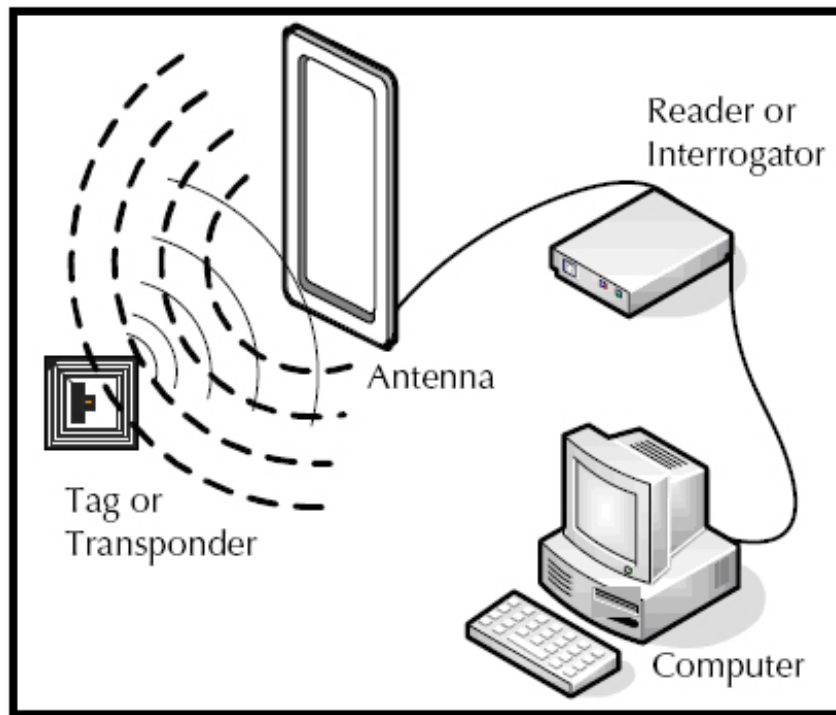


Figure 28 – RFID Communication [TTT]

*Permission Requested [P18]

RFID might be said to be a better technology than the use of barcodes but that is not always true. It is not a fair comparison because they may be similar but they are two very different technologies with different applications, which tends to overlap at times. The biggest difference between RFID and barcodes is the fact that some type of scanner has to see the barcode for it to be used, whereas RFID tags can be read from a distance, but still has to remain within range of the reader. Now barcodes do have some limitations when compared to the RFID technology. Once a barcode label

is ruined, it is unusable. This could happen when the barcode sticker is rubbed away, wet and peeled off or somehow becomes unreadable. The good thing about barcodes is that they are inexpensive to use on a product and they can be extremely effective for certain tasks. This helps reassure the barcode industry that they will not be running out of work, at least for the nearby future [RRR].

History of RFID

RFID has technically been around since before 1970s and has recently established itself as an inexpensive and effective technology. Before, the drawback was the fact that RFID was too expensive and not as widely usable. Although RFID will not completely replace the need and desire for barcodes, they do solve some of the problems that barcodes face, and could even compete with barcodes financially, depending on the price of the tags that get made [RRR].

Read-Only and Write-Only Tags

The microchips are either read-write or read-only in the RFID tags. Some read-only microchips have information stored on them during the manufacturing development. The information on such chips cannot ever be changed. Additional tags can have a serial number written to it once and then that information can also not be overwritten later.

Through read-write chips, the data can be added to the tag or written over current data, but only when the tag is in range of the reader, however, the read-write tags typically have a serial number that cannot be written over. It is possible for additional parts, or even all the data in read-write tags, to be locked and therefore not changed as well [RRR].

Reader and Tag Collisions

There are two types of collisions that can occur, reader collisions and tag collisions. With reader collision, the signal from one reader can affect the signal from another where coverage intersects. There is a way to avoid this collision, and that is to instruct the readers to read at different times, which is referred to as time division multiple access, TDMA. This does pose the issue of an RFID tag being read twice. To avoid this problem, the system is then designed to not read a tag by another reader once it has already been read by one reader.

The other type of collision, tag collision, occurs when the reader has to read many chips that are in the same field. The reader can become confused because a chip might reflect its signal at the same time another tag is reflecting its signal, causing the collision. There are ways to avoid the collision by designing the systems so they have the tags respond to the reader one at a time, which is acceptable since each tag can be read in milliseconds [RRR].

Active and Passive Tags

Passive RFID tags do not have a battery, whereas active RFID tags are battery-powered, allowing them to continuously transmit their own signal. The battery is also used to run the microchip's circuitry. Active tags can provide a longer read range than passive tags, but this means that passive tags are cheaper than active tags. Since passive tags are not battery operated, they tend to draw electromagnetic energy from the reader. Then there are semi-passive tags which are similar to

active tags in that they also use a battery to run the microchip's circuitry, but they also are similar to passive tags in that they draw energy from the reader and that is how they communicate as well. Semi-passive and active tags tend to be useful when tracking items over long ranges, but are typically used on more valuable items because they tend to cost more themselves [RRR]. The table in Figure 29 outlines the differences between active and passive tags.

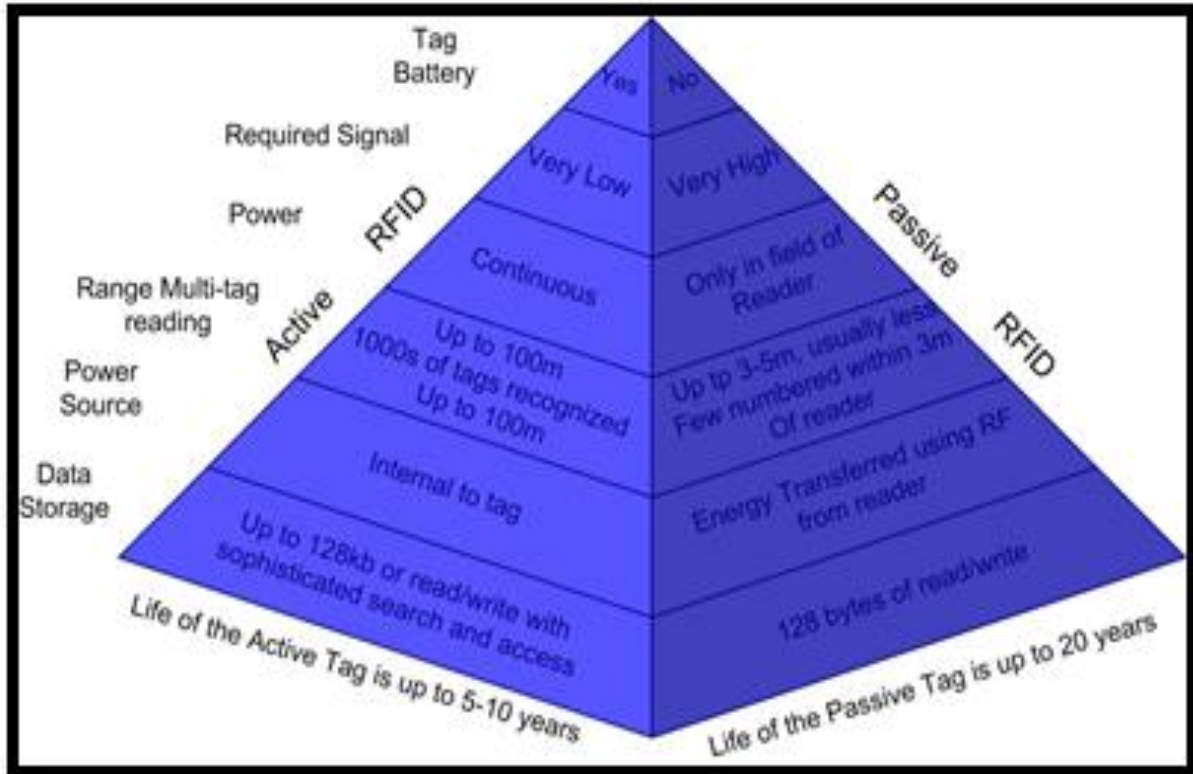


Figure 29- Active vs. Passive Tags [UUU]
*Permission Requested [P19]

Low, High and Ultra-High Frequencies

RFID tags and readers must be tuned to the same frequency in order to communicate. Although low, which is 125KHz, high, which is around 13.56 MHz, and ultra-high, which is around 850MHz, frequencies are more commonly used in RFID systems, as seen below in Figure 30, they can use multiple different ones. It is important to know that radio waves behave differently at different frequencies, which is why it is crucial for the right frequency to be chosen depending on the application [RRR].

RFID Frequency Bands			
Band	Regulation	Range	Data Speed
120-150 kHz	Unregulated	10 cm	Low
13.56 MHz (HF)	ISM Band Worldwide	10 cm – 1 m	Low – Moderate
433 MHz	Short Range Devices	1 – 100 m	Moderate
865 - 868 MHz (Europe) 902 – 928 MHz (North America) (UHF)	ISM Band	1 - 12 m	Moderate – High
2450 – 5800 MHz (microwave)	ISM Band	1 – 2 m	High
3.1 – 10 GHz (microwave)	Ultrawide Band	Up to 200 m	High

Figure 30 - RFID Frequency Band Comparison [QQQ]

Conclusion

Unfortunately, RFID raises a major security risk. This is because RFID tags can be attached to clothing, possessions or even implanted in animals and people. This can be wrongly used to gain personal and private information about a person. This is also a concern with the NAT device this team is designing, as it could be used wrongly to track a person or item, but as it is further discussed in this paper as a safety and ethical constraint, it is not the purpose of this device.

The senior design team and the sponsor for the NAT device decided against implementing RFID technology into this project. RFID, although has progressed to a more cost-efficient technology and solves many of the problems associated with bar codes, it also has a major issue of reader collision. This is when the signal from one reader interferes with the signal from another where coverage overlaps. This could pose a problem, for instance, in a hospital where there could be numerous interferences surrounding the device. So instead, the team decided on using a radio module which will be implemented into this NAT device and attached to anything and sent anywhere. At this time, the sponsor does not want to consider RFID, but it could be a possible implementation in a future, similar project.

3.3 Strategic Components and Part Selections

The sub-sections included below detail the components that we have researched, compared, and thoroughly analyzed towards our selection of components of our device. These include overviews of the possible technologies or information about modules that may be necessary to understand to

conduct proper research, then specific research into the modules and what separates them from modules similar, and then finally a comparison of the data for each module and a final recommendation of which technology should be used for the final product. However, since our product is sponsored, it is ultimately the choice of the sponsor which technologies, modules, and components our device will implement and utilize.

3.3.1 Embedded Computing Hardware

Coordination and control of the data passed between the GPS receiver, inertial measurement unit, and IoT communication modules requires central digital logic. Embedded systems may make use of specially-designed Application Specific Integrated Circuits, more flexible Field Programmable Gate Arrays, or more generalized microprocessor designs. Each shall be discussed in detail.

Application Specific Integrated Circuits (ASICs)

ASICs, capable of performing only their explicitly defined function, are the smallest physical size, fewest part count and lowest power consuming option for embedded logic, but also the most expensive. [JJ] ASICs are designed using tools such as a Hardware Description Language (HDL) and compiler to generate an electrical schematic that performs the desired computing functions, then realize the schematic by designing a layout database of individual transistors and logic gate in a workable layout which are then simulated to account for parasitic effects, redesigned, and finally sent to a foundry for fabrication. The cost of performing this design, analysis, and simulation is referred to as “non-recurring engineering costs” and require expertise and facilities costing the equivalent of tens of thousands of dollars if performed by professionals, so this option is far beyond the capabilities of the five-member student engineering team.

Field Programmable Hardware (FPGAs vs Microcontrollers)

FPGAs achieve a level of performance comparable to ASICs by generalizing using a minimal level of abstraction; instead of individual transistor-level design, the HDL is translated into a customizable routing fabric between configurable logic blocks. The low-level electrical design is handled by the FPGA vendor, who provides the FPGA chip at much lower costs than customized ASICs by the economy of scale provided by being able to sell the same FPGA design to many customers who can then design for their specific application using an HDL and automated tools. The field-programmability also allows for several software prototypes to be developed and tested for the same hardware revision, which is an essential property of our development cycle. While this makes using FPGAs a much more viable option than designing our own ASICs, a steep learning curve remains as our team has only minimal familiarity with HDLs, designing the most basic of digital systems. To meet our time constraints, we need a platform that can be developed for using a high level software programming language. This implies the use of a general purpose central processing unit. The use of external microchips for the onboard peripherals limits the available board area for a CPU and its associated chipset. The size constraint therefore dictates the use of a single system on a chip, including sufficient RAM and electrically erasable flash memory for successive prototype software iterations. Future needs in revisions beyond the timeframe of this project also requires sufficient processing power to support, such as the proposed inertial navigation algorithm to be developed by other teams.. Additional desired features include integrated timers, flexibility in available on-board communication protocols, low power

consumption, and a low-cost software development toolchain. These features are commonly provided by devices classed as microcontrollers [KK]. Solutions from several vendors are available and the following were evaluated.

3.3.2 The Microcontroller Modules

Texas Instruments MSP430FR596

The TI MSP430 CPU uses a 16-bit RISC architecture, and is used in the MSP430FR596x line as part of a “Mixed-Signal Microcontroller”. Its main selling point is its Low-Energy Accelerator for digital signal processing, allowing for “ultra-low-power modes” such as an active current draw of about 1.9 mA at the maximum 16 MHz clock speed, with standby draw of 500 nA and 45 nA at shutdown. Chips are available with a maximum 256KB of FRAM, providing the performance of standard RAM with the non-volatility of Flash, allowing for a unified memory system. Six 16-bit timers are provided and can be driven by a selection of internal oscillators, external timing crystals, and a real-time clock. Up to four serial ports supporting UART and IrDA and another four supporting I²C are available. Finally, TI provides the IAR-KICKSTART C/C++ development toolchain and the Code Composer Studio IDE free of charge for their users. [LL]

Atmel ATUC256L3U

The Atmel AVR CPU uses a 32-bit RISC architecture at a maximum clock rate of 50 MHz for significantly greater performance than the TI MSP430 previously described. The ATUC256L3U is capable of much greater processing speeds, with single-cycle flash memory access at up to 25 MHz, with single cycle SRAM capability at all clock rates. The ATUC256L3U also provides a full speed USB 2.0 interface, which the MSP430 lacks, which simplifies the configuration workstation interface. Atmel advertises “picoPower® Technology for Ultra-low Power Consumption”, but even at equivalent clock rates of 16 MHz the ATUC256L3U draws nearly 50% more power, and roughly three times as much as that at full speed, although standby power is halved. 32 KB of SRAM and 256 KB of flash and seven oscillator sources are provided on-chip, with 6 timer channels. Four UART ports and two I²C ports are available. Atmel does not provide a toolchain nor IDE, but development is supported by the free and open-source GNU Compiler Collection (GCC). [MM]

PIC24FJ128GC006

Upon the recommendation of the project sponsor, a Microchip PIC24 microcontroller was favored. Compared to the MSP430 and the Atmel AVR, the PIC24 is outclassed in every category considered. The PIC24FJ128GC006 provides 8KB of SRAM, 128KB on on-chip flash memory, seven timers, four UART ports and two I²C ports. A C compiler and IDE based on Netbeans is provided by Microchip free of charge, with a more heavily optimized C compiler available for paid license. [NN]

3.3.2.1 Microcontroller Comparison

Of the three microcontroller families considered, PIC24 had the lowest current draw, followed by TI, with ATMEL at the highest. Nevertheless, all three chips were found to have sufficiently low

active and standby current draw to be usable in our design. Some of the other chips that were researched had benefits in areas that the Microchip microcontroller did not, that would benefit our project. However, it was ultimately determined that these features were not desirable enough to outweigh the familiarity our sponsor had with the PIC24 architecture, flattening the learning curve significantly. Additionally, the sponsor already owned a PIC24-compatible In-Circuit Emulator device, providing significant project cost savings.

Therefore, the Microchip PIC24FJ128GC006 was decided that as long as it had the specifications that fit the requirements of our project, the request of the sponsor to continue utilizing this microcontroller in our new design, as well as the many benefits it provides in familiarity in use, resources available to program the new firmware for the microcontroller PIC chip, as well as the number of additional tools available including the In-Circuit Debugger as well as the development board that we have access too were acceptable. In further designs of this product, including the implementation of INS algorithms, another PIC may have to be tested to withstand the processing power needed by such a code heavy and power-hungry application. The comparison of specific values are shown below in Figure 31.

	TI MSP430FR5962	Atmel ATUC256L3U	Microchip PIC24FJ128GC006
Architecture	16-bit RISC	32-bit RISC	16-Bit C-Optimized
Active Current Draw @ 16MHz	1.9 mA	2.8 mA	100µA
Standby Current Draw	500 nA	220 nA	75 nA
Maximum Clock Rate	16MHz	50MHz	32MHz
RAM Size	256 KB FRAM	32 KB SRAM	8KB SRAM
Flash Size	Unified with RAM	256 KB	128 KB
Timers	6	6	7
UART Ports	4	4	4
I ² C Ports	4	2	2
USB Ports	None	1	None
Development Toolchain	Free (IAR-KICKSTART, Code Composer Studio)	Free and open-source (GCC)	Free, with basic optimizations (XC-16, MPLAB X)

Figure 31 - Microcontroller Comparison Table

3.3.3 GPS Module

The base of our project is that it is an asset tracker. Therefore, a main module in our project is the GPS (Global Positioning System). Regarding our project’s description and requirements, we require a GPS that is accurate (within 3 meters), that is small (smaller than the size of a credit card and less than about 1 inch thick, and consumes a low amount of power as to extend battery life, among other things. Thus, the GPS module section of this paper is to discuss and analyze the few

modules that have been researched individually, then view a few of the best ones together to make a final decision on which one is the best product for the design and specifications.

3.3.3.1 Introductory GPS Research

This section will go through a few of the major topics and technologies related to GPS. The technologies that will be discussed may or may not be implemented in the modules that we will look at more in depth further in this paper. What should be gained from this section is a better understanding of overall how GPS functions, as well as how some GPS modules have implemented certain technologies to fix some of the shortcomings that GPS modules may have.

GPS is not the only navigation system available, nor is the main technology. GPS functions by utilizing GNSS (Global Navigation Satellite System), which is a constellation of satellites providing signals from space that transmit positioning and timing data to GNSS receivers. Therefore, GNSS is an overall system of satellites and GPS is only a component of this, developed by the United States Department of Defense and was for a time the only fully functional GNSS system. There are other components of the GNSS system as well, those being GLONASS and GALILEO. GLONASS (**G**lobal **N**avigation **S**atellite **S**ystem) is operated by the Russian Defense Ministry with 24 satellites (21 standard and 3 reserve planned) on 3 orbital levels, and GALILEO is the European GNSS system developed by the European Union and has 30 satellites on 3 circular orbits in its constellation. The benefits of GLONASS is that it has an angle of 64.8deg from the equator, which is the highest angle of all GNSS systems which gives better reception in polar regions. Utilizing this benefit would enhance our project because it would allow our system to use GNSS longer before having to revert to code heavy, battery draining INS location. The Chinese also have their own GNSS systems, Beidou 1 (Current) and Beidou 2/Compass (Future). The current system has 4 satellites in service, the future planned service will consist of 35 satellites. Some of the proposed modules utilize more than just the GPS component, which is why it is important to have a basic understanding of these other GNSS. [V]

The base of GPS and other GNSS services is to calculate a position and determine the current time. GNSS and therefore GPS systems combine sophisticated satellite and radio technology to provide navigation to receivers, such as the modules that we will investigate to be implemented on our device. In order of this position to be determined, a GNSS receiver must receive time signals from at minimum 4 receivers. How this occurs, such as the algorithm or physics it uses, is not of importance to our implementation of the module, as we are not developing a new receiver. We assume the module works properly, and during testing we make sure this is true. Something that may be important to us, however, is that to determine the position the receiver will either use the last measurement value or estimate a new position and calculate errors until they are zero (by repeated iteration), we would greatly prefer the former option as it outputs a quicker result. [V]

In a few datasheets, white papers, and other informational documents there are a few terms in relation to how the receiver starts calculating its current position that are worth explaining here. There is a value that is an important specification in GPS/GNSS modules called Time To First Fix (TTFF), which “a measure of the elapsed time required for a receiver to acquire the satellite signals and navigation data, and calculate the first position solution” [W]. Basically, it is how long the module takes to figure out where it is. Other terms in relation to TTFF is cold start, warm start, aided start, and Hot start. A cold start occurs when there is problem with the receiver’s estimate of

the current time and position, or its copy of the satellite almanac data (described later). This could be due to an internal problem in the module (such as power loss), causing it to lose its data, a reprogramming of the module, or just a fresh start on the module. This type of start requires the longest wait time, since the module has no idea where it is and must start from nothing. The cold start can be around 12.5 minutes in addition to the normal start time, most, if not all, of that time coming from the acquisition of the new satellite almanac data. The warm, also known as normal, start assumes that the module already has some of the data stored, generally at least a recent copy of the satellite almanac data. This is typically around a 2-5min start time. A hot start occurs when a receiver has a standby feature, and thus is still maintaining the data necessary to find the fix on the location. Generally hot starts TTFF is around 10 seconds, much better than the cold or warm starts. On some devices, there is an aided start feature, which allows the person utilizing the module to give it an estimate of the devices location, so that if the device already has current almanac data it can achieve hot start TTFF numbers, rather than warm start TTFF numbers. [V]

The satellite almanac data, that was discussed just above regarding start times, is a signal/message that satellites will broadcast to the receivers using them. The information that it gives is the approximate orbit for each satellite in the constellation, valid for long periods (thus unnecessary to cold start or update it very often). It is used to predict the satellite visibility and estimate the pseudo-range to a satellite. The message can come from any satellite. [V]

A-GPS (Assisted GPS) is a technology that cellular devices uses, which aids in why cellular devices get a GPS fix at a seemingly much faster rate than other types of devices. A-GPS integrates GPS and communications. It utilizes the separate wireless communication channels to substantially improve the processing power of the GPS receiver so that they can operate successfully in disadvantaged locations and circumstances. The fundamental idea of Assisted GPS is to provide the GPS receiver with all the information it can, by some alternative means of communication, before it acquires the satellite signal. Research may be done into using this technology, but initially the project specifications does not include this technology. [X]

WAAS (Wide Area Augmentation System) is a satellite based augmentation system. It is intended to augment and enhance the accuracy of consumer grade GPS units (currently about 10-15m), bringing the accuracy to something of 3-5m. Since our project specifies that the GPS accuracy should be within 3m, this is a technology that should be researched further. The WAAS network is composed of 38 WAAS reference stations (WRSs) located across North America, 3 master control stations, 2 geostationary Earth orbit (GEO) satellites, and 4 ground uplink stations (GUS). Each WRS is equipped with a high- quality clock and multiple GPS receivers. This collected data is sent to master control stations, which process the data. This data will determine satellite integrity, differential correction, residual errors, and ionospheric delay. This information is what WAAS uses to correct the GPS location data. The data is then uploaded to the two GEO satellites that transmit the correction at the GPS L1 frequency (L1 frequency allows the message to be read by all WAAS enabled receivers at no cost, either in hardware, software, or usage fees. This is another benefit for our project because it doesn't cost us anything to utilize this technology.). The correction message includes 2 components, the location independent parameters of ephemeris (location of naturally occurring astronomical objects as well as artificial satellites) and clock error, and area specific ionospheric errors transmitted in a latitude-longitude grid [Y]. Additional research even shows that Garmin, in their description of WAAS, promotes an accuracy of better

than 3 meters, 95% of the time. An average of 5 times better than GPS signal alone. [Z] This would be phenomenal for our project, allowing us to meet our specifications with ease.

GPS modules can implement or support an active or a passive antenna. The difference between these two things is that an active antenna is any antenna with an integrated signal amplifier in the unit, whereas a passive antenna does not include the onboard amplifier. The actual antenna element between the two are the exact same. The only practical reason to use an active antenna over a passive is to compensate for cable loss in receive applications. Active antennas do not increase directional gain, which means it has no effect on the antenna's fundamental ability to receive RF energy. [BB]

3.3.3.2 The GPS Modules

The modules that we are going to investigate include Ada Fruit FGPMMPA6H, Nano Spider 4033, Nano Spider 4400, Nano Hornet, Trimble Aardvark, and the U-Blox UBX-M8230-CT.

FGPMMPA6H from GlobalTop Technology Inc.

FGPMMPA6H (shorthand written as PA6H) GPS Module is a standalone module from GlobalTop. It boasts the industry's highest level of sensitivity (-165dBm) and instant TTFF, with the lowest power consumption (when it's datasheet was written in 2011).

The PA6H module has a built-in antenna, but also allows the use of an external one through an embedded function for external antenna I/O with an automatic switching function, short circuit protection, and "Antenna Advisor" (detection and notification of antenna statuses). This allows for the connection of a better antenna, if it complies with certain external antenna requirements.

Antenna Advisor, as mentioned above, is an antenna system exclusive to the PA6H. It can detect and notify if the active antenna has been shorted, and which antenna is in use. This is generally useless to us, as we would likely just use the internal antenna to save space on our device, as space is limited and small size is of high priority.

It has ultra-high sensitivity, calculated at about -165dBm, without the use of a patch antenna, in open sky reception. This seems about average in the current year, however.

There is a 12 multi-tone active interference canceller, which is described as the ability to reject external RF interference which can come from other active components on the main board (through the sharing of Wi-Fi, GSM/GPRS, 3G/4G, and Bluetooth in the navigation system). This is done without any needed hardware changes.

AGPS support for fast TTFF, which allows users to download the Extended Prediction Orbit (EPO™) data to the GPS engine by internet or wireless network. The PA6H's engine will use the EPO™ data to assist position calculation when navigation information of satellites is not enough or if in a weak signal zone. This would assist in our requirement that the GPS signal be available as much as possible. We need a strong GPS module, so the use of INS is not always necessary (as it requires much power and therefore drains the battery supply). The more we can utilize GPS as opposed to INS the better for the overall performance of our product.

EASY™ is an embedded assist system for quick positioning. The PA6H's embedded engine will calculate and predict automatically the single emperies when powered on, and save the prediction into the memory, for a maximum of 3 days. The PA6H will use this information for positioning if it cannot get information from satellites. This enhances the PA6H's performance in indoor or urban conditions.

For power saving, which is necessary in our device, the PA6H has an AlwaysLocate™ Intelligent Algorithm (Advance Power Periodic Mode) for power saving. The Algorithm can be set to decide the operation level of the GPS function, thus reducing power consumption. This will cause the PA6H to suffer positioning accuracy, but will extend usage time of the product. This would be fine in our product, as our design specifications want the GPS module to be able to go into a sleep or low power mode when not moving or not in use, and allow software to awaken it when it detects movement, or a certain time frame that will report the devices location (as determined by user).

The PA6H also promotes an embedded logger function, which does not need a host MCU (Microcontroller Unit) or external flash to handle the operation. The PA6H will internally log UTC, latitude, longitude, valid, and checksum and can internally handle up to 2 days under the AlwaysLocate™ Condition. Our device does not need this operation, per say, because we will have Flash memory internal to our device yet external to the GPS module. It would be a problem, the PA6H only being able to handle 2 days' worth of data, but our separate flash memory should be able to handle much more than the internal flash could. The benefit of this enhancement would be that it should be easier to log data external to the module, if can already do it itself.

The PA6H communicates over UART, which our device should support.

While it has its own technology that may make up for it, but the PA6H doesn't utilize or allow to be utilized any of the technologies that we found in our research. These technologies would make the GPS much better suited for our application, would it have been implemented.

The device is also a few years old so it may be hard in the future to continue using this technology should we decide on it. It might be discontinued, and thus require expensive redesign costs or expensive costs to continue buying the same unit, as it would likely increase in price.

All information for the PA6H GPS module came from Research Source [AA]

The Nano Spider 4033 from OriginGPS

The Spider Family of GNSS Receiver Modules were designed to address markets where size, weight, stand-alone operation, power consumption, and design flexibility are all important. At the time of its development, it was the industry's smallest fully integrated, highly sensitive GPS and GNSS module. This is important for our design as our design is impacted by those qualities listed.

The Spider Family offers OriginGPS' proprietary NFZ™ Technology, which allows for high sensitivity and noise immunity even under marginal signal conditions.

All the Spider family needs is an antenna and a power supply on a 2-layer PCB. The problem with that is that we have similar options that have embedded antenna, so choosing one without one needs to have many more benefits.

Other than the benefits listed above as part of the Spider Family, there are many other more specific reasons why this module stands out from others.

The 4033 module supports active and passive antennas. It has integrated dual-stage LNA (Low Noise Amplifier), SAW (Surface Acoustic Wave) filter, TCXO (Temperature Compensated Crystal Oscillator), RTC crystal, GNSS SoC (System on Chip), LDO (Low Drop Out) regulator, RF shield, and PMU (Power Management Unit). The TCXO is important in our design for higher sensitivity, shorter TTFF, and better navigation stability. The RTC crystal is necessary for hot and warm start capabilities of the module.

The Constellation configuration that the 4033 module allows are both GPS and GLONASS as default, but also GPS and BEIDOU available. BEIDOU doesn't seem to allow much benefit in terms of accuracy, so in our applications design this addition would, at the current state of BEIDOU, be unnecessary.

The 4033 module implements static navigation. This means that when the module has sensed that it is stationary, it will 'freeze' the position fix. This is unfrozen when the computed position exceeds a set distance, as well as if the speed increases above the threshold again. This feature is disabled by default, but in our application this feature may be useful. The design right now will attempt to do this anyway, as when the product is not moving it will move into some variation of a low power mode. If the module already has hardware built in to do this, it may allow a faster restart time or an easier time attempting a low power mode instead of turning it completely off and back on again.

The 4033 module also has a variation of AGPS. This particular solution allows for hot starts even in weak signal as well as moving start-ups. It also has Locally generated AGPS, in the form of EASY™, which was described in a module above. In short, it allows for quick positioning. This would benefit our module in that the user would not have so much of a wait time, so the reported fix would come in faster. This would definitely be a benefit to the user experience.

The 4033 module also has a decent number of power management modes. These include full power continuous, and power save modes standby, periodic, and AlwaysLocate™. The best performance obviously comes from the full power continuous mode, however this mode will drain the power unnecessarily if used when the device doesn't need all of the features. The standby mode reduces the current drain by stopping navigation and entering the internal processor into standby. Once any byte is sent to the serial port the device will return to full power mode. The periodic mode will enter on/off modes autonomously with a few commands. This means that for a determined amount of time the device will go into full power and then for a determined amount of time it will enter standby. This is set by the user, and for certain applications of our product this would be very useful. AlwaysLocate™ is an intelligent controller of the periodic mode. Power distribution is internally controlled, and determined by factors such as environment and motion conditions. If this proves to be better suited for our application than just user defined periodic mode, then this will be useful for our product as well. There is also a backup mode, which means a low quiescent power state where receiver operation is stopped. When the module leaves backup mode, the receiver uses all internal aiding to give the fastest possible TTFF. [DD]

Other specifications, such as those values defined in the datasheet, will be shown and discussed in the comparison section below.

The negative of the Spider4033 doesn't have an embedded antenna. Therefore, not only would we have to take up more space with having an external antenna but we would also have to budget that into our finances.

The Nano Spider 4400 from OriginGPS

The 4400 and the 4033 are fairly similar modules. They are comparable in many ways, however there are ways that they differ. The 4400 is a smaller package than the 4033, coming in at only 4.1mmX4.1mmx2.1mm. It weighs less than the 4033, but almost negligibly. It only utilizes the GPS, not GLONASS, BEIDOU, or GALILEO. It is slightly less sensitive (-163dBm instead of 4033's -165dBm). It has the same power consumption, accuracy, TTFF values as 4033. However, it has more interface connections available (4033 has only UART while 4400 also has SPI and I²C). It also utilizes a different processor (SiRFStar IV™ instead of MT3333) [CC].

The Nano Spider 4400 is basically equivalent to the 4033, in terms of what is necessary for our design. If it comes between the two, more research into specifications would be necessary, including but not limited to the cost/availability of the units.

More specifications are listed in the comparison chart.

The Nano Hornet 1411 from OriginGPS

The Hornet family of GPS GNSS receiver modules were designed to address markets where size, weight, stand-alone operation, power consumption, and design flexibility are all very important. The Hornet family, specifically, is the industry's smallest fully integrated GPS and GNSS modules with integrated antennas or on-board RF connectors. These specifications all meet the requirements that our design needs. In addition, the family of GNSS receivers have other benefits, including OriginGPS' NFZ technology (Noise Free Zone System, which allows high sensitivity and noise immunity, even in marginal signal conditions) as well as the shortest TTM (time to market), just requiring a connection to a power supply on a single layer PCB [T].

The Nano Hornet itself is a complete SiP (System in Package). This means that it only needs to be connected to a power supply and it can support its own functions. This allows the small size that the Nano Hornet states (10mm x 10mm) to remain small on our board, not requiring any external modules or connections besides the one to the MCU (Microcontroller Unit) and to the power supply. It is a complete SiP because it already has integrated modules on the chip, including OriginGPS proprietary low-profile GPS antenna, dual-stage LNA, RF LDO, SAW filter, TCXO, RTC and RF shield with market-leading SiRFstarIV™ GPS SoC (System on a Chip) [T]. These are also available on other OriginGPS modules, so they are not special to this module specifically. Their benefits are all described above in the other OriginGPS section.

What is fairly special about the Nano Hornet 1411 is that it also includes an active antenna on-board, meaning no external antenna are required to take up additional space on the board, as well as requiring research into which antenna will be functional with this module. This will make our design cleaner, as well as theoretically function more smoothly. This would be because the antenna

is actually designed and tested extensively with the 1411 module and has shown to function very well, the datasheet for 1411 itself stating that the on-board low profile antenna element is perfectly matched to receiver front end, frequency trimmed to GPS band, and Right-Hand Circularly Polarized [T]. Especially with the <2.5 accuracy that the 1411 module boasts, this would be a good fit with our product design.

SiRFstarIV™ is a Qualcomm® comprehensive navigation processor, utilizing ARM® technology [U]. The Nano Hornet provides real time positioning data in NMEA format by overlaying SBAS (Satellite Based Augmentation System), QZSS (Quasi-Zenith Satellite System), and other regional systems over the GPS. Some important other factors that this module boasts are Time to First Find (TTFF) in less than 1 second, and accuracy of 2 meters, and a sensitivity of -163dbm (enabling indoor tracking). These specifications are well within the range of our projects requirements [T].

The 1411 also boasts autonomous operation, which means that the integration into our particular design should take minimal effort from both the computer science group as well as the electrical group. The autonomous operation extends as well into self-managed low power modes. The first low power mode is ATP or Adaptive Trickle Power. This mode is good for applications that require navigation solutions at a fixed rate as well as low power consumption and the ability to track weak signals. This is very much within the requirements of our design, because our design requires a battery that will last long (low power consumption is necessary for this specification), navigation in areas with weak signals, and a generally fixed rate of signal acquisition. There is also a Push to Fix (PTF™) mode, for infrequent navigation solutions. The host sends a toggle to the ON/OFF pad to wake up the module and the mode will stay in Full power state until a position is estimated or timeout. The benefit of this to our design is that one of our possible operation states is that the GPS will be fixed on a motion toggle. Therefore, for some applications this may not be very frequent, and a state that will autonomously work with infrequent toggles would be perfect for this type of solution. Then there is the advanced power management (APM™) low power mode. This one intelligently cycles between full power mode and hibernate states, on a user-defined interval. In the application of our design where the user sets a GPS fix interval (for applications where motion toggling is not exactly likely, such as a stationary shipping container) [T].

Other significant features of this module that will benefit our design include selectable UART, SPI, or I²C interfaces (our design supports all 3 of these so having them as options is only beneficial to laying out the design of the PCB) which all include their own benefits to choosing them, Programmable baud and message rates, small size (10mmx 10mm, 3.8mm high, weight of 1.4g), and FCC, CE and VCCI certified [T].

There are no outright poor design qualities of this chip, so the only negatives will come from its comparison to other modules.

The Aardvark from Trimble

This product will be included in the comparison chart; however, it generally does not have anything worth describing here.

The UBX-M8230-CT from U-Blox

This product will be included in the comparison chart as well.

3.3.3.3 GPS Module Comparison

In Figure 32 below, many of the comparable, yet differing specifications between the GPS modules are shown. Within this figure the Time to First Fix will be discussed, specifically the cold and warm starts (the hot start times were similar within the modules and thus are not worth showing), the sensitivity of the module, how many channels the module had, the supply power of the modules, as well as the horizontal positional accuracy (which is very important as it is one of the major engineering/technical requirements that in section 2.5 Quality of House Analysis the engineers discussed in relation to the other engineering/technical requirements as well as marking requirements). These specifications are necessary to the design of our project, and so are a few others. The specifications that were comparable between the modules, but are still worth mentioning are also discussed in this section but were not worthy enough to be included in the table. This section will give an in-depth overview to the GPS specifications, and which device was ultimately chosen as the GPS module that the team will utilize in our design.

	TTFF		Sensitivity		Channels	Supply Power	Horizontal Position Accuracy
	Cold	Warm	Re-acquisition	Tracking			
	GPS	GPS	GPS + GLONASS	GPS		Typical	
FGPMMOPA6H	35 sec	33 sec	-163dBm	-165dBm	66	3.3V	3m
ORG4033	<23sec	<26sec	-160dBm	-165dBm	132	3.3V	2.5m
ORG1411	<35sec	<32 sec	-162dBm	-163dBm	48	1.8V	<2.5m
AARDVARK	<39sec	<35sec		-160dBm		3.3V	5m
UBX-M8230-CT	26sec		-160dBm	-167dBm	72		2m

Figure 32 - GPS Module Comparison Table

In Figure 32 above, the TTFF (Time To First Fix) are compared for the GPS modules that were investigated. TTFF is very important for both user experience as well as for the overall performance of the product. If the time to first fix takes too long, it will delay other portions of the product from working, it may timeout other parts of the code, as well as it simply could make a potential customer decide on a product other than ours. Based on the numbers above, the hot start times are comparable between the products, all within 1 second. The numbers that vary the most between modules is the cold and warm (if applicable) starts. The best performance in this category is ORG4033 module, boasting numbers significantly better than its counterparts. Within the other modules, the numbers are relatively comparable and likely would not even be noticeably different between them by the user.

In Figure 32, we now move the focus to the sensitivity of the GPS. The sensitivity indicates how weak an input signal can be and still be received by the module. This means that the lower the

number, the better/more sensitive the module is. The different specifications are the types of mode a module may be in, and how sensitive tracking will be during those. Generally, the module will either sit in tracking, for our application. Therefore, looking at those numbers, the UBX module boasts the best performance overall. However, all modules operate in a valid range for our product.

The next data that will be analyzed is the number of channels and the supply power, as seen above in Figure 32. The number of channels that a receiver has makes searching and tracking satellites much easier. While all receivers only need 4 satellites to get a GPS fix, a GPS will search as many as possible to get a fix as fast and as accurate as possible. Thus, with more channels it can search and procure satellite data from the theoretical “best” satellites for its location. Having more channels will therefore lead to faster satellite acquisition, lower power consumption, reduced likelihood of GPS fix, better sensitivity, and better accuracy. In this sense, the Multi Micro Spider would provide the highest number of channels, and therefore the best sense of this.

Moving forward to look at supply power, our system will have a high supply power of 3.7V, but voltages 3.0V and 3.1V will also be available to the modules (as designated in the design by the Electrical Engineers). Therefore, all modules operate in the proper range, but only the ORG1411 operates fully in the range.

Figure 32 does not depict power consumption, as not all of the modules reported comparable power consumption concepts, however power consumption is still an important component in the decision of what module we will choose. Taking this difference into consideration, the Nano Hornet 1411 documents very small numbers, in the microwatt range, for the power saving modes it has which is very appealing for our application. The Nano Hornet’s values can be found in it’s datasheet.

The horizontal position accuracy is compared in Figure 32. This is important for our application as one of the main specifications of our design is that the accuracy would be within 3 meters. The AARDVARK module is automatically eliminated as a potential option in this comparison as it does not fall within the accepted upon specification. All other options are still within the limits, though the PA6H does come very close to being in danger, as it lies right on the limit of accuracy.

The final value worthy of at least some discussion is the dimensions of the module, not specifically denoted in Figure 32. Our product needs to be smaller than the size of a credit card, as designated by our sponsor. However, this would be hard as some of the applications require additional external pieces, such as additional antennas. Where the AARDVARK and the PA6H are probably too big for our application, the rest fall into a relatively good range. However, the best by far seems to be the UBX, at remarkable dimensions for its capability.

Overall, in comparison the GPS modules that analytically are the best seem to be UBX, with the Nano Hornet a good contender as well.

3.3.3.4 GPS Final Decision

The team ultimately chose the OriginGPS Nano Hornet module. This is ultimately because the sponsor has designated it to be the one he wishes to use. Overall, it is a very capable module, with everything the product may require. He also has experience working with this module, which is

invaluable in a project like this. However, the UBX may be a good replacement, should the design team need one.

3.3.4 IMU

3.3.4.1 Introductory IMU Research

MPU-9250

An IMU allows a GPS receiver to work when GPS-signals are inaccessible. It is an electronic device that calculates a body's exact force, angular rate and occasionally the magnetic field surrounding the body. It uses a combination of accelerometers, gyroscopes and magnetometers," as shown in Figure 33 below [QQQ].

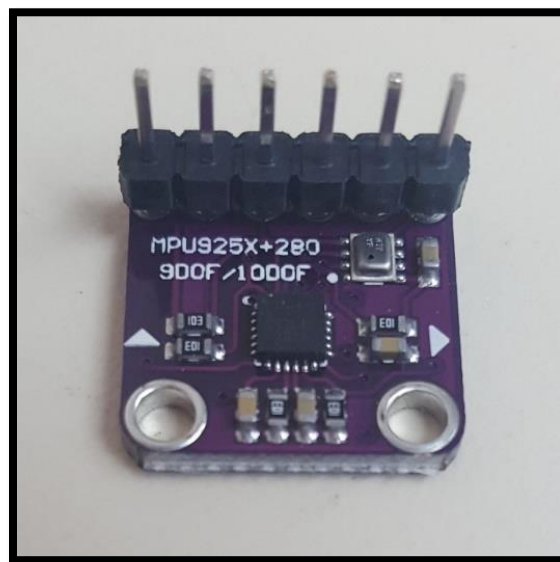


Figure 33 – MPU 9250

Taken from [QQQ], IMUs are typically integrated into the Inertial Navigation Systems, INS. The INS uses the raw IMU calculated data to find the attitude, angular rates, linear velocity and position relative to a global reference frame.

The IMU equipped INS forms the backbone for the navigation and control of many commercial and military vehicles such as submarines and even missiles. The data collected from the IMUs sensors allows a computer to track a position of an object. IMUs serve as orientation sensors and to measure motion.

The problem with using IMUs for navigation is that they typically suffer from accumulated error, which is something the team needs to consider when designing and testing the NAT device. This happens because the system is continually integrating acceleration with respect to time to calculate velocity and position, but any errors will accumulate which will eventually give wrong data.

IMUs designed to operate under harsh conditions are very often suspended by shock absorbers. These shock absorbers are required to master three effects which are to reduce the sensor errors due to mechanical environment solicitations, protect sensors as they can be damaged by shocks or vibrations and they contain parasitic IMU movement within a limited bandwidth, where processing will be able to compensate for them.

Suspended IMUs can offer very high performance, even when submitted to harsh environments, but it is necessary to compensate for three main resulting behaviors. These behaviors are coning, sculling and centrifugal acceleration effects [QQQ].

Taken from [PPP], the MPU-9250, a multi-chip module (MCM), contains two dies assimilated into a single Quad Flat No-Lead (QFN) package. One die holds the 3-Axis gyroscope and the 3-Axis accelerometer, while the other die contains the AK8963 3-Axis magnetometer. This is why the MPU-9250 is a 9-axis device that tracks motion because it combines a 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer and a Digital Motion Processor™ (DMP) all in a small 3x3x1mm package. The contents and size of this IMU is ideal for this project and is the reason we chose to implement it.

With the I2C sensor bus, the MPU-9250 provides complete 9-axis MotionFusion™ output. With its 9-axis integration, on-chip MotionFusion™, and run-time calibration firmware, the MPU-9250 device allows manufacturers to eliminate the costly and complex selection, qualification, and system level integration of discrete devices, ensuring optimal motion performance for customers. MPU-9250 is also designed to interface with multiple non-inertial digital sensors on its auxiliary I2C port.

MPU-9250 contains three 16-bit analog-to-digital converters (ADCs) for digitizing the gyroscope outputs, three 16-bit ADCs for digitizing the accelerometer outputs, and three 16-bit ADCs for digitizing the magnetometer outputs. For precision tracking of both fast and slow motions, the parts feature a user-programmable gyroscope full-scale range of ± 250 , ± 500 , ± 1000 , and ± 2000 degrees per second, a user-programmable accelerometer full-scale range of $\pm 2g$, $\pm 4g$, $\pm 8g$, and $\pm 16g$, and a magnetometer full-scale range of $\pm 4800\mu T$. The registers in the device communicate by either using I2C at 400kHz or SPI at 1MHz. The sensor and interrupt registers may be read using SPI at 20MHz for applications needing quicker communication.

By manipulating its patented and volume-proven CMOS-MEMS fabrication platform, which integrates MEMS wafers with companion CMOS electronics through wafer-level bonding, InvenSense brought the package size down to a footprint and thickness of 3x3x1mm, to provide a very small, high performance and less pricey package. The device brings strength by supporting 10,000g shock reliability [PPP].

Features of the Gyroscope, Accelerometer and Magnetometer

Taken from [PPP], the triple-axis MEMS gyroscope features include the digital-output X, Y and Z-Axis angular rate sensors (gyroscopes) with a user programmable full-scale range of ± 250 , ± 500 , ± 1000 and ± 2000 degrees/second and integrated 16-bit ADCs and a digitally-programmable low-pass filter. It also allows the gyroscope operating current to be 3.2mA, have a sleep mode current to be 8 μA , a factory calibrated sensitivity scale factor and to be self-tested.

The triple-axis MEMS accelerometer features include the digital-output triple-axis accelerometer with a programmable full scale range of $\pm 2g$, $\pm 4g$, $\pm 8g$ and $\pm 16g$ and integrated 16-bit ADCs and an accelerometer normal operating current to be $450\mu A$. It also has the low power accelerometer mode current to be $8.4\mu A$ at $0.98Hz$, $19.8\mu A$ at $31.25Hz$ and the sleep mode current to be $8\mu A$. It has user-programmable interrupts, a wake-on-motion interrupt for low power operation and is also self-tested.

The triple-axis MEMS Magnetometer features a 3-axis silicon monolithic Hall-effect magnetic sensor with magnetic concentrator, a wide dynamic measurement range and high resolution with lower current consumption and output data resolution of 14-bit ($0.6\mu T/LSB$) or 16-bit ($15\mu T/LSB$). It has a full scale measurement range of $\pm 4800\mu T$ and magnetometer normal operating current of $280\mu A$ at $8Hz$ repetition rate. It also has a self-test function with internal magnetic source to confirm magnetic sensor operation on end products.

Additional features include an auxiliary master I2C bus for reading data from external sensors, a $3.5mA$ operating current when all 9 motion sensing axes and the DMP are enabled, VDD supply voltage range of $2.4-3.6V$, VDDIO reference voltage for auxiliary devices, the smallest and thinnest QFN package for portable devices and minimal cross-axis sensitivity between the accelerometer, gyroscope and magnetometer axes. Some other features include a 512 byte FIFO buffer that enables the applications processor to read the data in bursts, digital-output temperature sensor and user-programmable digital filters for gyroscope, accelerometer and temp sensor $10,000g$ shock tolerant. Also, $400kHz$ Fast Mode I2C for communicating with all registers, $1MHz$ SPI serial interface for communicating with all registers, $200MHz$ SPI serial interface for reading sensor and interrupt registers, MEMS structure hermetically sealed and bonded at wafer level and is RoHS and Green compliant.

The Internal Digital Motion Processing (DMP) engine supports advanced Motion Processing and low power functions such as gesture recognition using programmable interrupts and the low-power pedometer functionality allows the host processor to sleep while the DMP maintains the step count [PPP].

The MPU-9250 is comprised of key blocks and functions. These include a three-axis MEMS rate gyroscope sensor with 16-bit ADCs and signal conditioning, a three-axis MEMS accelerometer sensor with 16-bit ADCs and signal conditioning and a three-axis MEMS magnetometer sensor with 16-bit ADCs and signal conditioning. It also contains the Digital Motion Processor (DMP) engine, the primary I2C and SPI serial communications interfaces, the auxiliary I2C serial interface for third party sensors, clocking and Sensor Data Registers. The MPU-9250 is FIFO, contains interrupts, a Digital-Output Temperature Sensor, gyroscope, accelerometer and magnetometer self-tests, bias and LDO and a charge pump [PPP].

Three-Axis MEMS Gyroscope, Accelerometer and Magnetometer with 16-bit ADCs and Signal Conditioning

The MPU-9250 is comprised of three gyroscopes. These gyroscopes rotate about the X, Y, and Z-axes. When the gyroscopes are rotated about any of the axes, a vibration is sensed. The signal is then amplified, demodulated and filtered to produce a voltage that is proportional to the angular rate. This voltage is digitized using individual on-chip 16-bit Analog-to-Digital Converters

(ADCs) to sample each axis. The full-scale range of the gyro sensors may be digitally programmed to ± 250 , ± 500 , ± 1000 , or ± 2000 degrees per second. The ADC sample rate is programmable from 8,000 samples per second, down to 3.9 samples per second, and user-selectable low-pass filters enable a wide range of cut-off frequencies.

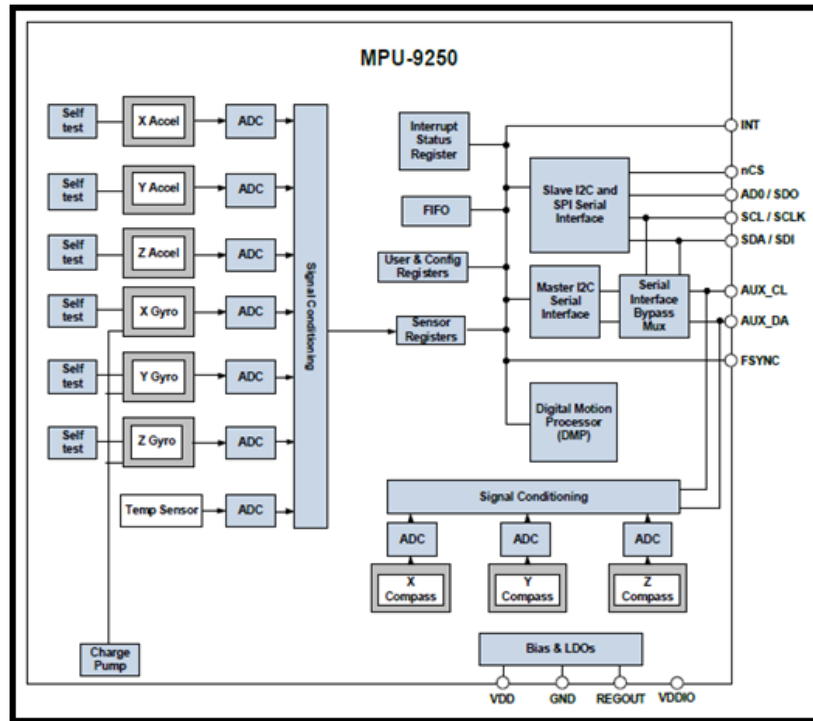


Figure 34 - MPU-9250 Block Diagram [PPP]

*Permission Requested [P16]

The MPU-9250's 3-Axis accelerometer uses separate proof masses for each axis. Acceleration along a specific axis causes displacement on the corresponding proof mass. The capacitive sensors detect the displacement differentially. The MPU-9250's architecture reduces the accelerometers' susceptibility to fabrication variations as well as to thermal drift. When the put on a flat surface, the device will measure 0g on the X and Y-axes and +1g on the Z-axis. The accelerometers' scale factor is calibrated at the factory and is nominally independent of supply voltage. Each sensor has a dedicated sigma-delta ADC for providing digital outputs. The full scale range of the digital output is adjusted to $\pm 2g$, $\pm 4g$, $\pm 8g$, or $\pm 16g$.

The 3-axis magnetometer uses highly sensitive Hall sensor technology. The magnetometer portion of the IC incorporates magnetic sensors for detecting terrestrial magnetism in the X-, Y-, and Z-Axes, a sensor driving circuit, a signal amplifier chain, and an arithmetic circuit for processing the signal from each sensor. Each ADC has a 16-bit resolution and a full scale range of $\pm 4800\mu T$ [PPP].

Digital Motion Processor

The embedded Digital Motion Processor (DMP) is located within the MPU-9250 and offloads computation of motion processing algorithms from the host processor. The DMP acquires data from accelerometers, gyroscopes, magnetometers and additional 3rd party sensors, and processes the data. The resulting data can be read from the DMP's registers, or can be buffered in a FIFO. The DMP has access to one of the MPU's external pins, which can be used for generating interrupts. This pin (pin 12) must be connected to a pin on the host processor that can wake the host from suspend mode.

The purpose of the DMP is to offload both timing requirements and processing power from the host processor. Typically, motion processing algorithms should be run at a high rate, often around 200Hz, in order to provide accurate results with low latency. This is required even if the application updates at a much lower rate; for example, a low power user interface may update as slowly as 5Hz, but the motion processing should still run at 200Hz. The DMP can be used as a tool in order to minimize power, simplify timing, simplify the software architecture, and save valuable MIPS on the host processor for use in the application [PPP].

Interfaces

As shown below in Figure 35, the MPU-9250 communicates to a system processor using either an SPI or an I2C serial interface. The MPU-9250 always acts as a slave when communicating to the system processor. The LSB of the I2C slave address is set by pin 9 (AD0) [PPP].

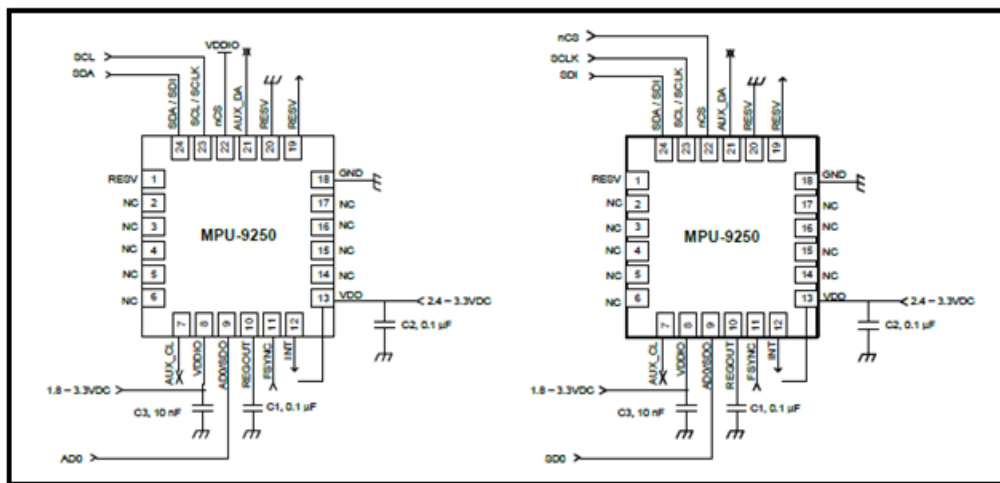


Figure 35 - MPU 9250: I2C operation, SPI operation (respectively) [PPP]

*Permission Requested [P16]

Auxiliary I2C Bus

The MPU-9250 has an auxiliary I2C bus for communicating to off-chip sensors. This bus has two operating modes, the I2C Master Mode and the Pass-Through Mode. The Master Mode in the MPU-9250 acts as a master to any external sensors and the bus has two operating auxiliary buses. The Pass-Through Mode in the MPU-9250 directly connects the primary and auxiliary buses

together allowing the system processor to directly communicate with any external sensors. AUX_DA and AUX_CL should be left unconnected if the Auxiliary mode is not used.

The Master Mode allows the MPU-9250 to directly access the data registers of external digital sensors, such as a magnetometer. In this mode, The MPU-9250 can be configured to perform burst reads, returning the following data from a magnetometer.

- X magnetometer – 2 Bytes
- Y magnetometer – 2 Bytes
- Z magnetometer – 2 Bytes

The I2C Master can be configured to read up to 24 bytes from up to 4 auxiliary sensors. A fifth sensor can be configured to work single byte read/write mode.

The Pass-Through Mode allows an external system processor to act as master and directly communicate to the external sensors connected to the auxiliary I2C bus pins (AUX_DA and AUX_CL). In this mode, the auxiliary I2C bus control logic (3rd party sensor interface block) of the MPU-9250 is disabled, and the auxiliary pins AUX_DA and AUX_CL are connected to the main bus through analog switches internally. The Pass-Through mode is useful for configuring the external sensors, or for keeping the MPU-9250 in a low-power mode when only the external sensors are used. In this mode, they system processor can still access MPU-9250 data through the interface. The Pass-Through mode is also used to access the AK8963 magnetometer directly from the host. In this configuration, the slave address for the AK8963 is 0XDC or 12 decimal.

Self-test allows for the testing of the mechanical and electrical portions of the sensors. The self-test for each measurement axis can be activated by means of the gyroscope and accelerometer self-test registers (registers 13 to 16). When the self-test is activated, the electronics cause the sensors to be actuated and produce an output signal. The output signal is used to observe the self-test response.

The self-test response is defined as:

Self-test response = sensor output with self-test enabled – sensor output without self-test enabled

When the value of the self-test response is within the appropriate limits, the part has passed self-test. When the self-test response exceeds the appropriate values, the part is deemed to have failed self-test. It is recommended to use InvenSense MotionApps software for executing self-test. Further details, including the self-test limits are included in the MPU-9250 Self-Test applications note available from InvenSense [PPP].

MPU-9250 Solutions

The system processor is an I2C master to the MPU-9250. In addition, the MPU-9250 is an I2C master to the optional external 3rd party sensor. The MPU-9250 has limited capabilities as an I2C Master, and depends on the system processor to manage the initial configuration of any auxiliary sensors. The MPU- 9250 has an interface bypass multiplexer, which connects the system processor I2C bus (SDA and SCL) directly to the auxiliary sensor I2C bus (AUX_DA and AUX_CL).

Once the auxiliary sensors have been configured by the system processor, the interface bypass multiplexer should be disabled so that the MPU-9250 auxiliary I2C master can take control of the sensor I2C bus and gather data from the auxiliary sensors. The INT pin should be connected to a GPIO on the system processor that can wake the system from suspend mode.

The system processor is a SPI master to the MPU-9250. The CS, SDO, SCLK, and SDI signals are used for SPI communications. Because these SPI pins are shared with the I2C slave pins, the system processor cannot access the auxiliary I2C bus through the interface bypass multiplexer, which connects the processor I2C interface pins to the sensor I2C interface pins. Since the MPU-9250 has limited capabilities as an I2C Master, and depends on the system processor to manage the initial configuration of any auxiliary sensors, another method must be used for programming the sensors on the auxiliary sensor I2C bus (AUX_DA and AUX_CL). When using SPI communications between the MPU-9250 and the system processor, configuration of devices on the auxiliary I2C sensor bus can be achieved by using I2C Slaves 0-4 to perform read and write transactions on any device and register on the auxiliary I2C bus. The I2C Slave 4 interface can be used to perform only single byte read and write transactions.

Once the external sensors have been configured, the MPU-9250 can perform single or multi-byte reads using the sensor I2C bus. The read results from the Slave 0-3 controllers can be written to the FIFO buffer as well as to the external sensor registers. The INT pin should be connected to a GPIO on the system processor capable of waking the processor from suspend [PPP].

Clocking

The MPU-9250 has a flexible clocking scheme, allowing a variety of internal clock sources to be used for the internal synchronous circuitry. This synchronous circuitry includes the signal conditioning and ADCs, the DMP, and various control circuits and registers. An on-chip PLL provides flexibility in the allowable inputs for generating this clock. Allowable internal sources for generating the internal clock are an internal relaxation oscillator and any of the X, Y, or Z gyros (MEMS oscillators with a variation of $\pm 1\%$ over temperature)

Selection of the source for generating the internal synchronous clock depends on the requirements for power consumption and clock accuracy. These requirements will most likely vary by mode of operation. For example, in one mode, where the biggest concern is power consumption, the user may wish to operate the Digital Motion Processor of the MPU-9250 to process accelerometer data, while keeping the gyros off. In this case, the internal relaxation oscillator is a good clock choice. However, in another mode, where the gyros are active, selecting the gyros as the clock source provides for a more accurate clock source. Clock accuracy is important, since timing errors directly affect the distance and angle calculations performed by the Digital Motion Processor (and by extension, by any processor). There are also start-up conditions to consider. When the MPU-9250 first starts up, the device uses its internal clock until programmed to operate from another source. This allows the user, for example, to wait for the MEMS oscillators to stabilize before they are selected as the clock source [PPP].

Registers, Sensors, Interrupts

The sensor data registers contain the latest gyroscope, accelerometer, magnetometer, auxiliary sensor, and temperature measurement data. They are read-only registers, and are accessed via the serial interface. Data from these registers may be read anytime.

The MPU-9250 contains a 512-byte FIFO register that is accessible via the Serial Interface. The FIFO configuration register determines which data is written into the FIFO. Possible choices include gyro data, accelerometer data, temperature readings, auxiliary sensor readings, and FSYNC input. A FIFO counter keeps track of how many bytes of valid data are contained in the FIFO. The FIFO register supports burst reads. The interrupt function may be used to determine when new data is available.

Interrupt functionality is configured via the Interrupt Configuration register. Items that are configurable include the INT pin configuration, the interrupt latching and clearing method, and triggers for the interrupt. Items that can trigger an interrupt are (1) Clock generator locked to new reference oscillator (used when switching clock sources); (2) new data is available to be read (from the FIFO and Data registers); (3) accelerometer event interrupts; and (4) the MPU-9250 did not receive an acknowledge from an auxiliary sensor on the secondary I2C bus. The interrupt status can be read from the Interrupt Status register. The INT pin should be connected to a pin on the host processor capable of waking that processor from suspend.

Mode	Name	Gyroscope	Accelerometer	Magnetometer	DMP
1	Sleep Mode	Off	Off	Off	Off
2	Standby Mode	Drive On	Off	Off	Off
3	Low-Power Accelerometer Mode	Off	Duty-Cycled	Off	On or Off
4	Low-Noise Accelerometer Mode	Off	On	Off	On or Off
5	Gyroscope Mode	On	Off	Off	On or Off
6	Magnetometer Mode	Off	Off	On	On or Off
7	Accelerometer + Gyroscope Mode	On	On	Off	On or Off
8	Accelerometer + Magnetometer Mode	Off	On	On	On or Off
9	9-Axis Mode	On	On	On	On or Off

Figure 36 - MPU-9250: The User-Accessible Power Modes [PPP]

An on-chip temperature sensor and ADC are used to measure the MPU-9250 die temperature. The readings from the ADC can be read from the FIFO or the Sensor Data registers. The bias and LDO section generates the internal supply and the reference voltages and currents required by the MPU-

9250. Its two inputs are an unregulated VDD and a VDDIO logic reference supply voltage. The LDO output is bypassed by a capacitor at REGOUT. An on-chip charge pump generates the high voltage required for the MEMS oscillators [PPP].

Wake-on-Motion Interrupt

The MPU-9250 provides motion detection capability. A qualifying motion sample is one where the high passed sample from any axis has an absolute value exceeding a user-programmable threshold. The following flowchart explains how to configure the Wake-on-Motion Interrupt. In order to properly enable motion interrupts, the INT pin should be connected to a GPIO on the system processor that is capable of waking up the system processor [PPP].

Digital Interface

The internal registers and memory of the MPU-9250 can be accessed using either I2C at 400 kHz or SPI at 1MHz. SPI operates in four-wire mode. With the I2C Interface, the I2C is a two-wire interface comprised of the signals serial data (SDA) and serial clock (SCL). In general, the lines are open-drain and bi-directional. In a generalized I2C interface implementation, attached devices can be a master or a slave. The master device puts the slave address on the bus, and the slave device with the matching address acknowledges the master. The MPU-9250 always operates as a slave device when communicating to the system processor, which thus acts as the master. SDA and SCL lines typically need pull-up resistors to VDD. The maximum bus speed is 400 kHz. The slave address of the MPU-9250 is b110100X which is 7 bits long. The LSB bit of the 7 bit address is determined by the logic level on pin AD0. This allows two MPU-9250s to be connected to the same I2C bus. When used in this configuration, the address of the one of the devices should be b1101000 (pin AD0 is logic low) and the address of the other should be b1101001 (pin AD0 is logic high).

With the SPI Interface, SPI is a 4-wire synchronous serial interface that uses two control lines and two data lines. The MPU-9250 always operates as a Slave device during standard Master-Slave SPI operation. With respect to the Master, the Serial Clock output (SCLK), the Serial Data Output (SDO) and the Serial Data Input (SDI) are shared among the Slave devices. Each SPI slave device requires its own Chip Select (CS) line from the master. CS goes low (active) at the start of transmission and goes back high (inactive) at the end. Only one CS line is active at a time, ensuring that only one slave is selected at any given time. The CS lines of the non-selected slave devices are held high, causing their SDO lines to remain in a high-impedance (high-z) state so that they do not interfere with any active devices [PPP].

The SPI Operational Features:

1. Data is delivered MSB first and LSB last
2. Data is latched on the rising edge of SCLK
3. Data should be transitioned on the falling edge of SCLK
4. The maximum frequency of SCLK is 1MHz
5. SPI read and write operations are completed in 16 or more clock cycles (two or more bytes). The first byte contains the SPI Address, and the following byte(s) contain(s) the SPI data. The first bit of the first byte contains the Read/Write bit and indicates the

- Read (1) or Write (0) operation. The following 7 bits contain the Register Address. In cases of multiple-byte Read/Writes, data is two or more bytes:
- Supports Single or Burst Read/Writes.

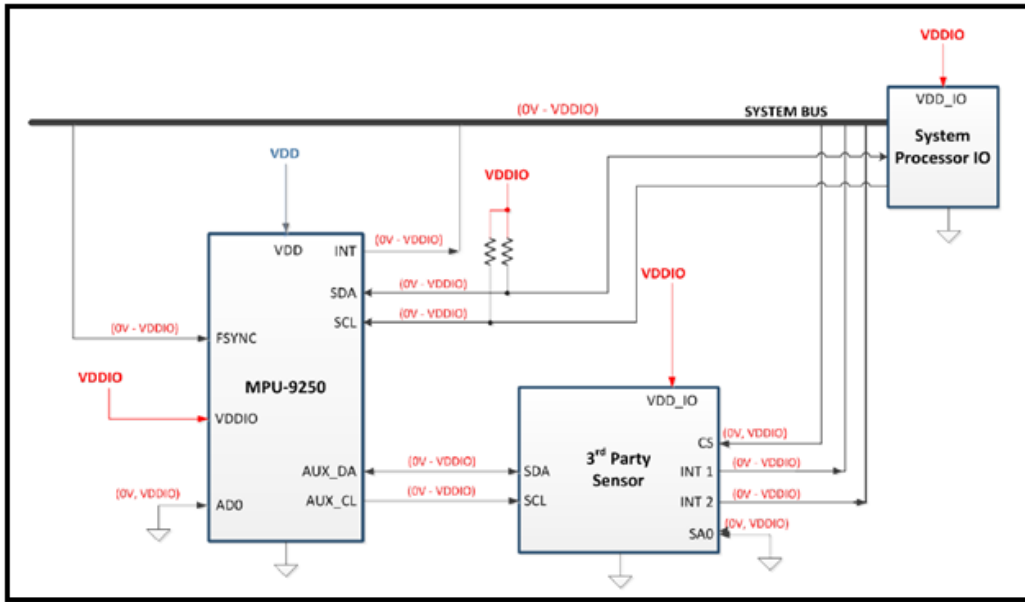


Figure 37 - I/O Levels and Connections [PPP]

*Permission Requested [P16]

Assembly

The orientation of the axes of sensitivity and the polarity of rotation are shown in the diagrams in Figures 38 and 39:

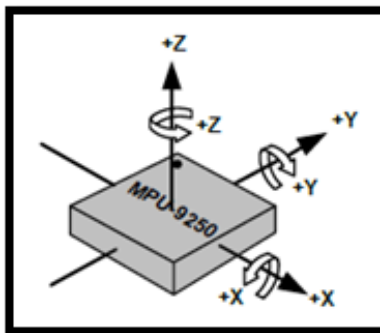


Figure 38 - Orientation of Axes of Sensitivity and Polarity of Rotation

*Permission Requested [P16]

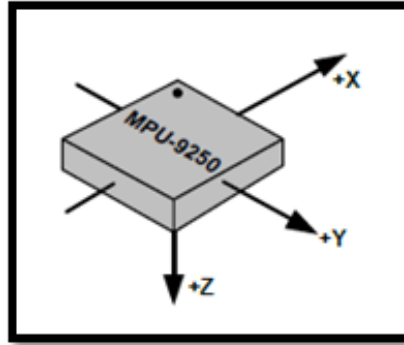


Figure 39 - Orientation of Axes of Sensitivity for Compass

*Permission Requested [P16]

Future Uses

Typically, IMUs are integrated into INS, Inertial Navigation Systems. When IMU is equipped with INS, it becomes an important support for navigation and control of commercial and military vehicles. INS is outside the scope of this project, but could be implemented in a future project. Angular rates, linear velocity and position are some of the measurements that can be calculated. IMUs also serve as orientation sensors and to measure motion.

The major disadvantage of using IMUs for navigation, and a consideration for the group with this project, is that it accumulates error over time, leading to drift. Drift is "an ever-increasing difference between where the system think it is located and the actual location of it." The error is due to the constant integration of acceleration with respect to time, which calculates velocity and position [QQQ].

3.3.4.2 The IMU Modules

Types of Accelerometers

Acceleration is calculated in either units of meters per second squared (m/s^2), or G-force (g), which is about $9.8m/s^2$. Accelerometers are used to sense both static (e.g. gravity) and dynamic (e.g. sudden starts/stops) acceleration. One of the more extensively used applications for accelerometers is tilt-sensing. Because they are affected by the acceleration of gravity, an accelerometer can display how it is oriented with respect to the Earth's surface. An accelerometer can also be used to sense motion. For example, an accelerometer in Nintendo's WiiMote can be used to sense emulated forehands and backhands of a tennis racket, or rolls of a bowling ball. Finally, an accelerometer can also be used to sense if a device is in a state of free fall. This feature is implemented in several hard drives: if a drop is sensed, the hard drive is quickly switched off to protect against data loss [PPP]. WiiMote can be used to sense emulated forehands and backhands of a tennis racket, or rolls of a bowling ball. Finally, an accelerometer can also be used to sense if a device is in a state of free fall. This feature is implemented in several hard drives: if a drop is sensed, the hard drive is quickly switched off to protect against data loss [PPP].

Characteristics

According to [PPP], the characteristics include:

- **Range:** The upper and lower limits of what the accelerometer can measure is also known as its range. In most cases, a smaller full-scale range means a more sensitive output; so you can get a more precise reading out of an accelerometer with a low full-scale range. You want to select a sensing range that will best fit your project, if your project will only be subjected to accelerations between +2g and -2g, a $\pm 250g$ -ranged accelerometer won't give you much, if any, precision. There is a good assortment of accelerometers, with maximum ranges stretching from $\pm 1g$ to $\pm 250g$. Most of our accelerometers are set to a hard maximum/minimum range, however some of the fancier accelerometers feature selectable ranges.
- **Interface:** This is another one of the more important specifications. Accelerometers will have either an analog, pulse-width modulated (PWM), or digital interface.
 - Accelerometers with an analog output will produce a voltage that is directly proportional to the sensed acceleration. At 0g, the analog output will usually reside at about the middle of the supplied voltage (e.g. 1.65V for a 3.3V sensor). Generally, this interface is the easiest to work with, as analog-to-digital converters (ADCs) are implemented in most microcontrollers.
 - Accelerometers with a PWM interface will produce a square wave with a fixed frequency, but the duty cycle of the pulse will vary with the sensed acceleration. These are pretty rare; we've only got one in our catalog.
 - Digital accelerometers usually feature a serial interface be it SPI or I²C. Depending on your experience, these may be the most difficult to get integrated with your microcontroller. That said, digital accelerometers are popular because they usually have more features, and are less susceptible to noise than their analog counterparts.
- **Number of axes measured:** Out of the three axes possible (x, y, and z), how many can the accelerometer sense? Three-axis accelerometers are usually the way to go; they are the most common and they are really no more expensive than equivalently sensitive one or two axis accelerometers.
- **Power usage:** If the project is battery powered, it should be considered how much power the accelerometer will consume. The required current consumption will usually be in the 100s of μA range. Some sensors also feature sleep functionality to conserve energy when the accelerometer isn't needed.
- **Bonus features:** Many more recently developed accelerometers may have a few nifty features, beyond just producing acceleration data. These newer accelerometers may include features like selectable measurement ranges, sleep control, 0-g detection, and tap sensing.

SparkFun Triple Axis Accelerometer Breakout – ADXL335

Description

This is the breakout board for the 3 axis ADXL335, pictured below in Figure 40, from Analog Devices. This is the latest in a long, proven line of analog sensors - the holy grail of accelerometers. The ADXL335 is a triple axis MEMS accelerometer with extremely low noise and power consumption - only 320A! The sensor has a full sensing range of $\pm 3g$. There is no on-board

regulation, provided power should be between 1.8 and 3.6VDC. Board comes fully assembled and tested with external components installed. The included 0.1uF capacitors set the bandwidth of each axis to 50Hz [VVV].

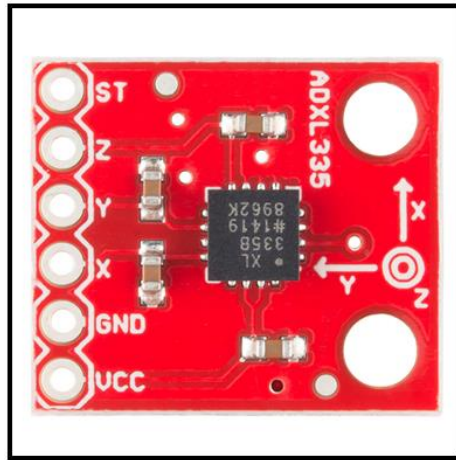


Figure 40 - ADXL335 [VVV]

*Permission Requested [P17]

SparkFun Triple Axis Accelerometer Breakout – ADXL345



Figure 41 - ADXL345 [WWW]

*Permission Requested [P17]

Description

This new version adds 2 standoff holes as well as an extra decoupling capacitor. The ADXL345, pictured above in Figure 41, is a small, thin, low power, 3-axis MEMS accelerometer with high resolution (13-bit) measurement at up to +16 g. Digital output data is formatted as 16-bit two's complement and is accessible through either a SPI (3- or 4-wire) or I2C digital interface.

The ADXL345 is well suited to measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (4 mg/LSB) enables measurement of inclination changes less than 1.0 degrees.

Several special sensing functions are provided. Activity and inactivity sensing detect the presence or lack of motion and if the acceleration on any axis exceeds a user-set level. Tap sensing detects single and double taps. Free-fall sensing detects if the device is falling. These functions can be mapped to one of two interrupt output pins. An integrated, patent pending 32-level first in, first out (FIFO) buffer can be used to store data to minimize host processor intervention. Low power modes enable intelligent motion-based power management with threshold sensing and active acceleration measurement at extremely low power dissipation [WWW].

Features

The features of the ADXL345 include the 2.0-3.6VDC supply voltage, its ultra low-power of 40uA in measurement mode, 0.1uA in standby at 2.5V, its tap and double tap detection, its free-fall detection and its SPI and I2C interfaces [WWW].

SparkFun Triple Axis Accelerometer Breakout – LIS331

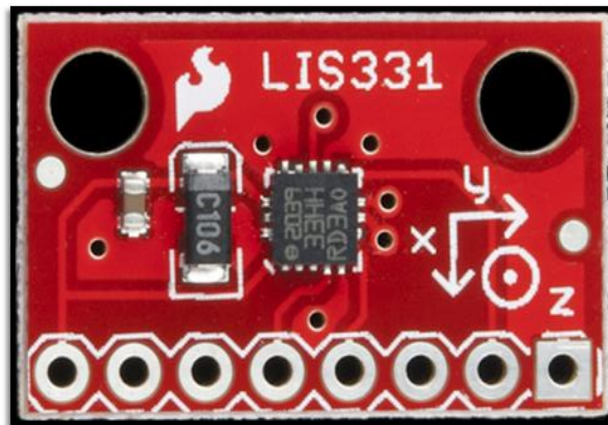


Figure 42 - LIS331 [XXX]

*Permission Requested [P17]

Description

The ultra-low-power LIS331HH, pictured above in Figure 42, has a three axis linear accelerometer. has a digital I2C/SPI serial interface which makes it ideal for using in embedded applications. All major pins are broken out to 0.1" spaced headers. It also includes two mounting holes as well.

The LIS331HH is an ultra low-power full-scale three axis MEMS linear accelerometer. It has a digital I2C/SPI serial interface which makes it ideal for using in embedded applications. The device also features ultra low-power operational modes that allow advanced power saving and smart sleep to wake-up functions.

The LIS331HH has dynamically user selectable full scales of $\pm 6g/\pm 12g/\pm 24g$ and is capable of measuring accelerations with output data rates from 0.5 Hz to 1kHz. The self-test capability allows the user to check the functioning of the sensor in the final application [XXX].

Features

The features of the LIS331 include the ultra low-current mode, down to 10uA, a 2.16-3.6V input, the 6g/12g/24g selectable, I2C and SPI digital output and a 16 bit data output [XXX].

Types of Gyroscopes

Gyroscopes measure angular velocity, how fast something is spinning about an axis. If you're trying to monitor the orientation of an object in motion, an accelerometer may not give you enough information to know exactly how it's oriented. Unlike accelerometers gyros are not affected by gravity, so they make a great complement to each other. You'll usually see angular velocity represented in units of rotations per minute (RPM), or degrees per second ($^{\circ}/s$). The three axes of rotation are either referenced as x, y, and z, or roll, pitch, and yaw [PPP].

In the past, gyros have been used for space navigation, missile control, under-water guidance, and flight guidance. Now they are starting to be used alongside accelerometers for applications like motion-capture and vehicle navigation [PPP].

A lot of what was considered when selecting an accelerometer still applies to selecting the perfect gyroscope:

- **Range:** Make sure the maximum angular velocity you're expecting to measure does not exceed the maximum range of the gyroscope, but also, in order to get the best possible sensitivity, make sure your gyroscope's range is not much greater than what you are expecting.
- **Interface:** There is actually not much diversity in this section. About 95% of the gyroscopes feature an analog output. There are a few that have a digital interface, either SPI or I2C.
- **Number of axes measured:** Compared to accelerometers, gyroscopes are a little behind the curve. Only recently have inexpensive, 3-axis gyroscopes begun to appear on the market. Most of the gyroscopes are either 1 or 2-axis. When selecting those, you need to pay attention to which of the three axes the gyroscope will measure. Some 2 axis gyroscopes will measure pitch and roll, while others measure pitch and yaw.
- **Power usage:** If the project is battery powered, consider how much power the gyroscope will consume. The required current consumption will usually be in the 100s of μA range. Some sensors also feature sleep functionality to conserve energy when the gyroscope is not needed.
- **Bonus features:** Many gyroscopes feature a temperature output, which is very useful when compensating for drift.

SparkFun Tri-Axis Gyro Breakout – L3G4200D

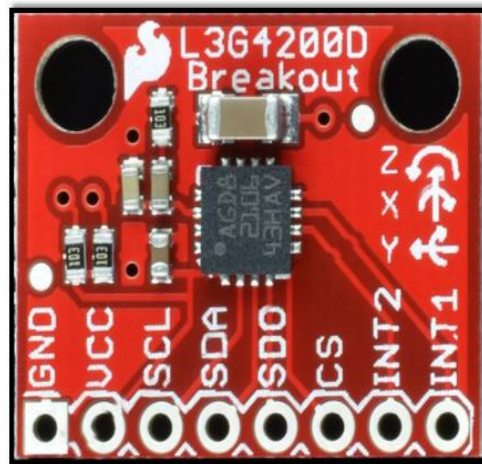


Figure 43 - L3G4200D [YYY]

*Permission Requested [P17]

Description

This is a breakout board for the L3G4200D, pictured above in Figure 43, low-power three-axis angular rate sensor. The L3G4200D is a MEMS motion sensor and has a full scale of $\pm 250/\pm 500/\pm 2000$ dps and is capable of measuring rates with a user-selectable bandwidth. These work great in gaming and virtual reality input devices, GPS navigation systems and robotics [YYY].

Features

The features of the L3G4200D include the three selectable full scales (250/500/2000 degrees per second), the I2C and SPI digital output interface, the 16 bit-rate value data output, the 8-bit temperature data output, has a wide supply voltage of 2.4 V to 3.6 V, low voltage-compatible IOs (1.8 V), embedded power-down and sleep mode, embedded temperature sensor and has high shock survivability [YYY].

IMUs

Gyroscopes and accelerometers are great, but alone they do not give you enough information to be able to comfortably calculate things like orientation, position, and velocity. To measure those and other variables many people combine the two sensors, to create an inertial measurement unit (IMU) which provides two to six degrees of freedom (DOF). IMUs are widely used in devices that require knowledge of their exact position, for example robotic arms, guided missiles, and tools used in the study of body motion.

SparkFun's IMUs can really be broken down into two classes: simple IMU combo boards, which just mount an accelerometer and gyro onto a single PCB, and more complex units that interface a microcontroller with the sensors to produce a serial output. If you've glanced over the previous sections, you should know what kind of specifications to be looking for in IMUs: the number of

axes (both for the accelerometer and gyro), the measuring range of the sensors, and the interface [CCCC].

SparkFun 6 Degrees of Freedom IMU Digital Combo Board – ITG3200/ADXL345

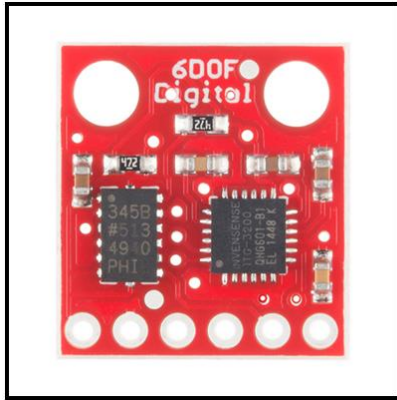


Figure 44 - ITG3200 / ADXL345 [ZZZ]
*Permission Requested [P17]

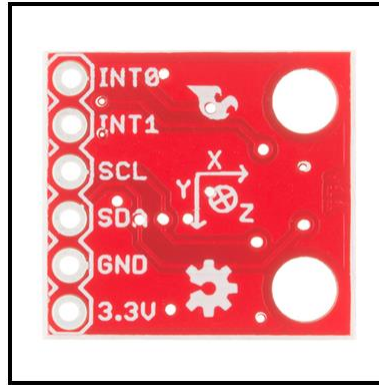


Figure 45 - ITG3200 / ADXL345 [ZZZ]
*Permission Requested [P17]

Description

This is a simple breakout for the ADXL345 accelerometer and the ITG-3200 MEMS gyro, pictured above in Figure 44 and Figure 45. With this board, there is a full 6 degrees of freedom. The sensors communicate over I2C and one INT output pin from each sensor is broken out. If you need a simple and tiny board that gives you 6 degrees of freedom, this would be a good choice [ZZZ].

Features

The features of the ADXL345 include its tiny size, its two mounting holes, the ADXL345 accelerometer, the ITG-3200 gyroscope, 3.3V input and the I2C interface [ZZZ].

SparkFun Triple Axis Accelerometer and Gyro Breakout – MPU 6050

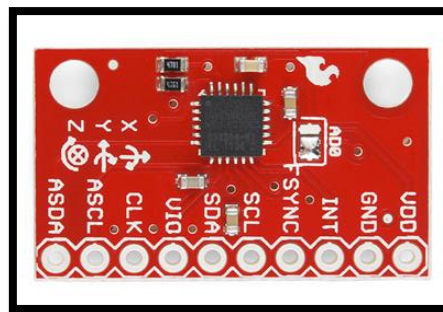


Figure 46 - MPU 6050 [AAAA]
*Permission Requested [P17]

Description

The MPU-6050, pictured above in Figure 46, is a serious little piece of motion processing tech. By combining a MEMS 3-axis gyroscope and a 3-axis accelerometer on the same silicon die together with an onboard Digital Motion Processor™ (DMP™) capable of processing complex 9-axis MotionFusion algorithms, the MPU-6050 does away with the cross-axis alignment problems that can creep up on discrete parts.

The breakout board for the MPU-6050 makes this tiny QFN package easy to work into your project. Every pin you need to get up and running is broken out to 0.1" headers, including the auxiliary master I2C bus which allows the MPU-6050 to access external magnetometers and other sensors [AAAA].

Features

The features of the MPU 6050 include the I2C Digital-output of 6 or 9-axis MotionFusion data in rotation matrix, quaternion, Euler Angle or raw data format, has an input voltage of 2.3 -3.4V, selectable solder jumpers on CLK, FSYNNC and AD0, tri-axis angular rate sensor (gyro) with a sensitivity up to 131 LSBs/dps and a full-scale range of ± 250 , ± 500 , ± 1000 and ± 2000 dps, tri-axis accelerometer with a programmable full scale range of $\pm 2g$, $\pm 4g$, $\pm 8g$ and $\pm 16g$, Digital Motion Processing (DMP) engine offloads complex MotionFusion, sensor timing synchronization and gesture detection. embedded algorithms for run-time bias and compass calibration and digital-output temperature sensor [AAAA].

3.3.4.3 IMU Comparison and Selection

	Axis	Noise/Power Consumption	Voltage	Interface
Accelerometers				
ADXL335	Triple	Low Noise & Low Power (320A)	1.8-3.6VDC	---
ADXL345	Triple	Low Power (40 μ A)	2.0-3.6VDC	Either an SPI or I2C Digital Interface
LIS331	Triple (linear)	Ultra-Low Power (10 μ A)	2.16-3.6V	Digital I2C/SPI Serial Interface
Gyroscopes				
L3G4200D	Triple	Low Power	2.4-3.6V	I2C and SPI Digital Interface
Combo Boards				
ITG3200/ADXL345 Accelerometer and Gyroscope	Triple	---	3.3V	I2C Interface
MPU-6050 Accelerometer and Gyroscope	Triple	Low Power	2.2-3.4V	I2C Digital-Output Interface
MPU-9250 Accelerometer, Gyroscope and Magnetometer	Triple	Low Power and Low Noise	2.4-3.6V	I2C or SPI Interface

Figure 47 - IMU Comparison Chart

The MPU-9250 is a 9-axis device that tracks motion because it combines a 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer and a Digital Motion Processor™ (DMP) all in a small 3x3x1mm package. The contents and size of this IMU is ideal for this project and is the reason we chose to implement it.

In the IMU comparison chart in Figure 47, it is shown that not only is the MPU-9250 the only option with the accelerometer, gyroscope, and magnetometer, but it also has the interfaces that are available through the choice of MCU, as well as the voltage range that the NAT device operates on. It is the best choice for what the NAT devices needs, especially due to its implementation of all 3 motion readings. This will be necessary for all motion triggered values, including the power cycles as well as the motion trigger location services.

3.3.5 HID

Overview

Human Interface Device, HID, is a type of computer device that takes in an input and gives an output that is acceptable to humans. The HID standard was implemented mainly to allow innovation in and simplify the process of installing PC input devices. As shown in Figure 48 and Figure 49 below, the host communicates with the device. It receives input information from the device. The device is what a human will interact with. It typically comes in the form of a mouse or keyboard.

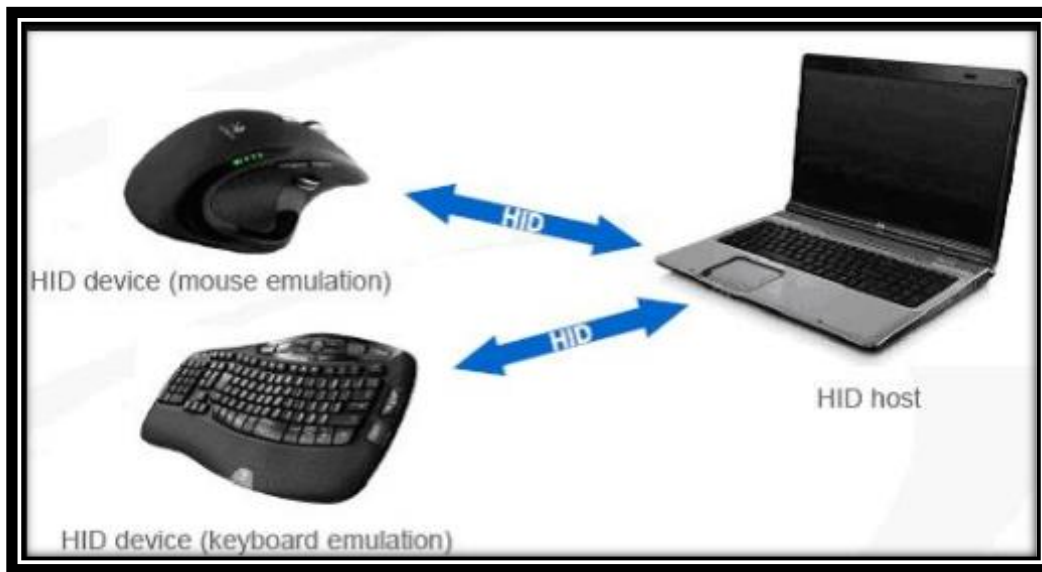


Figure 48 - HID Device and Host Communication [BBBB]

*Permission Requested [P20]

Taken from [OOO], the Human Interface Device (HID) class specification allows designers to create USB-based devices and applications without the need for custom driver development. Their high levels of on-chip integration and robust USB interfaces make Silicon Laboratories microcontrollers ideal devices for HID designs. USB devices communicate with PCs. Creating a

USB interface between an embedded system and a PC requires the writing of code for the following software subsystems:

- Embedded device firmware
- Host-side operating system drivers
- Host-side PC application

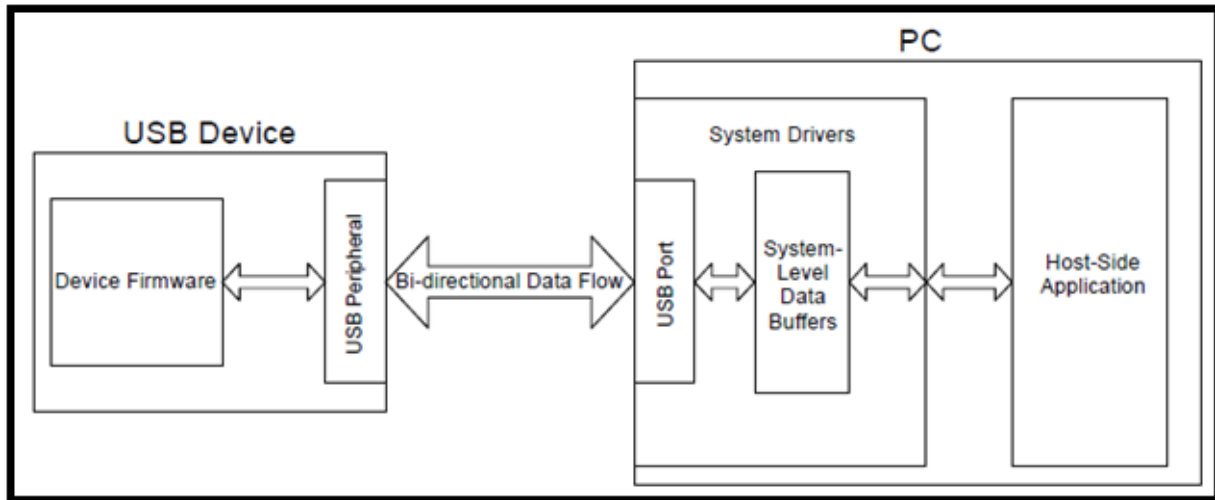


Figure 49 - USB Interface between a PC and an Embedded System

*Permission Requested [P9]

Universal Serial Bus

The USB protocol presents significant advantages over other PC interfaces in versatility, speed, and reliability. USB systems communicate under device/host relationships where a device is attached to a USB port of a host PC or a hub that is then connected to a PC. Host-side application software interacts with device-side firmware through the native operating system or customized drivers [000].

Device Endpoints

In USB-based systems, all data travels to or from device endpoints. All devices have a control endpoint, which is required as part of the USB specification. The host uses this endpoint to find information about the device through descriptors, which are data packets. Many USB devices also support additional endpoints. These transfer data to and from the host. IN endpoints transfer data from the device to the host while OUT endpoints transfer data from the host to the device [000].

Silicon Laboratories Microcontroller Capabilities

Silicon Laboratories microcontroller families with USB functionality can support a control endpoint and at least one additional endpoint. USB hardware controls low-level data transfer to and from the host. The hardware sends and receives data through user-accessible buffers. The microcontroller signals firmware about USB events, including data reception and transmission-

related events, by setting flags. These flags trigger the servicing of an interrupt service routine (ISR) if interrupts have been enabled [OOO].

USB Device Classes

The USB specification and supplemental documents define a number of device classes that categorize USB devices according to capability and interface requirements. When a host retrieves device information, class classification helps the host determine how to communicate with the USB device [OOO].

HID Class

The HID class devices usually interface with humans in some capacity. HID-class devices include mice, keyboards, printers, etc. However, the HID specification merely defines basic requirements for devices and the protocol for data transfer, and devices do not necessarily depend on any direct human interaction [OOO].

Requirements

Taken from [OOO], HID devices must meet a few general requirements that are imposed to keep the HID interface standardized and efficient.

- All HID devices must have a control endpoint (Endpoint 0) and an interrupt IN endpoint. Many devices also use an interrupt OUT endpoint. In most cases, HID devices are not allowed to have more than one OUT and one IN endpoint.
- All data transferred must be formatted as reports whose structure is defined in the report descriptor.
- HID devices must respond to standard HID requests in addition to all standard USB requests.

Enumeration and Device Detection

Before the HID device can enter its normal operating mode and transfer data with the host, the device must properly enumerate. The enumeration process consists of a number of calls made by the host for descriptors stored in the device that describe the device's capabilities. The device must respond with descriptors that follow a standard format. Descriptors contain all basic information about a device.

The USB specification defines some of the descriptors retrieved, and the HID specification defines other required descriptors. The next section discusses the descriptor structure a host expects to receive. The two sections after that describe the responsibilities of the device and the host during enumeration. These sections refer to sections of the HID firmware template, which is discussed in detail later in this document [OOO].

Descriptor Structure

Descriptors begin with a byte describing the descriptor length in bytes. This length equals the total number of bytes in the descriptor including the byte storing the length. The next byte indicates the

descriptor type, which allows the host to correctly interpret the rest of the bytes contained in the descriptor.

The content and values of the rest of the bytes are specific to the type of descriptor being transmitted. Descriptor structure must follow specifications exactly; the host will ignore received descriptors containing errors in size or value, potentially causing enumeration to fail and prohibiting further communication between the device and the host. Descriptor contents are typically stored in the Flash/EEPROM memory space. The file named *USB_Descriptor.h* in the HID firmware template declares each value of every descriptor. The file, *USB_Descriptor.c*, defines the contents for each descriptor [OOO].

Device Responsibilities during Enumeration

A device's main responsibility during enumeration is to respond to requests for information made by the host system in a timely and accurate manner. The device transfers all enumeration data across the control endpoint. In the firmware template, this endpoint is handled during execution of the USB ISR [OOO].

The Control Endpoint Handler

The USB ISR examines USB registers to determine the cause of the interrupt. If the ISR finds that an Endpoint 0 transaction caused the interrupt, the ISR calls the control endpoint handler. The Endpoint 0 handler parses the Setup Packet sent by the host and stored in the Endpoint 0 buffer to determine what standard USB request has been made by the host system. The handler then calls the appropriate function. The firmware template file named *F3xx_USB0_Standard_Requests.c* defines all standard requests. Some of these standard requests require the device to transmit information back to the host. One such standard request, *Get_Descriptor*, allows the host to gather all basic information about the newly-attached device. Other standard requests require the device to receive additional packets of information before the transaction terminates [OOO].

Device Detection after Successful Enumeration

Standard requests sent during enumeration by the host system are not controlled by user-level code. When a device connects with a host's USB port, the host system software will automatically retrieve descriptors and determine whether to enable communication with the device. Host-side application software wishing to interface with the device can then begin communicating with the device using standard API calls [OOO].

Application Communications

Once a device has successfully enumerated, the host can begin sending and receiving data in the form of reports. All data passed between an HID device and a host must be structured according to specifications found in the report descriptor. These reports can be transmitted across either the "Control" pipe (endpoint 0) or the "Interrupt" pipe [OOO].

Report Descriptors

All data transferred to and from an HID device must be structured in the form of reports. The report descriptor defines the report structure, which contains all the information a host needs to determine the data format and how the data should be processed [OOO].

Report Structure Overview

Although the report structure must follow a few constraints and guidelines, the HID specification purposefully allows for a high degree of customization. This potential for customization gives HID device designers freedom to create a wide variety of HID-class devices [OOO].

Usage Page and Usage Items

A report descriptor begins with a usage page item that describes the general function of all of the reports that follow. For instance, in a report descriptor describing reports for a USB keyboard or a USB mouse, such as the one found in the USB mouse example in this document, designers would use the “Generic Desktop” usage page. Reports contained in defined usage pages have defined usages, which provide more specific information about the report contents. For example, a keyboard would use the “Keyboard” usage for its “Generic Desktop” usage page [OOO].

Collections

“Collections” group similar data. Every report descriptor must have at least one top-level collection in which the data is contained, but the descriptor can define more than one top-level collection. A keyboard with an attached mouse would have two top-level collections, one describing mouse reports and one describing keyboard reports. Each collection must have a usage tag. For example, the “Keyboard” usage can tag a collection of USB keyboard related data. Also, collections can be nested.

The HID specification defines three types of collections, the Applications Collections, Logical Collections and the Physical Collections. The *Application Collections* group variables carry out a common purpose. All report items must be contained inside an application collection. The *Logical Collections* group variables of different types that form a composite data structure. Think of logical collections as a collection designator for “struct” data types, such as a struct that groups a buffer with index variables for that buffer. The *Physical Collections* group data describing a single data point. For instance, a physical collection of data could contain readings from a temperature sensor [OOO].

Data Types

Report descriptors also contain extensive information describing the characteristics of each data item. Logical Minimum and Logical Maximum items describe the boundary conditions the data contained in reports can reach. The Report Size item describes how many bits each data item uses, and Report Count describes how many data items are contained inside the report. A report of Size 8 and Count 2 would contain 16 bits, or 2 bytes of data.

Data values are further described by designating each data item as Input, Output, or Feature. Input items travel from device to host; Output items travel from host to device, and Feature items can travel in either direction. Data items can also be designated as Variable, meaning that the values can be read and written, or Constant, meaning that the values are read-only. Another often-used designation indicates whether the value is Absolute, meaning that the value contained in a report is measured from a fixed origin, or Relative, meaning that the value has no fixed reference and instead describes the change of value since the last report.

Systems using more than one defined report structure also need to give each report a unique Report ID tag. This Report ID precedes reports during transfer and signals to the receiver which report is being transmitted. Firmware and Software example. For a more detailed discussion about the items in a report descriptor, see the latest revision of the HID specification [OOO].

Two Transfer Types

Data traffic contained in HID reports can be transferred between device and host through one of two methods. Reports travel across the “Control Pipe” when control endpoint transfers are initiated by host calls to the Windows API functions, `HidD_SetOutputReport()`, `HidD_GetInputReport()`, `HidD_GetFeatureReport()`, and `HidD_SetFeatureReport()`. Reports travel across the “Interrupt Pipe” when data is made available for transfer across endpoints configured as Interrupt IN or Interrupt OUT. The next subsections examine Control Pipe and Interrupt Pipe data transfers [OOO].

Control Transfers from the Perspective of the Host

The HID specification defines six HID-specific requests. For a complete list of HID-specific requests, see the relevant section in the HID specification. `HidD_SetOutputReport()` and `HidD_GetInputReport()` allow host applications to send and receive IN and OUT reports. Parameters passed in with the call to `HidD_SetOutputReport()` include the handle to the HID device, the buffer containing the report, and the number of bytes to transmit. Similarly, calls to `HidD_GetInputReport()` require parameters for the handle to the HID device, a buffer where the incoming report will be stored, and the number of packets that the system expects to receive. `HidD_GetFeatureReport()` and `HidDSetFeatureReport()` receive and send Feature reports, which are bidirectional. The API calls first send a packet to the control endpoint of the device that contains reserved command byte corresponding to `HidD_SetOutputReport()` or `HidD_GetInputReport()`. In the case of the `HidD_SetOutputReport()` command, the system then transmits a second packet containing the report. In the case of the `HidD_GetInputReport()` command, the API transmits a second packet containing the Report ID of the report the application wishes to retrieve. The host then expects the device to ready that packet for transmission to the host, and the host makes an attempt to retrieve that packet [OOO].

Control Transfers from the Perspective of the Device

After the host initiates a control endpoint transfer, the device’s control endpoint handler parses the bytes of this setup packet to determine what request has been transmitted to the device. If a `HidD_SetOutputReport()` request has been transmitted to the device, the Report ID of the report to be retrieved will be included as part of the setup packet. The device then transmits that report back

to the host. If a `HidD_GetInputReport()` request has been transmitted, the firmware switches the handler into `EP_GetReport` mode. The host then transmits a report to the device. After the report has been transmitted, the microcontroller signals firmware of the availability of the report, and the report can be retrieved from the buffer [OOO].

Interrupt Transfers from the Perspective of the Host

During enumeration, host system software learns about the interface of the attached USB device. After reception of endpoint descriptors, the system polls any endpoint configured as interrupt IN or interrupt OUT at an interval defined in the endpoint descriptor. To retrieve IN Endpoint reports transmitted across the interrupt pipe by the device after a poll from the host, the application calls a Windows API function called `Readfile()`. This function requires parameters for the handle of the device, a buffer to store the information, the number of bytes requested, and a variable where the number of bytes successfully retrieved will be stored. To transmit an OUT Endpoint Report across the Interrupt Pipe, an application calls the Windows API routine named `Writefile()`, and passes into this function parameters including the device handle, a buffer containing the report to be transmitted, the number of bytes to be transmitted, and a variable where the number of bytes successfully transmitted will be stored [OOO].

Interrupt Transfers from the Perspective of the Device

Until a device has data to send across the IN endpoint, it should simply NAK the host's polled requests for data by not signaling that data is ready to be received. Once data has been collected into a report structure and placed onto the IN endpoint's buffer, firmware signals that data is ready to transmit. When the host sends a packet across the OUT endpoint, the microcontroller signals the firmware, and the OUT Endpoint handler retrieves the bytes from the OUT endpoint buffer [OOO].

Two Transfer Types

Data traffic contained in HID reports can be transferred between device and host through one of two methods. Reports travel across the "Control Pipe" when control endpoint transfers are initiated by host calls to the Windows API functions, `HidD_SetOutputReport()`, `HidD_GetInputReport()`, `HidD_GetFeatureReport()`, and `HidD_SetFeatureReport()`. Reports travel across the "Interrupt Pipe" when data is made available for transfer across endpoints configured as Interrupt IN or Interrupt OUT. The next subsections examine Control Pipe and Interrupt Pipe data transfers [OOO].

Control Transfers from the Perspective of the Host

The HID specification defines six HID-specific requests. `HidD_SetOutputReport()` and `HidD_GetInputReport()` allow host applications to send and receive IN and OUT reports. Parameters passed in with the call to `HidD_SetOutputReport()` include the handle to the HID device, the buffer containing the report, and the number of bytes to transmit.

Similarly, calls to `HidD_GetInputReport()` require parameters for the handle to the HID device, a buffer where the incoming report will be stored, and the number of packets that the system expects to receive. `HidD_GetFeatureReport()` and `HidDSetFeatureReport()` receive and send Feature

reports, which are bidirectional. The API calls first send a packet to the control endpoint of the device that contains reserved command byte corresponding to HidD_SetOutputReport() or HidD_GetInputReport().

In the case of the HidD_SetOutputReport() command, the system then transmits a second packet containing the report. In the case of the HidD_GetInputReport() command, the API transmits a second packet containing the Report ID of the report the application wishes to retrieve. The host then expects the device to ready that packet for transmission to the host, and the host makes an attempt to retrieve that packet [OOO].

Control Transfers from the Perspective of the Device

When the host initiates a control endpoint transfer, the device's control endpoint handler analyzes the bytes of this setup packet to determine what request has been transmitted to the device. If a HidD_SetOutputReport() request has been transmitted to the device, the Report ID of the report to be retrieved will be included as part of the setup packet. The device then transmits that report back to the host.

If a HidD_GetInputReport() request has been transmitted, the firmware switches the handler into EP_GetReport mode. The host then transmits a report to the device. After the report has been transmitted, the microcontroller signals firmware of the availability of the report, and the report can be retrieved from the buffer [OOO].

Interrupt Transfers from the Perspective of the Host

Throughout enumeration, host system software learns about the interface of the attached USB device. After reception of endpoint descriptors, the system polls any endpoint configured as interrupt IN or interrupt OUT at an interval defined in the endpoint descriptor. To retrieve IN Endpoint reports transmitted across the interrupt pipe by the device after a poll from the host, the application calls a Windows API function called Readfile(). This function requires parameters for the handle of the device, a buffer to store the information, the number of bytes requested, and a variable where the number of bytes successfully retrieved will be stored. To transmit an OUT Endpoint Report across the Interrupt Pipe, an application calls the Windows API routine named Writefile(), and passes into this function parameters including the device handle, a buffer containing the report to be transmitted, the number of bytes to be transmitted, and a variable where the number of bytes successfully transmitted will be stored [OOO].

Conclusion

HID has built in software and therefore there is no programming to be done by the team. HID is a technology that the team decided to implement to make the NAT device easier and more user friendly. It allows us to upload data from the NAT device and connect the device to the host computer in a simpler way than compared to other options.

3.3.6 Power Components

3.3.6.1 DC/DC Converter

This design makes use of a Texas Instruments TPS6220 step-down DC-DC converter to regulate the voltage output of our battery supply and provided power to the microcontroller and all other active modules. This device accepts DC voltages from 2.5 – 6 V, which fits our 3.7 V Lithium Ion battery perfectly, and outputs voltages configurable from 0.7 V up to the input voltage at 300 mA. The TPS6220 will be set to output 3.1 V because this supply voltage fits within the range of all the active components being used.

Our device was chosen over other similar devices such as the Torex Semiconductor XC9260 or the Enpirion EP5348UI because of factors like: quiescent current draw, efficiency and features such as soft start. The TPS6220 boasts a quiescent current of just 1.5 μA , 95% efficiency, and 300mA of output current. However, both the XC9260 and the EP5349UI only advertise 90% efficiency, the XC9260 draws 2.5 μA of quiescent current and the EP5348UI does not have a soft start feature. Lastly, the TPS6220 features a low power mode when being used under light loads which increase the overall power efficiency over extended periods of use.

TPS6220

This device is designed for operation in portable devices that run on a single cell lithium ion battery or it can operate off of a 3.3 V – 5 V rail. Some of the most appealing features of the TPS6220 is the high efficiency operation under normal loads, its ability to automatically switch into a low power mode, Low Drop Out (LDO) operation, an adjustable output voltage, high output current and the minimal size and part requirements [A].

The TPS6220 boasts up to 95% efficiency under normal loads around 300 mA and a quiescent current of only 30 μA . However, this device is unique in that it attempts to maintain high efficiency over a wide range of load currents (1 mA – 300 mA) by employing a low power mode. The device can enter low power operation under two conditions: if the peak switching current falls below an internally set value determined by the input voltage or if the device detects discontinuous conduction. When the device enters low power mode the switching frequency is lowered from its nominal value of approx. 1 MHz to its minimum value of approx. 650 kHz and the quiescent current falls to just 15 μA .

This low power operation feature is important to our project because our device is designed to spend a large portion of its active life cycle in standby waiting for a ping from the main servers so over the long term this feature will save a substantial amount of power and add a lot of time to our devices battery life.

An important feature for any device that operates off of a lower voltage than the input is Low Drop Out operation. In normal voltage regulators, the difference between the input and output voltages must be higher than a minimum value called the dropout voltage, but this is made difficult when the input voltage is higher than the output voltage and when the input and output voltages are very close to one another. For this reason, LDO voltage regulators are designed to operate with a very small dropout voltage (approx. 300 mV) [B] and where the input voltage is higher than the output.

Obviously, this kind of regulator is necessary for our project because the input lithium ion battery produces a maximum voltage of 3.7 V and we require an output of 3.1 V which gives a maximum dropout voltage of only 600 mV. Also, an adjustable output voltage is important for our project because 3.1 V is not a commonly offered voltage in voltage regulators.

Lastly, The TPS6220 only requires a minimum of four external components, for the nonadjustable version, and seven components for the adjustable version and the SOT-23-5 packaging of only 2.90 mm x 1.30 mm keeps the power supply circuit size to a minimum. Also, the fast response times of this device allow for the use of very small, and cheap, ceramic capacitors. The last crucial factor that will influence our decision is the price point of this device which is compared with other similar devices in Figure 50. The TPS6220 seems perfect for our application but we must ask if there are better devices or comparable devices offered at lower prices.

LTC1701

The first device up for comparison is the LTC1701 from Linear Technologies which on the surface seems comparable to the TPS6220 but upon further inspection it can be seen that this device is much better suited for operations other than ours [C].

The LTC1701 also offers an acceptable efficiency, LDO operation and an adjustable output voltage as well as similar parts requirements and small package size. However, the LTC1701 does not offer a comparable range of load currents for which it maintains efficiency (10 mA – 300 mA) because of its lack of a low power operating mode and even if it did the power consumption on this device would be significantly higher, even under nominal loads, because of the high quiescent current of 135 μ A. What the LTC1701 does have going for it is a high output current at 500 mA while still maintaining good efficiency, load and line regulation.

The LTC1701 seem much more suitable for applications where high output power is required and battery life is not a stringent concern. The last major concern with the LTC1701 is that it has the highest price point of any of the devices considered here at \$2.12 a unit (when purchasing at least 100 units).

FAN5307

The next device is the FAN5307 from Fairchild Semiconductor and in many respects, it is very similar to the TPS6220, in fact it hits most of the same important points that the TPS6220 does: high efficiency over wide load currents using a power saving operation mode, acceptable output current, low quiescent currents, adjustable output voltage and small package size [D]. However, there are two points where the FAN5307 does not match up to the TPS6220.

The FAN5307 does not offer an LDO operating mode and it is a fair bit pricier than the regulator from Texas Instruments at \$1.23 per unit (when purchasing at least 100 units). The omission of LDO makes this much harder to implement, not impossible but with the option available why would we make our design process harder and the price difference makes the FAN5307 almost unusable.

	LDO	Quiescent Current	Low Power	Output Current	Cost ¹
TPS6220	Yes	15 μ A – 30 μ A	Yes	\leq 300 mA	\$0.96
LTC1701	Yes	135 μ A	No	\leq 500 mA	\$2.12
FAN5307	No	15 μ A – 30 μ A	Yes	\leq 300 mA	\$1.23

¹ Cost per 100 units, pricing found on supplier websites

Figure 50 - Voltage Regulator Comparison Table

3.3.6.2 Voltage Reference

The next important power-providing component is the ISL2108, a voltage reference generator that provides the 3 V reference voltage that our microcontroller requires to operate its internal ADC that measures the battery voltage. This device was chosen because of its low current draw and output voltage noise. The only external components required for this device are two bypass capacitors C1 and C2 (Component identifiers with respect to Figure 69) to filter noise from the input and the output.

ISL2108

The ISL2180 is characterized by its low operating current, 310 nA typical and only 1.5 μ A at a maximum, under normal applications, and this particular voltage reference also guarantees an initial accuracy of 0.2%. Another important parameter of voltage references is the temperature coefficient and the ISL21080 has a temperature coefficient of only 50 ppm/ $^{\circ}$ C, this means that this voltage reference's output voltage will only vary by 5×10^{-6} % per degree Celsius [F].

Voltage references come in many forms as discussed in Section 5.3 and the ISL21080 uses CMOS transistors to form a floating gate that holds the output voltage constant as well as to amplify the output voltage to the desired value. A floating gate voltage reference makes use of the gate charge of a MOSFET with is precisely set by the manufactures to set the voltage output. This kind of voltage reference offers a consistent voltage output with low temperate drift, high initial accuracy and low operating current and the reference voltage is not bounded by bandgap voltages or Zener diodes.

The last appealing features of the ISL21080 are the small package size, low part requirements and the price point. Much like the DC/DC converter discussed above this voltage reference uses a small SOT-23 package (2.92 mm x 1.30 mm) and only requires two decoupling capacitors on the input and output. Lastly, this device can be used for just \$1.16 per unit when buying in sets of 100 units.

LT6656

In many ways, the LT6656 is a comparable voltage comparator to the ISL21080 and if fact may be preferable in applications where higher accuracy is a larger concern. As an aside, the LT6656 is a bandgap voltage reference, discussed in Section 5.3. However, for our project the 0.05 ppm/ $^{\circ}$ C temperature coefficient is not as important as the, comparably, very large supply current of 850nA and the price point of \$6.76 per unit when buying in sets of 100 units [G].

In short, this device has the same initial accuracy as the ISL21080, the same packaging and the same external parts requirement and the only parameter that is worth consideration is the low temperature coefficient. However, in our opinion the slight advantage this device has in accuracy is not worth the huge losses in power consumption and the financial burden.

REF3030

The REF3030 is also a bandgap voltage reference that is comparable to the ISL21080 under almost every parameter except for possibly the most important one, at least for our application. The REF3030 has a similar initial accuracy and temperature coefficient to the voltage reference from Intersil, but has a much higher operating current than both of the above voltage references at 50 μA [H].

However, the REF3030 does have something going for it. In applications where battery life is not a major concern the REF3030 is a bit cheaper than the ISL21080 at \$0.98 per unit when buying in sets of 100 units. The advantages and disadvantages of all of the above voltage references are compared in Figure 51 below.

	Initial Accuracy	Temperature Coefficient	Supply Current	Cost ¹
ISL2108	0.2 %	50 ppm/ $^{\circ}\text{C}$	310 nA	\$ 1.16
LT6656	0.2 %	0.05 ppm/ $^{\circ}\text{C}$	850 nA	\$ 6.76
REF3030	0.2 %	50 ppm/ $^{\circ}\text{C}$	50 μA	\$ 0.98

¹ Cost per 100 units, pricing found on supplier websites

Figure 51 - Voltage Regulator Comparison Table

Update: There was an addition of a TLV700 1.8 V Linear Regulator from Texas Instruments to power the GPS. This is discussed in section 6.1 the schematic.

3.3.7 RPMA Modules

As an Engineer, you will find that keeping up with technological advancements is a task in itself. To stay relevant and design relevant products, we as Engineers must use the latest cutting edge technologies. This is one of the reasons why we chose to integrate RPMA technology into our design. Since RPMA is a closed standard and only provided by Ingenu, our selection is limited to modules they design.

The team decided on the Nano S100 module designed by Ingenu and produced by U-blox. This module will utilize the license free radio band of 2.4GHz. This band is license free because it is significantly populated by microwaves creating much noise on the band. The team will use this module in conjunction with the LTE module to send the messages our device needs to send to the various places it needs to send data to.

Update: In the final design, RPMA did not make the cut for our communication module. Implementing RPMA into our design turned out to be inconvenient and expensive and therefore did not meet our budget. In addition to this, Ingenu as an organization is going through some

changes. As a result, acquiring the RPMA module for this project would be a bad idea due to the uncertainty of its future.

3.3.8 Cellular Modules

About half of the cellular technologies that we were initially interested in are not as popular or fully supported in the United States as they are in other countries like Europe and Asia. These newer technologies like LoRa, Sigfox, and NB-IoT were either developed or began their deployment in Europe. This is why modules using these technologies are either not available in the U.S. or not fully supported. The LTE Cat-1 network has already been deployed and is fully functional, which is why the XBee LTE Cat-1 module from Digi International was selected. In addition to availability, Digi assures that changing the cellular module from LTE Cat-1 to Cat-M or NB-IoT will have little to no impact on our design which leaves room for future improvements and exploration of technologies that we were initially interested in.

Update: LTE-M1 and NB-IoT are great considerations for achieving Low power IoT wireless connectivity. These technologies however, are new and not as popular compared to other IoT communication technologies. This makes getting a device that we can implement in our design very difficult. There would also be limited support for any issues we may encounter during the development of our project.

To overcome these issues with our communication module, we decided to use the LTE CAT-1 communication standard to handle the wireless communication interface between our hardware and the necessary software. With LTE CAT-1 operating on an already deployed network, we are guaranteed more reliable coverage for our wireless module. The fact that CAT-1 has been established for much longer than other LPWA technologies, we found that support for this and the modules used to implement it is readily available online or through the suppliers.

3.4 Possible Architectures and Related Diagrams

Hardware Components Architecture Comparison

The hardware architecture could take many forms, and a few are discussed below. Discussion into possible architectures early in discussions is imperative as many of the design and development is based off of what is discussed and decided for the architecture. Two possible options are discussed below, a modular design as well as a more integrated, fully PCB design.

Option 1:

Under initial development, the hardware architecture will consist of a separate printed circuit board for each component module. Development kits for the microcontroller, cellular radio, RPMA radio, GPS, and IMU will be interconnected using jumper wires on communication and power headers. The exact board-to-board communication protocol will be some combination of UART, SPI, and I²C as dictated by part selection. This separate modular design will simplify unit testing of the software components, as the associated hardware for each can be tested in isolation. Later hardware development may retain this modularity, but at minimum each development board will

be replaced with a much smaller design retaining only the essential circuitry. This option is diagrammed in Figure 52 below.

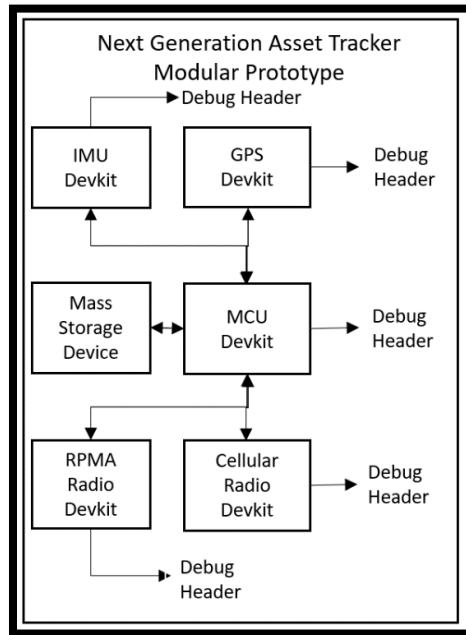


Figure 52 - Hardware Prototype Architecture: Modular Design

This hardware architecture option would allow us to meet all of our needs, but would come with many issues. To begin with, the design would be very loose and would require many unnecessary components. Such as, every development board would require its own circuit so that it can run on its own and be connected properly. There would also be difficulty if this was made the final design as the device would be harder for the user to work with and not as easy to integrate into their systems.

Option 2:

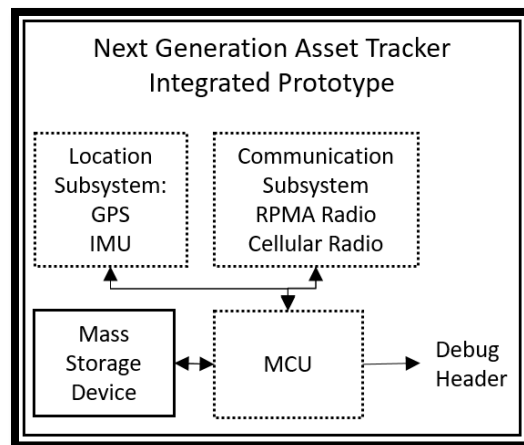


Figure 53 - Hardware Prototype Architecture: Integrated Design

Alternatively, to more aggressively meet the desired size constraints, the main chip for each module can be mounted onto the same PCB, with interconnection provided by direct traces. This design is diagrammed in Figure 53 above.

Figure 54 below is our first draft of the block diagram for our project, it gives us an idea of how our device might function and how it might be laid out a single PCB board.

This is more of the design that the engineers had in mind when the engineers discussed the product initially. Whereas, the design in Figure 52 would be sufficient for functionality, it does not allow the engineers to meet the size requirements. The size requirement, dictated in both Section 2.3 Requirements Specification and Section 2.5 Quality of House Analysis, is essential in our design more than just wanting to produce a small design. The size requirement is there because the user is going to attach our NAT device to their own products. They would not integrate it fully into say their electronics, where the device being more modular may be fine since another engineer would have to go in and integrate it into their design. Instead, the NAT device is meant to be fully independent of other devices and products and instead be placed on top of or inside another device. Meaning that if the user had a big shipping container they wanted to track, they would only place the device on top of or inside the container. Having to deal with wire connected external components instead of one integrated small device would not be beneficial to the user. And since this design is meant to be sold to consumers, making a device that would fit their needs more appropriately would be better for the marketing and selling of the NAT device.

Therefore, between the more modular design and the fully integrated design, the team will go with a more integrated design, but will use the benefits of the modular design during testing while the electrical engineers will integrate the components together to be fully integrated on the PCB.

Hardware Possible Block Diagram

The top portion of the block diagram in Figure 54 below is the conversion of 3.7V from the Lithium Ion battery into 3.1V and 3.0V via a DC/DC converter and a voltage reference generator. This is necessary because the NAT device's components need more than just one voltage for all of them. Some components need 3.1V and some need 3.0V. These parts of the block diagram allow us to achieve these different voltage levels with a single voltage source.

Within the main part of the block diagram in Figure 54 below it is seen that all of the functionality stems from the PIC. The PIC microcontroller will store the firmware that each component needs, as well as providing all input and output from every device, except the ones that take input and output external to the device as well. Stemming of the PIC in the center at the top the diagram shows the Micro SD, which will be in the form of a Micro SD socket and chip. The Micro SD will be connected via serial data, possibly over the UART. The right of this is the GPS, which will be connected over I2C or UART. Both are available to the component through the PIC through both hardcoded and variable connections. The RPMA module will be connected through an SPI connection, while the Cellular module was yet to be determined in regards to the connection type. The IMU will have to be connected over I2C. There is also a ESD or Electrostatic Discharge protection unit, to protect the device when the micro USB is connected.

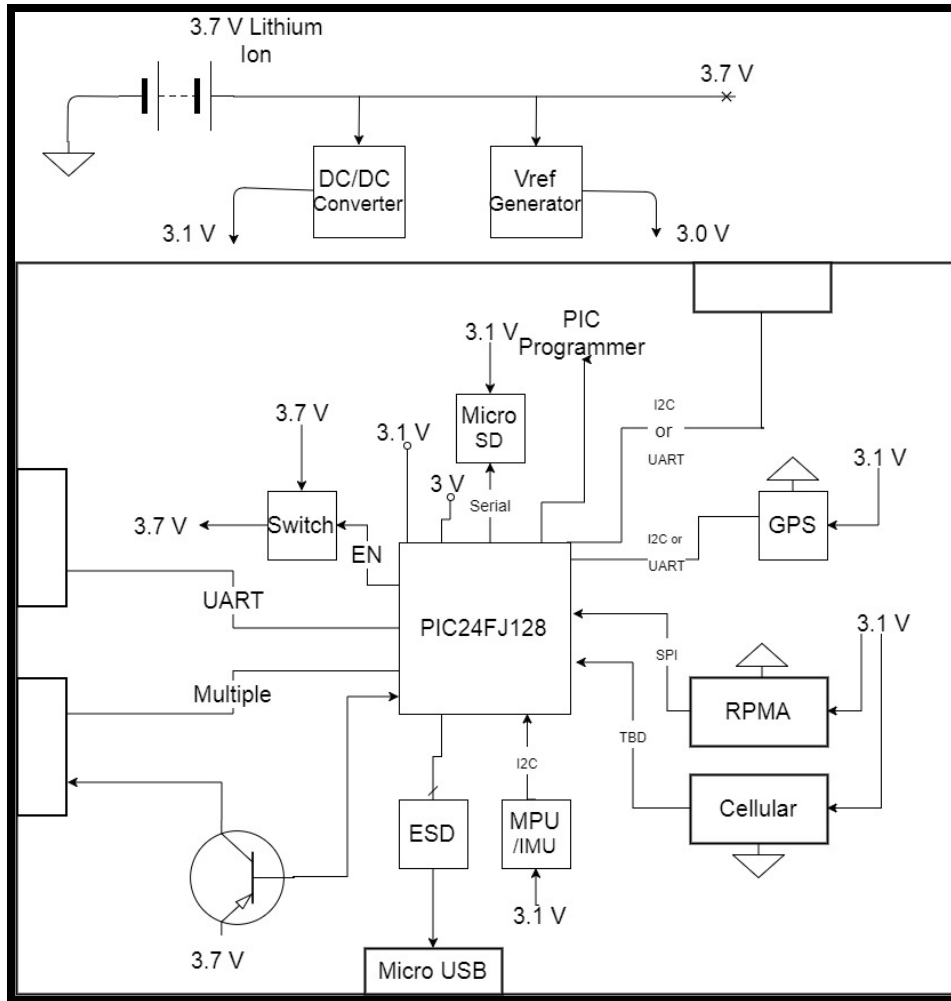


Figure 54 - Hardware Prototype Possible Block Diagram

The various rectangles that are located at the edges of the main square indication connections to external devices. There will be a port for a UART connection, a space that can be programmable to multiple connection, as well as both and I2C/UART open connection and a Micro USB connection. The device will allow for external devices to be added, as well as debugging to be done through the UART ports.

Software Possible Architectures

The architecture discussion now moves into the software, an equally important, yet unseen, design component. Discussion of the architecture for the software is important because it is necessary to get an overall idea of how the software will look, before any line of code is ever written. The computer engineers will decide on an architecture for every piece of software that needs to be written, and the discussion of such is shown below.

Firmware Architecture

The MCU software will be a structured program, with execution parameters defined in a configuration file on the mass storage device. In operation, a loop will execute, reading the values from the GPS and IMU, logging the sensor data to the mass storage device and transmitting the data over the RPMA and Cellular links, logging success or failure. Mature peripheral drivers provided by the MCU vendor will be used wherever possible to reduce the likelihood of faults in the bulk of the code. The operational loop will be controlled by the power-saving profile defined in the configuration file, either waking up based on a timing interval, or via motion trigger.

Configuration GUI Architecture

The configuration GUI will run on a standard Windows workstation using the .NET platform. Execution will be event-based, reacting to the detection of an attached NAT device over USB and user initiation of all other features through on-click handlers for an associated form element for each of the software's functionalities, including self-test procedures, configuration profile downloads, data log dumps, and debug log dumps. While attached, diagnostic data will also stream over the USB link, allowing the operator to determine whether any of the hardware modules or the microcontroller firmware are in a failure state.

Location Database and Customer GUI

To support the remote tracking of assets, a secure internet-facing database will store the recorded location coordinates along with a timestamp and identifier for each NAT device. For the purposes of our prototype, the database will only need to support a small number of devices for a single customer with a data structure unlikely to need to be extended. A database implementing a Structured Query Language (SQL) schema is appropriate as long as these constraints remain in place. However, our design is meant to eventually support an Internet of Things ecosystem, with many customers each with any number of NAT devices to track, and the ability to integrate additional sensors as the customer sees fit. To meet these scalability needs, a semi-structured database system, such as one based on XQuery, may be used instead. [OO]

A customer-accessible web application will allow the customer to see the latest and historical data from their registered NAT devices. By targeting the web browser, virtually all customer mobile devices as well as personal computer will be supported through a single programming effort, which would otherwise require at minimum separate versions for iOS, Android, and PC. The web application will be based on a Model-View-Controller framework (MVC). In this paradigm, a "model" is a software module that handles all business logic, such as the database querying code. A "view" consists of the user interface, which in a web application includes HTML for content, CSS for presentation, and JavaScript for behavior. Finally, the "controller" bridges the models and views, allowing them to be developed and executed in isolation from each other, providing the desired information while hiding the back-end business logic. Logged-in users visiting the customer website will have their browsers pass their request to the controller, which obtains the data from the model, then presents it to the user through the appropriate view. [PP]

3.5 Parts Selection Summary

In the above sections, the relevant technologies and the strategic components and parts selection were discussed at length. Where as anyone who wishes to see an in depth reasoning for why the devices the engineers chose were selected, this section will summarize relatively quickly and very broadly why the components were chosen. Overall, this section will review the technologies and the components corresponding to such technologies that were chosen for our design, images and discussions of the materials and components that the engineers have received and already tested (and possibly did not receive in time to adequately test before the paper) and a quick review summary as to why the final selection was made, as the full discussion is described in each individual section in the paragraphs above.

Below are figures that depict all of the components relevant to the NAT device. Figure 58 is a picture of all of the components that are currently available in the Winter Park lab relevant to the production and testing of the NAT device. The discussions over these components are shown below the images. Figures 56 and 57 is the development board that was available to the team. It was a fully functional development board, allowing the engineers to work with some of the components more readily than if they were in their singular components form. Figure 55 is an image depicting the design setup for a few of the development boards to be programmed and tested on one of the engineer's personal computers.

Below in Figure 55 on the left is the complete testing setup of the development board that our team is using to test the components that we will be using in our design. Our sponsor, Young Engineering Services LLC, had a board with many of the components that we ultimately chose to use on a development board. That development board is shown in a closer image below, in Figure 56 and Figure 57.

The testing setup in Figure 55 shows some of the various testing components are labeled. Component A is the ICD3, an in-circuit debugger. What this does is it allows the computer engineer to debug a software application on our own hardware in real time, debug hardware breakpoints, debug software breakpoints, set breakpoints based on internal events, monitor internal file registers, emulate full speed, and program the hardware [GG]. It is connected to the development board (component B) via one of the connectors to the device, shown below in Figures 56 and 57, and the host PC (component D) via another USB cable.

Component B is the development board. The development board is shown in more detail in Figures 56 and 57 below. This development board contains the MCU, IMU, SD mount, and other various components that we will utilize in our design. Therefore, it was utilized to test these components to ensure that they were applicable for our design. The development board (component B) is connected to an external development board for the GPS via the blue cable (component C), as well as to the host PC (component D) to view the serial data output that it has and the ICD3 (component A) to allow the development board to test the code that was designed and programmed to the development board.

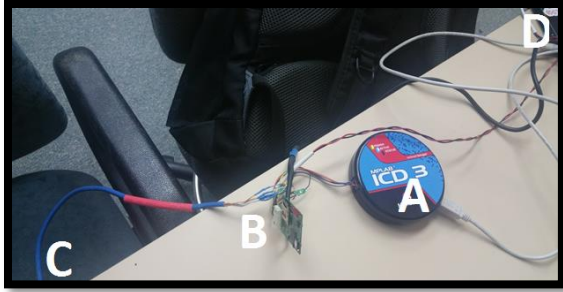


Figure 55 – (Left) Testing Setup for our Development Board, (Right) Ingenu RPMA Module
*Permission Requested [P8]

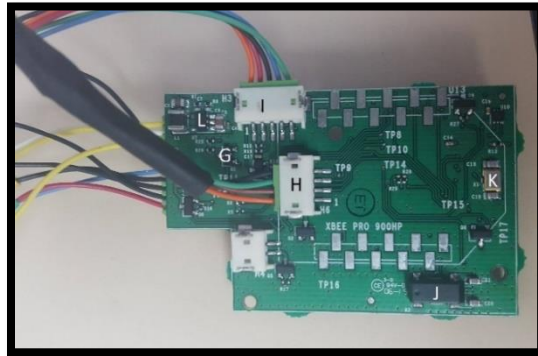
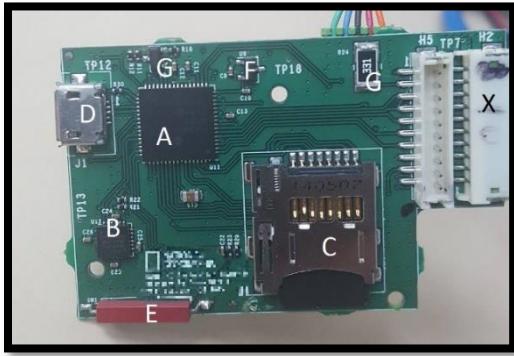


Figure 56 – NAT Dev Board: Front View Figure 57 - Nat Dev Board: Back View

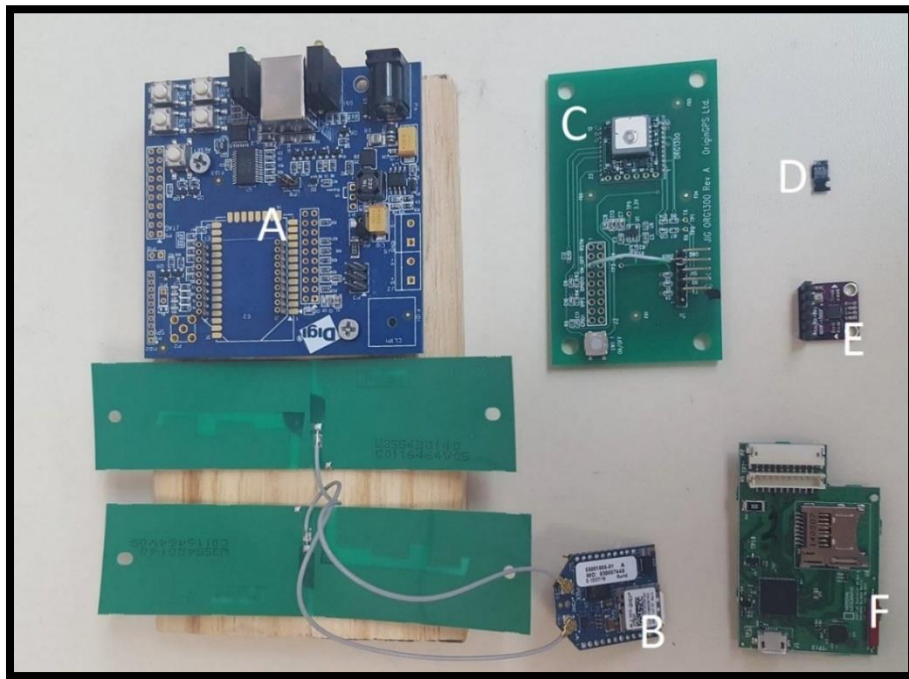


Figure 58 - All Components

Component C is the GPS (only the connector cable is shown in this image but the actual device can be seen in Appendix Testing Image 5 - GPS Module, in Appendix Section 9.4 Additional Testing Images). This GPS resides in the window to allow the development board to read the GPS location that it reads, for coding, debugging, and testing purposes. The GPS development board is only connected to the development board via the blue cable shown above, which the electrical team designed and developed for testing purposes with the assistance of our sponsor.

Component D is the host computer, or better the connections that go to it. The connections from the host PC go to the ICD3 to program the development board (the gray wire), and to the development board itself to receive the serial output it sends (the black wire), shown clearer in Figures 56 and 57 above.

Figure 56 above shows the front view of the development board that the sponsor request that the team use to test the components that we decided to use in our design. These components are as follows.

Component A is the microcontroller, the PIC24FJ128GC006 was chosen because it meets all of the requirements of our design, it comes with an In-Circuit Emulator device, of which our sponsor already owned so it allowed us significant cost savings. While some of the other possible microcontroller options provided better specifications overall, it was ultimately determined that these features were not desirable enough to outweigh the familiarity our sponsor had with the PIC24 architecture, flattening the learning curve significantly. With this knowledge, we could cut back some time spent learning, as well as having a dependable contact should we need help debugging any issues that we had with the device. Therefore, for our main processing power, we chose the PIC24FJ128GC006.

Component A is connected to every other major component as shown in Figure 56. Therefore, to test some of these features we would have needed some connection to a processor, and therefore the development board provided us significant benefits in terms of designing, writing, and testing the software and hardware components of our project.

Component B is the IMU, MPU 9250. It provides many benefits to our design including allowing us to implement motion triggers as well as in the future design an INS algorithm to get tracking in places where GPS cannot get signal. In addition to these benefits, ultimately our sponsor determined that having experience with this technology was a significant benefit to choosing this module for our design. However, in the future it will be interesting to note that some of the radio modules that were researched included an integrated IMU, which could be utilized instead of having a separate IMU stuffed onto the PCB. This would reduce size and energy consumption, but in the scope of this current project the IMU will be embedded separate to the radio module. The MPU-9250 was specifically chosen because it is a 9-axis MotionTracking device because it combines a 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer and a Digital Motion Processor™ (DMP) all in a small 3x3x1mm package. The contents and size of this IMU make it an ideal choice for this project and is the reason the team chose to implement it into the NAT device. More details into choosing this module are located in the section 3.3.4.3 IMU Comparison and Selection.

Component C is the SD mount. It is a major component (Position Card Connector Secure Digital - microSD™ Surface Mount, Right Angle Gold), but we decided that the one on the development board was fine enough for the design we are using and no additional research really needed to be done into finding the best one for our needs. It is in the development board, so testing with it is easy and the sponsor has experience with the component and that is invaluable for our project.

Component D is the USB port utilized for our USB communication between devices. The part (USB - micro AB USB 2.0 Receptacle Connector 5 Position Surface Mount, Right Angle) was again utilized because the sponsor had experience with this component and we assumed components do not differ much between them so additional research was not necessary.

Component E is the reed switch, specifically the Molded Body Reed Switch SPST-NO 15. The reed switch will be utilized to keep our unit consuming as little power as possible when in residing in the warehouse and when in transit. This is to conserve as much power as possible, so as the user has as much power in their device as we can allow them. We decided on this particular reed switch as it was already on the development board, so we could test with it, as well as the sponsor had experience with it and we determined to go with experience over additional research.

Component F is the TPS62203 voltage regulator from Texas Instruments. This device was chosen because of its low supply/quiescent current, high efficiency over a wide range of loads and its low cost/component requirements.

Multiple components of Figure 56 and Figure 57 are the same, labeled as the G components. These are a FDN358P Single P-Channel MOSFET, utilized as an ON Semiconductor. They will allow us to turn power on and off to certain components of our design. This is necessary to conserve as much power as possible during the operation of our product. We chose this specific brand of PMOS because they are readily available in large quantities for a low cost, our project uses 6 of them.

The final component shown here is the X component, which is a bulkhead connector allowing for us to connect the GPS development board and the external power source (currently a wall-wart connection) into the same pins. You can see these wires in Figure 55, component C connecting from the development board to the GPS gets connected to this component.

Figure 57 above is the same development board, this time just the back view of the development board. These are where some more of the components we will use reside.

Component G, again, is the same as the component G from the front board. It is a ON Semiconductor that we will also utilize in our design to turn on and off certain components to preserve power in our device.

Component H is the UART Interface connector. It is where the serial data will come out of the device. On the other end of this connection and cable is a USB connection to the host computer where PuTTY, a free SSH and Telnet client will be utilized to view the serial data coming out of the development board.

Component I is the connection to the PIC Programmer, or the ICD3 (component A from Figure 55). This connection will ultimately allow the computer engineers to write and debug code to the development board, and a similar connection will be made in our design for the NAT device.

Component J is a 32.768kHz Oscillator. Component K is a 32MHz Oscillator.

Finally, Figure 58 shows all of the components that have been ordered for the NAT device. This includes some development boards and some individual components ready to be placed on the board.

Component A is the development board for the Digi XBee Cellular LTE CAT-1 module that we have decided to use for one of wireless communication components. The cellular device is less cutting edge than the other wireless communication module that we plan to implement, the RPMA module, shown in Figure 55 on the right. Although the RPMA module is more cutting edge than the XBee module, its maturity is questionable. Coverage from the RPMA network is not as great as the LTE module. Ingenu is still deploying the network while Verizon Wireless has been operating for many years and has a fully functional LTE network. This reliability is why we chose to implement the XBee LTE module in our design to compliment the RPMA module.

Component B is the actual XBee Cellular LTE CAT-1 module. This can be taken off of the development board and placed directly into our device once the device has been printed. It comes with an SIM card, which gets its own phone number and IP address.

Component C is the GPS development board. This board was used to test the GPS module both separately, and with our NAT development board. This specific GPS was chosen because it met all of the specifications that were required for our device, and the sponsor already had a development board for the device in the lab, helping with the budgeting of the project. In addition to this, the sponsor had experience working with this device, which is invaluable when the team has to design code and schematics including this module. A more specific breakdown of the reasoning behind choosing the GPS is found in Section 3.3.3 GPS Module.

Component D is a Texas Instruments Switching Voltage Regulators Adj 300mA Hi-Eff Step-Down Converter, which is necessary for our design. The TPS6220 only requires a minimum of four external components, for the nonadjustable version, and seven components for the adjustable version and the SOT-23-5 packaging of only 2.90 mm x 1.30 mm keeps the power supply circuit size to a minimum. Also, the fast response times of this device allow for the use of very small, and cheap, ceramic capacitors. The last crucial factor that will influence our decision is the price point of this device which is compared with other similar devices in Figure 50. The TPS6220 seems perfect for our application, and it was decided to be used for the design of the project.

Component E is the MPU 9250. The discussion of this unit has already been discussed above.

Component F is the development board that the team used to test a few of the components. These components have all been proven to work, and they have been detailed in an above section.

In Figure 55 on the right, there is the RPMA module that did not arrive in time to be pictured with the rest of our components. The RPMA module was chosen because RPMA is a new growing technology and we wanted to be on the cutting edge of this technology, with one of the first devices implementing this. Ingenu is essentially the only option for this technology with a module that is available for purchase.

Update: RPMA module never arrived, and was removed from the design. Reference the updated section of 3.3.7 RPMA for reasoning. We added a MCP73831 USB LiIon Charging circuit from Microchip and 2 indicator LEDs for battery charging

3.6 Bill of Materials

The Bill of Materials, or BOM, are the materials that are necessary to the NAT device. These are shown in the schematic in 6.1.2, the Bill of Materials include the description of the device, the manufacturer and their part number for the component, DigiKey P/N. Many of these components have been tested, and the team has been designing with these components.

QTY	Description	Manufacturer	Manufacturer P/N	DigiKey P/N
1	H_6-POS_1.5MM_RA	TE Connectivity AMP Connectors	1761681-1	A122759-ND
1	10-POS_RA	TE Connectivity AMP Connectors	1-111626-7	ASA10L-ND
2	NPPN10	TE Connectivity AMP Connectors	1-1470109-0	1-1470109-0-ND
1	Micro USB Port	Amphenol FCI	10104111-0001LF	609-4053-1-ND
1	Reed Switch	Coto Technology	CT10-1540-G1	306-1124-2-ND
1	Linear Voltage Regulator	Texas Instruments	LP5951MF-1.8/ NOPB	LP5951MF-1.8/ NOPBCT-ND
1	Microprocessor	Microchip Technology	PIC24FJ128GC010 T-I/PT-ND	PIC24FJ128GC010T-I/ PT-ND
1	ESD Suppressor	Vishay	VBUS054B-HSF	751-1453-1-ND
1	RPMA Radio Module	U-Blox	N/A	N/A
1	Switching Voltage Regulator	Texas Instruments	TPS62200DBVT	296-12719-1-ND
1	Series Voltage Reference	Intersil	ISL21080CIH330Z- TK	ISL21080CIH330Z- TKCT-ND
1	Inertial Measurement Unit	InvenSense	MPU-9250	1428-1019-1-ND
1	GPS Module	OriginGPS	ORG1411-PM01	N/A
7	On Semiconductor	Fairchild Semiconductor	FDN358P	FDN358PTR-ND
	SD Card Mount	Molex Inc	492250821	WM3295CT-ND
1	Oscillator - 32MHz	TXC CORPORATION	7M-32.000 MBBK-T	887-1925-1-ND
1	Oscillator - 32.768kHz	AVX Corp/Kyocera Corp	ST3215SB327 68E0HPWAA	478-5429-1-ND

Figure 59 - Bill of Material (BOM) Table

The Bill of Materials is a list of all of the components and parts necessary to manufacture an end product ready for release to the client. This Bill of Materials is meant to be a full inventory of sorts for the material the schematic and ultimately the PCB need to go into full design and manufacturing. The items that need to be included in such a document are as follows. The quantity is necessary as many designs require many components, such as in the NAT BOM above in Figure 59 the schematic required 2-10 pin connectors of a specific type. Then, the description. The description is written so that someone who did not create the BOM can easily find a component

they are looking for without having to know the Manufacturer P/N (Part Number) or DigiKey P/N. The manufacturer is necessary as many companies make similar products. Simply designating that the team needs, for instance, 3-10 Pin connectors is not enough information for someone reading the document in order to form a purchase order enough information. This, as well as the manufacturer's P/N allow anyone reading the document to find the specific part the person who designed the circuit wanted. The DigiKey P/N is because so many companies in industry, including the company of our sponsor, utilize DigiKey's vast variety of components to order as many, if not all, of the components they need at once. Even looking at the NAT device BOM above in Figure 59 a user can see that almost all of the components we need are available on DigiKey. Making a purchase order of all of our necessary components together would save us money on shipping, as well as allow us to purchase everything we possibly can at once, an incredible benefit to the engineer. The final Bill of Materials that the team will utilize will be when the final schematic is made, as it will be created using the OrCAD software. The OrCAD software has functionality built into it to allow, from a schematic such as the one in Section 6.1.2, a BOM to be made of the materials that the designer has used in their design.

3.7 Printed Circuit Board Technology

Almost all electronic products today use some kind of Printed Circuit Board (PCB) in the design. Using PCB makes for a simpler and much cleaner looking design. Also, with advances in technology, PCB changed the way we connect our circuits, which allowed designs to be very small and compact.

One of the requirements of this project was that the project had to contain a printed circuit board component. Our design is almost entirely printed circuit board, so understanding this technology is integral in the ability to create the best PCB that the team can.

In this section, PCB technology will be discussed. Such as, what is the PCB actually made of and why can this allow us to disregard physical external wires for embedded circuits. Also, the software that the team will utilize to create such a PCB as well as some constraints that they may face during their attempts at creating a fully functional GPS/INS tracking device are discussed below as well.

3.7.1 Composition of a PCB

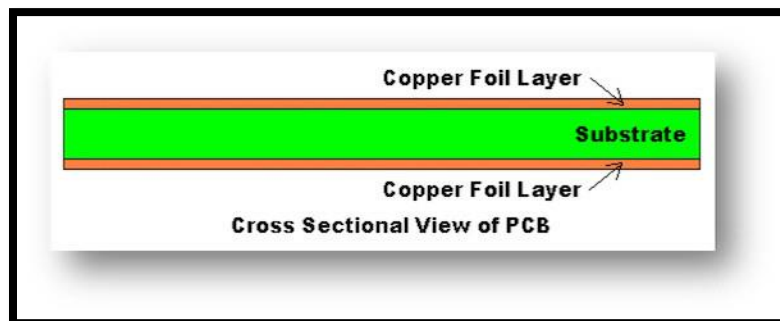


Figure 60 - Basic PCB composition

*Permission Requested [P14]

Figure 60 above shows the basic composition of a PCB includes three main parts; Substrate, Pre-Preg, and Copper foil. These are the basic materials that make a complete PCB. The substrate is a thin sheet of insulation with two sheets of copper stuck to either side of it or in some cases only once sheet of copper on one side. The substrate is sometimes called the dielectric and is a very important component of a PCB. The next part of the PCB is the Pre-Preg. This material is made of material similar to the core material, but is in a soft form and comes in standard-sized thin sheets [NNN]. Multilayered-boards can use pre-preg between these cores to create a single board with multiple copper layers. Furthermore, these sheets may be utilized to comply with thickness requirements. The third element is the copper foil. These thin copper foils serve as the conductive element in the composition of printed circuit boards [NNN]. Through an adhesive, the copper foil can be bonded in between or directly on pre-preg material.

3.7.2 PCB Design Software

As with many things that require software, there are many options to choose from. The PCB design software is usually a Computer Aided Design (CAD) software that has different components for different parts of the design process. The schematic capture part of the software is used to finalize the circuit design and how the components are connected together. This portion can commonly be combined with a circuit simulation software to simulate the behavior of the circuit before finalizing the design. Upon completion of the schematic with the schematic capture tool, the design can move to a PCB editor software. Here, the layout of the actual PCB will take place. Each component can be strategically placed and their connection can be routed in a manner chosen by the designer. Some PCB editors have an option to automatically route the connection which can simplify the design process and take the headache out of how to lay out the copper traces. When the editing of the PCB is complete, a gerber file can be created, which is used in the software of PCB manufacturing to describe the image of the PCB for printing. The PCB software is a major part of the design process for printed circuit boards, and choosing the right software is important.

3.7.3 PCB Design Constraints

There are many constraints that affect the overall design of a PCB. Some simple ones to consider are the mounting points, number of layers, board size, and fiducial points.

A fiducial or reference point is a point of reference and is used by pick and place machines. These points should be chosen to suite the manufacturer of the PCB. Fiducial points can be holes used for seating the PCB into a fixture or they can be markings that must be read by optical sensors. Depending on the design process of the manufacturer, the fiducial points must be chosen carefully.

Usually, the dimensions of a PCB is defined by the size of the overall board. It is important to consider the size of a circuit board and if all the components will fit with the specified design. It is common for an estimate to be made for the size of the PCB in the beginning of the design process.

Other things to consider are the number of layers the board will have and the location of mounting holes. Something to keep in mind is that as a rule of thumb, is that as the number of layers in a PCB increases, so will the cost to manufacture it. As for placement of mounting holes, it is

important that the area where the mounting holes will be placed is clear of copper traces and components. This will prevent physical and electrical damage to the PCB.

Updated:

3.7.4 Final PCB

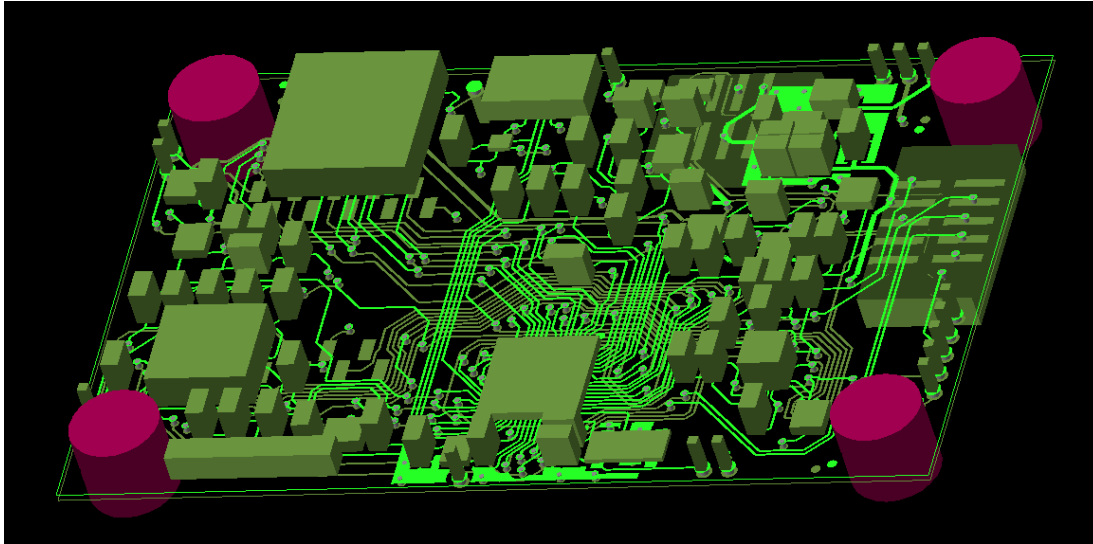


Figure 61 - Final NAT PCB Layout

The device is integrated on a four-layer PCB designed using the Cadence suite of software: OrCad Schematic Capture and Allegro PCB Designer. The top and bottom layers of the board are data and signaling traces, the second layer is the ground plane which is placed below the top layer for shielding and the third layer is the power plane which is split into three sections: The battery voltage (3.7 V), the operating voltage (3.1 V) and the USB power input. All of the passive and discrete components are placed on the top layer for ease of application by a pick-n-place machine. However, the three external connectors to the board are placed on the bottom: the power connector where the battery and signaling pins connect, the debugging/programming header and the connection to the XBee daughter board.

4.0 Related Standards and Realistic Design Constraints

Every project has its limitations that it must understand and work around or through. These come in the form of standards and constraints.

4.1 Standards

There are many sets of standards that the NAT should abide by that the team needs to have knowledge of before the design phase of the project. The design phase is very important to the overall production flow of the project, as if a mistake is made in the design phase, such as not complying with some standard, then the project can be delayed many weeks. This might cause the project to not be demonstration ready by its completion date, which is unacceptable. Therefore, the applicable standards need to be understood and met. There are going to be standards for many parts of our project, including the wireless communication, the microelectronics, the power supply, and even various coding standards.

4.1.1 Product Relevant standards

The standards will be broken down into multiple sections. First the hardware/electrical standards that we will meet during the building and design of our NAT device, then the software standards that apply to the writing of our software, the documentation standards for documents such as this, and also specifically communication standards that must be met with the radio modules.

4.1.1.1 Hardware Relevant Standards

IPC

The institute for printed circuits (IPC) is a trade association for the electronics industry and recognized globally in this industry. They were founded in 1957 and changed their name to the Institute for Interconnecting and Packaging Electronic Circuits because of their expansion from PCBs only to packaging and electronic assemblies. The organization collaboratively develops and releases a multitude of acceptability standards, which define the entire manufacturing and testing process of printed circuit boards and other electrical and electronic parts and assemblies. These standards cover things from Acceptability of Electronic Assemblies (IPC-A-610) to Performance Test Methods and Qualification Requirements for Surface Mount Solder Attachments (IPC/JEDEC-9701). Having said that, going through all of the standards laid out by IPC would be a bit beyond the scope of our senior design project. Therefore, we will discuss a few relevant topics from the IPC-HDBK-001 which is a guide to supplement the Joint Industry Standard, J-STD-001.

J-STD-001 Purpose

The purpose of the J-STD-001 is to describe materials, methods and acceptance criteria for producing soldered electrical and electronic assemblies [HHH]. To ensure quality and consistency in the manufacturing process of products, the document depends on process control methodology.

Classification

When it comes to acceptability, there are three different classes of acceptance that products can be grouped in:

- Class 1 General Electronic Products
- Class 2 Dedicated Service Electronic Products
- Class 3 High Performance Electronic Products

Class 1 refers to products that are suitable for applications where the most important thing is the functionality of the overall assembly. This is the most relaxed of the three classes. The reliability of these products are usually not very good and we can think of these like a cheap toy. The circuit board assembly will be hidden in some kind of case or enclosure where the solder joints are not visible. Product in this class typically have to meet some kind of price requirements where going to a higher class will cause it to go over the budget.

Class 2 Dedicated Service Electronic Products is the midway point. This is usually requested for non-critical electronic assemblies. It includes products where ongoing performance and longer life is required and where uninterrupted service is desired but not critical if failure occurs. Products in this class are generally designed to sustain the end-use environment conditions with little to no issues. The benefit of class 2 is that it deems a certain amount of imperfections acceptable, for example, some surface mount component may be soldered slightly off the pad and still be electrically and mechanically in tack, but can look bad aesthetically. With more room for error, time and money can be saved on the rework process.

The highest class of the standard is class 3. This is for high performance electronics products that require continued high performance or performance-on-demand is critical, equipment downtime cannot be tolerated, end-use environment may be uncommonly harsh, and the equipment must function when required, such as life support or other critical systems [HHH]. This is for high performance electronics products that require continued high performance or performance-on-demand is critical, equipment downtime cannot be tolerated, end-use environment may be uncommonly harsh, and the equipment must function when required, such as life support or other critical systems [HHH]. Usually a good manufacturer that is contracted by another company will make parts and equipment that conform to class 3 as their standard so the extra money paid to have these products manufactured are due to the extra testing and rework that has to be done to ensure high performance products.

The IPC handbook highlights the importance of understanding that the decision about the class of acceptance criteria is not related to a specific product or product category. They continue to state that the decisions need to be based on criticality of need (continued operation) and the operating environment [HHH]. Also notable, is that IPC does not define the product class, and that is defined by the user. A user is defined as the individual, Organization Company or agency responsible for the acquisition of electrical or electronic hardware.

Solder

Although our design will be sent to manufactures to be made, our team still has to perform some hand soldering tasks and will come in contact with some of the different solder types. The section

on solder covers many of the materials considerations of the items used in electronic manufacture. Solder is an alloy that solidifies below 430°F and is used for joining metals. The most common alloy is tin and lead but other alloys such as tin-silver, tin-antimony, and indium-based solders are used. For more information on the different types of solder composition and their melting points, the J- STD-006 documentation can be viewed.

RoHS (Reduction of Hazardous Substances), which took effect on July 1, 2006, is a legislation created by the European Union to reduce the harmful effects of dangerous substances to people and the environment. It has had a huge effect on the solder alloys used in electronics. The legislation established a requirement that defined the maximum concentration of lead in a solder alloy to be 0.1%. This essentially wiped out the use of lead bearing solder alloys except for special applications. The industry has turned their focus to one lead free solder alloy family in particular, Tins (Sn)/Silver (Ag)/Copper (Cu) or SAC. A few of the main reasons for the use of type of alloy are cost, metallurgical complexity, melting point, availability and process-ability.

Thermal Protection

The use of heat in any process could potentially damage thermally sensitive components. Application of heat can not only damage components if applied directly but also if applied to an adjacent component. As mentioned earlier, the industry shifted its use of solder alloy from Tin/Lead to SAC. SAC family solder alloy has a higher melting point as much as 93°F higher than the traditional Tin/Lead alloy. According to the handbook, when thermally sensitive components are identified, heat sinks should be used to isolate and prevent the induction of heat into the component body and internal elements [HHH]. Heat sinks are typically used in protecting through-hole components and surface mount component can take several different protective methods. These may include preheating in a controlled manner to reduce or prevent thermal shock, and masking or shielding adjacent components that are known to be thermally sensitive.

Electrostatic Discharge (ESD)

During the prototyping and testing of our design, we will be involved in the handling of electronic hardware which makes it important for us to be aware of and limit the potential for delayed or immediate damage to ESD sensitive electronics components. Electrostatic discharge is the release of static electricity when two objects come into contact. This is the same thing we experience when we rub our feet across a carpet and touch a metal object. Typical ESD voltages occur in the range of kilovolts. This can be dangerous for a lot of electronic components like integrated circuits that are only rated for a few volts at it pins. ANSI/ESD-S20.20 and ESD-TR20.20 ESD Handbook provide useful information on proper ESD handling and storage for electrical and electronic components.

Holding Devices and Materials

In the initial stages of setting up our test fixture, sometimes holding devices were used as an aid to keep our board in place during the soldering operation. These devices are typically safe to use and do not prevent any solder wetting and fillet formation, nor do they contaminate any of the surfaces that are being soldered. One thing to keep in mind when using such devices, is to take note not to

pre-load component leads or conductors with stress while the solder is being solidified. This is to prevent any spring-back force that can damage solder connection.

4.1.1.2 Intentional and Unintentional Radiator Regulatory Requirements

All wireless devices used will be FCC certified

When using the Ingenu wireless link: the license-free 2.4 GHz band as per FCC, 47 C.F.R, Part 15

The RPMA module operates on the license free 2.4GHz radio band. This band is license free because microwaves operate at and emit a 2.4GHz frequency to heat up food. The FCC established what frequencies unlicensed gadgets could operate on, and they decided that the band that microwaves used would be good for an ISM band or the industrial, scientific, and medical radio band. Devices that operate at this band have to fend for themselves, but they generally have such advanced technology that operating on such a crowded band is not too difficult. [HH]

When using cell systems: licensed radio band held by the cell provider (Verizon, AT&T, ect) FCC, 47 C.F.R, Part 22

This standard will be handled by the cellular service that we decide to connect over. However, the overall concept of this standard is that “Stations in the Public Mobile Services must be used and operated only in accordance with the rules in this part and with a valid authorization granted by the FCC under the provisions of this part.” [II]

Certified Unintentional Radiator

An unintentional radiator is any device which creates radio frequency energy within itself, which then unintentionally radiates from itself. This may interfere with other devices so it needs to be certified as an unintentional radiator.

4.1.1.3 Software Relevant Standards

Consistent coding practices assist software engineering by creating “a consistent look to the code, so that readers can focus on content, not layout,” expediting comprehension by justifying “assumptions based on previous experience,” and promoting the use of best practices. To facilitate cooperation as a programming team and for the benefit of those who may inherit the code, the following standards shall be followed and enforced using linting tools: [QQ]

C Programming Standards

Every microcontroller module will be coded in C. There are many C coding styles promoted by widely varying sources, so this project will defer to the eminence of the GNU project and conform to their published recommendations. [RR]. Of particular emphasis are the following:

- Source lines shall not exceed 79 characters
- Open-braces that start a function, struct, or enum shall be placed in column one; all other open-braces will be indented away from column one

- Spaces will be used before open-parentheses and after commas
- All source files will have comments explaining their purpose before the header
- The main file header comments will briefly explain the purpose of the entire program
- Each function will have a comment explaining their purpose, their arguments, and their return values
- Functions and variables shall have descriptive names or have their purpose explained by comment

C# Standards

The code that resides in the Windows Form GUI will be written in C#. The standards that the computer engineering team will follow in regard to writing such code is as defined by Microsoft in their online C# Programming Guide. [QQ] The recommendations include:

- Do not override Code Editor indentation settings (“smart indenting, four-character indents, tabs saved as spaces”)
- Include at least one blank space between method definitions and property definitions
- Place comments on a separate line; do not place them at the end of a line of code
- Include a space between the comment delimiter and an uppercase first character
- Do not use variable names to indicate variable types (they may be weakly typed)

HTML, Javascript and CSS Standards

The user-visible “view” of the Model-View-Controller implementation of the customer product tracking user interface will be written using the industry standards of HTML5 for content, CSS 3 for presentation, and the ECMAScript 2017 version of JavaScript for behavior. The HTML and CSS 3 will be coded to pass the W3C Markup Validation Service and W3C CSS Validation Service respectively. The JavaScript code will conform to the recommendations provided by the JavaScript Standard Style hosted at <https://standardjs.com/> and adopted by GitHub among many others. [SS].

Some of the rules enforced by JavaScript Standard Style include:

- Indentation using two spaces
- Strings enclosed in single quotes (avoids need for escaping)
- Spaces after commas
- Variables and functions capitalized using camelCase
- Constructor names begin with a capital letter
- Multiple blank lines not allowed
- Declare variables as their own statement
- Files must end with a newline.

PHP Standards

The backend server side components (the “model” and “controller”) will follow the server-side scripting standards used by the PHP Extension and Application Repository (PEAR) project [TT]. The PHP coding rules include:

- Four space (no tab) indenting
- Line length limit of 80 characters
- One space between control statement keywords and the opening parenthesis of the conditional expression
- No space between function names and opening parenthesis of the parameter list
- Optional curly braces included for readability and to prevent new logic errors on revision
- One space after each comma in function call parameter lists
- Class and function declarations have their opening brace on a new line by itself “K&R style”
- Variable, class, and function names capitalized using camelCase
- Private variables and methods preceded by an underscore after the dollar sign
- Source files will use Unix-Style line endings with a newline at the end of the file

Configuration GUI Specific Programming/Naming Standards

The GUI is to be written in C# in the Visual Studio programming environment. The standards for such are detailed above in the C# programming standards section.

The naming standards that we will utilize are as follows. Overall, since the configuration GUI is separated into tabs, the objects on the GUI will be named as <which tab>_<type of object>_<object Name>. Therefore, the possibilities for which tab will be wither ‘CS’, ‘DD’, or ‘TM’ for Configuration and settings, Data Display, and Testing Modules respectively. This will allow the programmer to understand which tabs controls and objects they are dealing with at any given moment, as to avoid confusion and debugging issues. Moving forward onto type of object, the options for this will be more varied. There will be textboxes delineated ‘tb’, labels delineated ‘lbl’, group boxes as ‘gb’, buttons as ‘btn’, checkboxes as ‘chk’, combo boxes as ‘cbx’, list boxes as ‘lbox’, progress bars as ‘pb’, and gauges as ‘gauge’. There are likely more that will come across during the design and programming of the GUI, however at this moment these are the types of objects that the engineers will utilize in the design and development of the GUI. Finally, the object name is less strict. The object name will reference what the object is for, such as if the checkbox is for location frequency it will be named something similar to ‘locFreqDaily’ so as the engineer will know relatively immediately what they are working with. With objects similar or close to each other, similar naming conventions will be used. An example of a full named object would be CS_chk_locFreqDaily, being the checkbox for location frequency Daily selection in the Configuration and Settings tab of the Configuration GUI.

4.1.1.4 Documentation Standards

Our documentation, while written by all members of our engineering group, needs to be consistent throughout and across all possible documents, should they need to be written. This will include possible user guides, overall project documentation such as this paper, and anything else of relevance. Therefore, there needs to be some standard that is followed while drafting documentation.

The overall standard of papers will be that the layout will be 1” margins, single spaced, fully justified, with 1.0 line spacing. The paragraphs will not be indented, instead each paragraph will

be distinguished from others with a space in between them, made with a 12-point space after paragraphs. The overall font will be Times New Roman font, and the general writing will be size 12-point font. Any headers will range from size 20-point font to size 12-point font, nothing going larger or smaller than that. Pictures will range in size as appropriate, but will remain in the

The overall standard for the paper headers is as such. Major headings will utilize Microsoft Word's built in headings, any major heading (such as 1.0 Executive Summary, 2.0 Project Description, etc) will have a Heading 1 style with edits in formatting to Times New Roman 20-point font. Any sub-heading (1.1, 2.2, etc.) will have a Heading 2 style with edits in formatting to Times New Roman 16-point font. Further subsections (1.1.1, 2.1.2, etc) will have a Heading 3 style with edits in formatting to Times New Roman 14-point font. The final heading that documentation will utilize will be the final subsection (1.1.1.1, 2.1.2.3, etc.) and this will have a Heading 4 style, which is already italicized, and will be edited to have Times New Roman 12-point font. There will be no major headings after this, and any heading not significant enough to be listed in a table of contents (such as if a subsection has organized its content under many headings, but still has only 1 major one) it will be sized at 12 point font, bolded, and offset with spacing (12-point automatically under paragraph formatting).

The header style will allow the table of contents to be automatically formed and updated. The table of contents will reside at the beginning of the document, immediately after the cover page.

All figures (tables, pictures, diagrams, charts, etc.) will be listed in a table of figures, and the figure itself will be labeled with a citation underneath the object with Microsoft Word's auto citation feature. The table of figures will reside immediately after the table of contents, at the beginning of the document.

4.1.2 Design Impact of Relevant Standards

The relevant standards that are listed above may cause design impacts on our device. The design impacts of the separate sections will be viewed separately.

4.1.2.1 Hardware Relevant Standards Design Impact

The hardware and electrical standards do not significantly affect our design, except in the case for solder. The engineers decided to solder some of the components onto our NAT Device ourselves, it will affect how the engineers do that. Also, this product will be available to consumers. With this in mind, the solder and how the consumer will be able to handle our product will be effected.

4.1.2.2 Intentional and Unintentional Radiator Certification Standards Design Impact

The design should not be affected by its FCC certification as an intentional or unintentional radiator. The manufacturers of the parts of our projects that radiate intentionally will be certified by the manufacturer. The engineering group will simply have to make sure in the end, to certify it as an unintentional radiator as well.

4.1.2.3 Software Relevant Standards Design Impact

The design impact from the software standards should be negligible, as the coding standards do not really provide constraints in our design, instead more guidelines on how to write the program.

4.1.2.4 Documentation Design Impact

The impact that the documentation standards has on the design is negligible, if anything at all. The documentation of the project will have no effect on its actual design and development.

4.2 Realistic Design Constraints

In addition to design impacts from standards that the design will follow, the device's design may also be impacted by constraints. These constraints vary over a wide array of possibilities including economic, time, environmental, social, political, manufacturability, and sustainability. These will be discussed regarding our project below.

4.2.1 Economic and Time Constraints

Trying to build a project while not knowing future obstacles is a major constraint. The team could potentially face unforeseen issues such as financial constraints. Although there is a sponsor for this project, he has set a budget of \$1200, which is to be kept. The team members all signed a contract stating that, before purchasing any component, it must be approved in writing by the sponsor.

Also, time constraints could be an issue in the sense that research and testing should be completed by the end of the summer so that the design can be started and completed by the of the fall semester when the team presents the device. The team should keep in mind the amount of time that must be devoted to this project to complete it before the deadline. The project manager has developed a timeline with all upcoming goals and deadlines to help everyone stay on task and on time with the research, design and completion of the project. The team is also instructed to document the time spent each day on the project and has an excel sheet with assigned tasks that gets updated after each task completion.

4.2.2 Environmental, Social, and Political Constraints

There are other tracking devices out there, which could lead to some competition, but this team is trying to design a less costly and smaller device. This would set the NAT device apart from other existing ones. Other issues that could pose an issue would be operating range. If the temperature reaches out of range or battery life is exceeded or extreme weather conditions such as snow, ice or a flood, the device could be compromised and either stop working or read incorrect data. There are no foreseen political or social constraints at this time.

4.2.3 Ethical, Health, and Safety Constraints

Some constraints could be of concern in the ethical, health and safety areas. A tracking device could be used to locate and stalk people which is of major concern, but this group has not designed it for this purpose and does not advocate that as a way to use the device. This group cannot control

the way someone decides to use it, but the plan is to design it in such a way that it can easily be seen from a distance unless it is intentionally hidden. Another concern is electrical shock from the device. The team will be placing the device in a sealed case so as to prevent any shock while handling. Radiation from the transmitters could also pose a major health risk, but they comply with the FCC limits as provided by the manufacturer.

4.2.4 Manufacturability and Sustainability Constraints

When designing products, manufacturability is said to be the engineering preparation of considering the ease of manufacturing. This could help to reduce future costs by fixing potential problems at the beginning stages of design and development. This device is easily designed for mass production which is cost efficient. The only way this device could potentially be affected would be if one of the components was discontinued in the near future and had to be replaced with another version.

5.0 Project Hardware and Software Design Details

This section will detail the overall hardware and software design. This will include the initial design architecture, the electrical subsystems and their related schematics, and finally the software design, including the modules and the flow charts.

5.1 Initial Design Architectures and Related Diagrams

5.1.1 Electrical Architecture

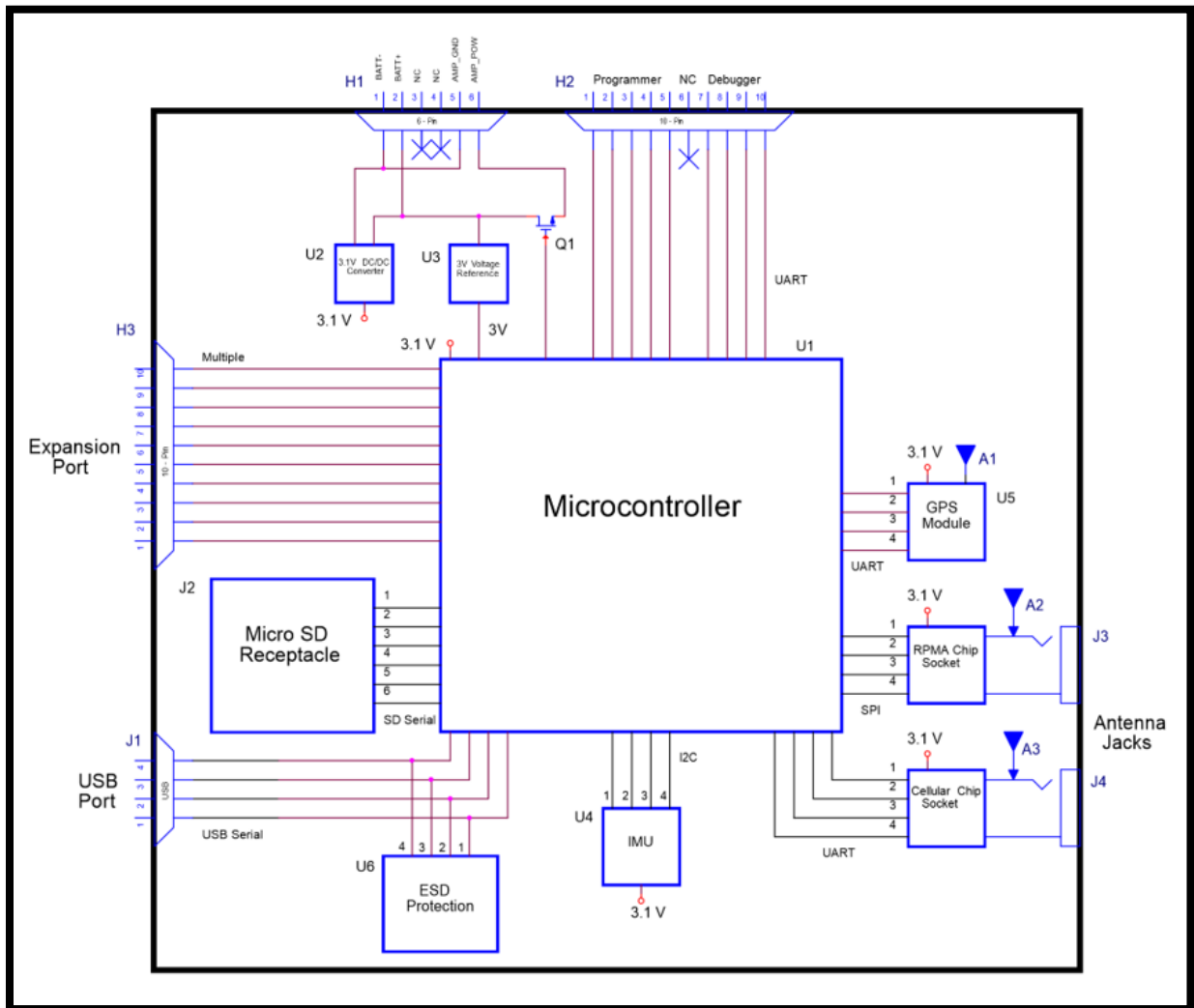


Figure 62 - Electrical Block Diagram

Figure 62 is an expansion on Figure 54 that now shows the interfaces in between components as well as the communication protocols that are used to deliver information through the device. This diagram also shows the board's external interfaces and the precise number of pins all the components need to communicate.

5.1.2 Power State of Components

	Warehouse/ Transit	Wait Motion Trigger	Motion Trigger On	Wait Location Frequency	Location Frequency On	Reed SW Pulled
PIC	X	X	ON	ON	ON	ON
CLK	X	ON	ON	ON	ON	ON
GPS	X	X	ON	X	ON	ON
IMU	X	ON	ON	X	X	X
RPMA	X	X	X/ON	X	ON/X	X
Cellular	X	X	ON/X	X	X/ON	X
SD	X	X	ON	X	ON	X

Figure 63 - Power State of Components Table

These are the possible states of the device in regards to which components are receiving power. This can happen because we have ON Semiconductors that remove power from going to certain devices depending on what components need power. The different times we need to change the power states are above.

When the device is sitting in the warehouse or in transit the device will have a magnet attached to the reed switch. This will essentially turn off all of the components, saving as much battery as physically possible. Meaning no components have power, as shown above.

Once the reed switch's magnet is pulled from the device the immediate power that turns on is the PIC the clock and the GPS to get the initial time and download the GPS's required data from the satellites.

The device has two possible states that allow it to read GPS, based on a motion trigger from the IMU and based on the location frequency which relies on the clock. When the state is motion trigger and it is not currently receiving a motion trigger, it is waiting on a motion trigger. This requires the IMU to be on and the clock to be on, but no other component is necessary, and thus to save power they are turned off.

When the motion trigger is received, the PIC is turned on, which turns on the GPS, either the LTE cellular module or the RPMA module, and the SD card so the data can be saved if the settings have set the device that way.

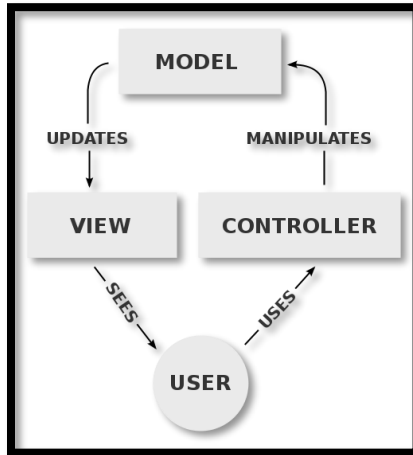
Then, the other option is location frequency, based on a setting from the user from the Configuration GUI stating that instead of a motion trigger the GPS location will be relayed on a time schedule. While the device is waiting for this time to arrive the PIC and the clock are on, as the PIC stores the time and the clock is necessary to keep the time. Everything else is powered off waiting for this time to arrive.

Once the location frequency has arrived, however, the PIC turns the GPS, the SD card, and one of the radio modules on to relay the location information to the user.

5.1.3 Software Architecture

The three software sub-projects are to be developed using three separate programming paradigms: the Web GUI will use a client/server design, while the NAT device firmware will be a structured procedural program, and finally the configuration GUI will be event-based.

5.1.2.1 Web GUI



*Figure 64 - MVC Pattern
Permission to use granted under
Public Domain*

The Web GUI will depend on three server processes, which may be collocated or may run on separate servers. An API frontend server will have a predefined web address that exposes an update script. The NAT device firmware sends a request to that address with identifying information, the timestamp, and the current GPS location in the form of latitude and longitude coordinates, which will then be processed by the server, and validated as having come from an actual NAT device with measures taken to prevent NAT device spoofing. The results will then be forwarded to the second server process, which is the backend Database Management System (DBMS). The DBMS will receive SQL INSERT requests from the API frontend, updating the location log for all NAT devices in service. The DBMS will also service SQL SELECT queries from the third server process, the Web GUI proper, which will be based on a Model-View-Controller framework (MVC). The customer user navigates a web browser to the publicly accessible web address, where a “controller” component

obtains data from “model” components containing the logic that produces the database queries, and then formats the returned data according to a defined “view” of the user interface. In bridging the models and views, the controller allows them to work in isolation from each other, displaying the needed information to logged-in users without exposing the back-end implementation. The relationships of the MVC paradigm are diagrammed in Figure 64 [VV].

5.1.2.2 Firmware

The NAT device firmware components are defined as a sequence of modules, each servicing a separate peripheral according to its specific interface. The USB library being used requires a cooperative multitasking environment, and so each module will execute under the expectation that it will cede control to the next module when it has nothing to do at the present time in an infinite loop, broken only by transitions to low power mode and returned to by configured timer or peripheral interrupts. Care will be taken in implantation that no long-term blocking code is used, deferring to the next loop as often as necessary for the smooth operation of all firmware modules.

5.1.2.3 Configuration GUI

The configuration GUI is a .NET-based Windows Forms application, designed around mouseclick and USB events, calling the appropriate code based on the current state of the form controls. For example, code that polls the NAT device for its current configuration will only send requests for

those parameters that are visible on the current tab of the form. For example, the GUI will not poll for specific IMU data readings from the Configurations & Settings tab, only the Data Display tab. Other functionality, such as the View SD Card Contents dialog, will likewise only execute in response to its activation button being clicked. The architecture of the GUI is thus two-layer, with presentation handled by the Windows Form code generated by the Visual Studio design tools, and behavior defined in event handlers for each form element.

5.2 DC/DC Converter.

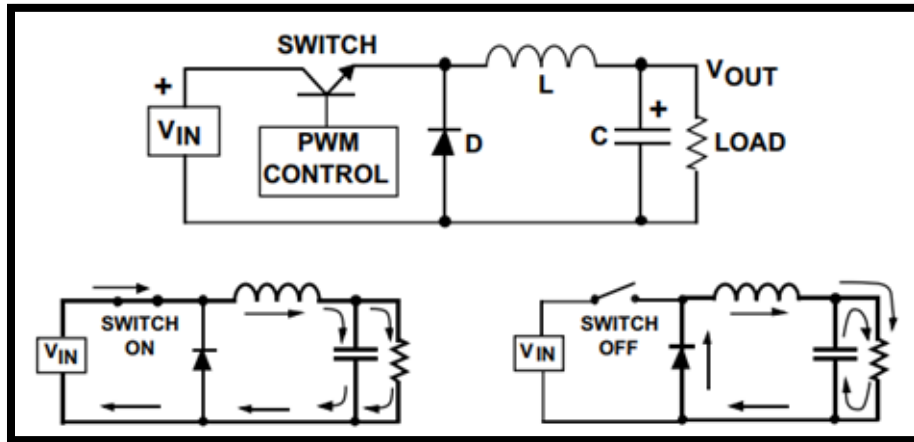


Figure 65 - Buck Regulator Operating Cycle. On State (Left) Off State (Right)

While L1 does help to maintain a consistent voltage at the output it also creates a fair bit of voltage ripple which can be smoothed by a bypass capacitor C4 from the output to ground. L1 and C4 must be chosen such that the output current ripple is acceptable but care must be taken to not slow the transient response of the regulator. Lastly, a bypass capacitor C3 should be placed from the Input pin to ground to regulate the input voltage under High load.

Nominal values of $10\ \mu\text{H}$, $10\ \mu\text{F}$ and $4.7\ \mu\text{F}$ for L1, C4 and C3 respectively, shown below in Figure 66, are recommended by Texas Instruments and these are used for the initial design. Above in Figure 65 is a diagram of the operating states of a typical buck regulator.

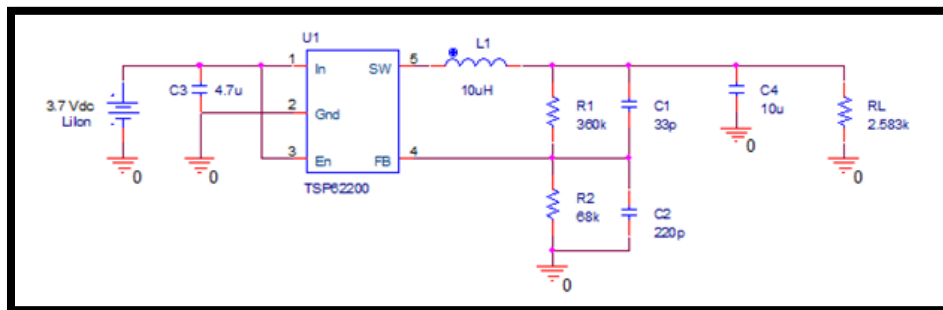


Figure 66 - TPS62200 Application Circuit

5.3 Voltage Reference

Voltage references have a lot in common with voltage regulators in fact both devices are classified with the same parameters, but voltage references have much tighter output voltage tolerances. The first very simple form of a voltage reference is extremely easy and cheap to implement, it consists of a forward biased diode. These voltage references also have the benefit of being small because they can be implemented by using one of the p-n junctions in a BJT. The second type of voltage reference is equally simple and of a comparable price it also uses a forward biased diode along with a reverse biased Zener diode. The above voltage references are shown in Figure 67.

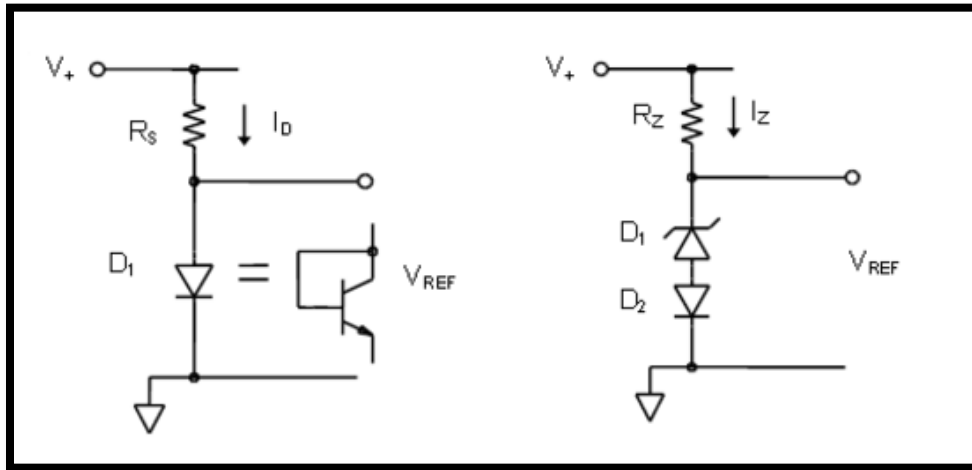


Figure 67 - (left) Diode Voltage Reference Implemented by a BJT (right) a Zener Avalanche Voltage Reference [E]

*Permission Requested [P22]

Both of the above voltage references have advantages in their simplicity but they are susceptible to temperature variations which is useful for temperature sensors but precision voltage references that are used to calibrate a 16-bit ADC or DAC requires accuracy lower than ± 1 mV and rigorous initial calibrations. Temperature variations in a voltage reference become even greater with age and use and are often overlooked as a source of noise in a device. The need for a voltage reference that maintains high accuracy over long periods of use stimulated the development of the third kind of voltage reference, the band gap voltage reference.

Bandgap voltage references can be design in many ways but the two basic concepts of a bandgap voltage reference are: two BJTs working at different current densities and a positive and negative temperature coefficient counteracting one another. A basic bandgap voltage reference is shown in Figure 68. A bandgap voltage reference can be implemented in a small IC and maintain the high accuracy that is needed to run devices such as a ADC, DAC or in the case of this project a microcontroller. An application circuit of such is shown below in Figure 69.

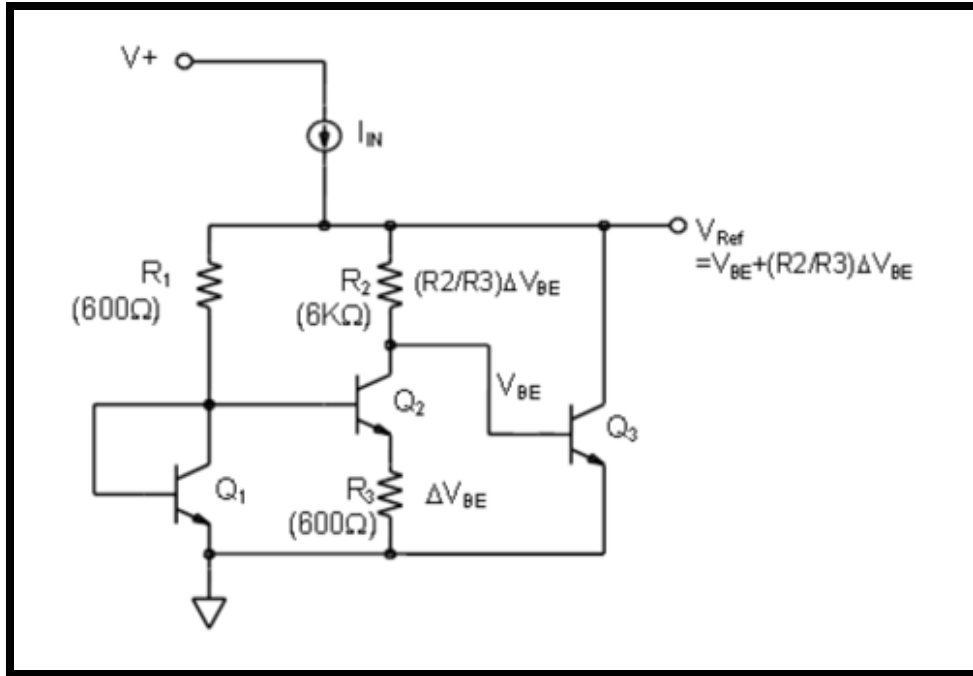


Figure 68 - Basic Bandgap Voltage Reference [E]
 *Permission Requested [P22]

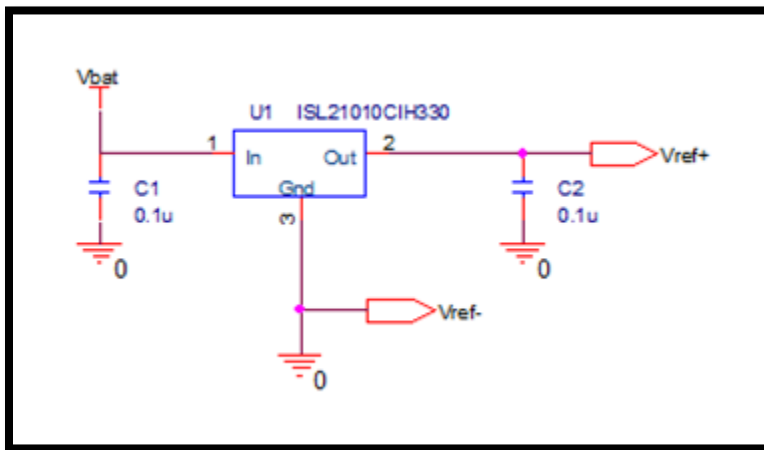


Figure 69 - ISL2108 Application Circuit

5.4 Microcontroller

The microcontroller requires one external capacitor on the VCAP pin. Two crystal oscillators (32 MHz and 32.768 MHz) need to be placed on the OSC1, SOSCI, OSCO and SOSCO pins to provide clock signals in order for the microcontroller to function. These oscillators also need to be stabilized by two bypass capacitors C19 – C22 (More on oscillators below). Our microcontroller also features input/output ports that can be configured to use different communication protocols

(UART, SPI, I²C, etc.) and attached to these are various other components including: a micro USB port, a micro SD card slot, an IMU, the GPS module and the LPWAN modules.

Switches

Also, throughout the device FDN358P p-channel MOSFETS (denoted by Q#) are used as active low logic switches with the gate attached to a signaling pin on our microcontroller. The components are used as switches by manipulating the voltage at the gate through code executed by the microcontroller, if the microcontroller drives the gate of one of the MOSFETs to a high voltage level of 3 V it acts as an open switch and if the microcontroller drives the gate of one of the MOSFETs low, to 0.4 V, then it acts as a closed switch. A few examples of where these switches are used are: to provide power to the SD memory card (Q9), to activate and deactivate battery voltage output terminal (Q11), and to turn the GPS/RPMA/LTE modules on and off (Q14, Q16 and Q15). If the source of one of these switches is tied directly to the battery supply voltage of 3.7 V then a pull-up resistor R1, shown in Figure 70, is needed to open the switch by default until the gate is driven low. If the switch is tied to the regulated voltage of 3.1 V this pull-up resistor is not needed.

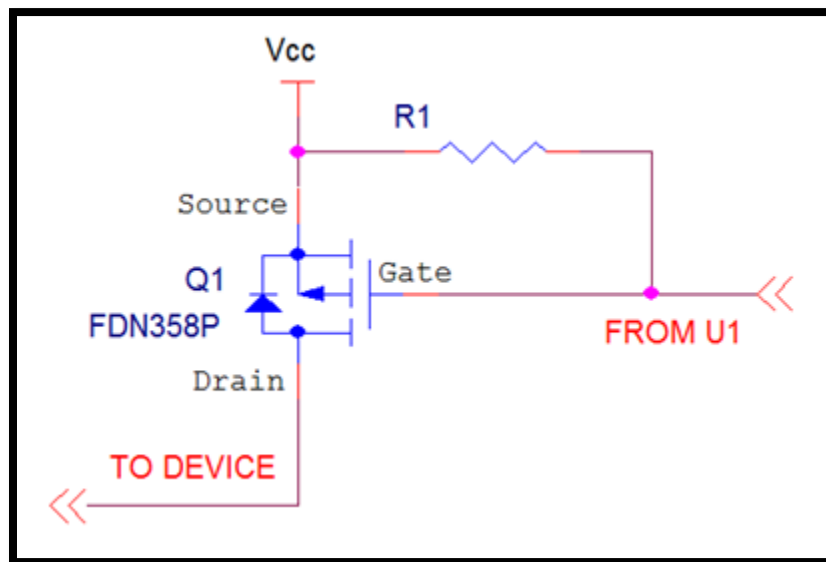


Figure 70 - MOSFET Switch

Oscillator

A Crystal Oscillator is a device that makes use of the piezoelectric properties of a crystal to create an electric signal with a precise frequency. The frequency created by one of these crystals is so consistent and accurate that many digital watches use them to keep time. Other applications of these devices include: clock signals in digital circuits and transmission stabilizers for radio communication. The phenomenon that causes this device to operate is part electrical and part mechanical. When a voltage is applied across one of these crystals it physically distorts, known as electrostriction, and when the voltage is removed the crystal creates its own electric field as it restores its original shape. A crystal oscillator behaves much in the same way as an RLC circuit.

In fact, Figure 71 shows the equivalent representation of a crystal oscillator using passive components.

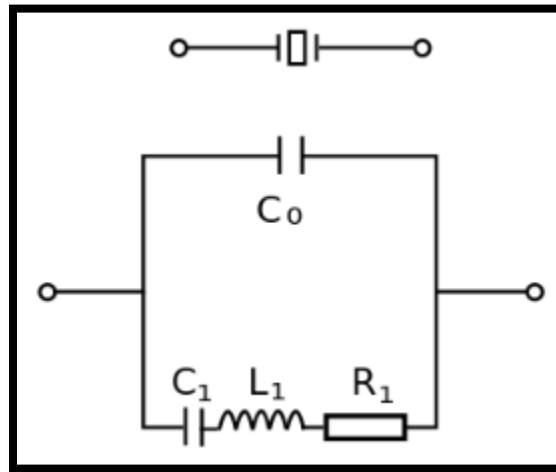


Figure 71 - Crystal Oscillator Equivalent Circuit

*Permission Requested [P23]

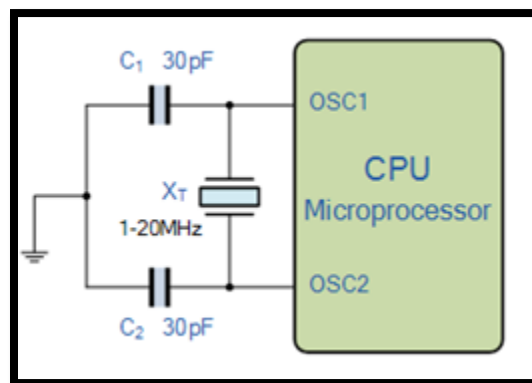


Figure 72 - Crystal Oscillators used with a Microcontroller [I]

*Permission Requested [P23]

Here R_1 , C_1 and L_1 are used to model the mechanical vibrations of the crystal (Friction, Inertia and Stiffness) and C_0 is used to represent the parallel capacitance of the electrode plates. A crystal oscillator will produce two different frequencies depending on whether it is placed in series or parallel with its load fittingly called the series frequency f_s and parallel frequency f_p , the series frequency is usually a few kilohertz lower than the parallel frequency. Figure 72 above shows a typical application of a crystal oscillator with a microcontroller.

5.5 Software Design: Modules

This section will focus on the specific programming modules that our product will utilize. The code will be in the MCU and will be run when the hardware modules deem it necessary. The code will be broken down into modules. Each module will discuss at what point the module would be

called and executed, its overall inputs and outputs, and the overall function of the module, as well as any functions/modules it will call external to itself.

5.5.1 GPS

The GPS module will be executed any time the product needs to relay its current GPS fix. This may happen in a multitude of ways. For some applications, this will occur on a motion trigger. Our device will have an IMU, and therefore once the device receives input from the IMU indicating that it is moving (whether that be a jostle, an acceleration, or a rotation) the GPS will determine the location with a satellite fix and relay that information to the MCU, which may then send it to either the on-board memory, the wireless communication module, or both. Another application will allow the user/client to determine some set time frame that the GPS will find its location and relay it. This will be set by the configuration GUI. When the clock determines that the GPS should relay its data, the GPS will determine the location and, again, send it wherever the data is required. There may be other functions that turn on the GPS, such as on start-up, initial configuration of the client, and module testing, but for the overall use of the device, these are the two main scenarios the GPS will encounter.

The GPS module itself will not take any inputs. This is because of the scenarios above. The GPS will not stay on constantly. That would drain the battery and relay more information that may be necessary, such as if the device is not moving or constantly on the move but always within GPS signal. However, the inputs that the GPS hardware itself may need to turn on will not be held within the GPS module. The GPS module will determine what will happen when the module is on. The MCU should determine what turns on/off the GPS module.

The GPS module will output GPS data as National Marine Electronics Association (NMEA) formatted strings to the MCU via onboard UART. This data will eventually be turned into either a radio message over the wireless communication that the device is currently using (either RPMA or cellular), or it will be saved as a defined message in the on-board memory. Both could also happen, depending on the application, and all of this is determined by the MCU, not by the GPS or GPS code module. The reason the data may be saved in memory would be so that should the device lose GPS signal, it can use its last known value to transfer into INS and determine the current location based on an INS algorithm. The INS algorithm will utilize the GPS location and the IMU data (which relays information based on the devices motion) to calculate where the device current is.

The helper functions the GPS code module may use is finding the last saved data on the memory, to be used to possibly help the GPS itself find a fix faster using an aided start. The GPS code module may also change the over focus of the MCU to the INS, should the GPS lose signal.

The overall function of the GPS code is to obtain the data that the GPS hardware receives and output it to the MCU.

5.5.2 Configuration GUI

The Configuration GUI is an important software component of the NAT device that allows users to configure the device in the best way for their specification and specific implementation.

Overview

The configuration GUI is less of a module and more of an entire program in itself. However, it is self-contained section of code, removed from the code residing within the MCU. Therefore, it will be described here as though it is a module instead of a whole program.

The configuration GUI will program various settings in the hardware. The GUI will be a windows form, accessible only to the person initially turning it on. In our project, however, will have also expanded it to see some of the devices data values, such as the values being read from the IMU and the GPS. This will be for testing, demo, and presentation purposes. The final configuration GUI is not being designed to include these features. Features such as these will be included in the optional client GUIs (windows form or web-based or both).

The input to the GUI is a configuration message from the device. The MCU in the device will send the GUI the current configuration settings that it has. It will also receive data, through the MCU, about the state of all of the hardware modules on the device. This will tell the GUI whether the modules are all connected and active, as well as in the case of some of the modules, what data they are outputting.

The output from the GUI will be a configuration message to the MCU if any of the configuration settings are changed. It will also output any necessary data required to test any of the modules. This may include sending messages over the two wireless communication modules, testing to make sure that the messages are delivered and acknowledged properly. It will also send a message to the various modules, via the MCU, asking for their data.

The GUI will utilize functions from almost all the functions in the MCU. The GUI function, to both set the configuration settings and to acquire the GPS location. The IMU function, to set the configuration settings and to acquire the IMU motion settings. The wireless communication modules, setting the configuration settings as well as testing the RPMA and Cellular communication links.

The overall function of the Configuration GUI is to set the configuration settings within the devices as well as viewing the data output from some of the modules. This is intended to be used by the person deploying the device to the client. They will configure the device as the client needs, for their application. This is necessary because our device is very versatile, for a asset tracking device.

The Software: Microsoft Visual Studio

The Configuration GUI will specifically be coded in the Visual Studio environment. This is an easy software to create and code Microsoft Windows based forms that can be connected to the Microchip Microcontroller. Visual Studio is an IDE (Integrated Development Environment) with an integrated debugging environment that includes a source level debugger and a machine level debugger, allow the engineers of this project to debug both the software itself and its connection to the hardware, finding issues wherever they may come from. The important feature for this project is that it includes a designer to allow windows forms to be designed with its own GUI, instead of requiring code to be written to create the interface. This is invaluable in designing such a detailed form, with many objects to keep track of. Also, since it is a Microsoft Windows software, it allows easy integration and cooperation with both the Windows operating system, with direct

commands into opening and using Windows based software. This will be valuable should our design need to open a file explorer for the SD card component, or if we need to directly access the computers time and date, or information of the like.

Layout and Usage

The configuration GUI is a Windows Form written in C# in Visual Studio.

Figures 73, 74, and 75 below show what the draft of the current configuration GUI looks like. The configuration GUI is split into 3 tabs, which are shown at the top of each figure, Configuration and Settings, Data Display, and Testing Modules. These are split so that the GUI is easier to use for the reason that the user needs it for. The Configuration GUI that is detailed here and elsewhere in this document is referencing the GUI that is shown in the figures below. These, however, are currently just drafts. They may be updated, manipulated, or discarded depending on how the project develops. Currently, this Configuration GUI will connect to the NAT device over the UART connection with support from the ICD (In-Circuit Debugger). Upon opening the GUI, the screen will show Figure 73. The code that actually allows for functionality will be described in another section within this module's overall description.

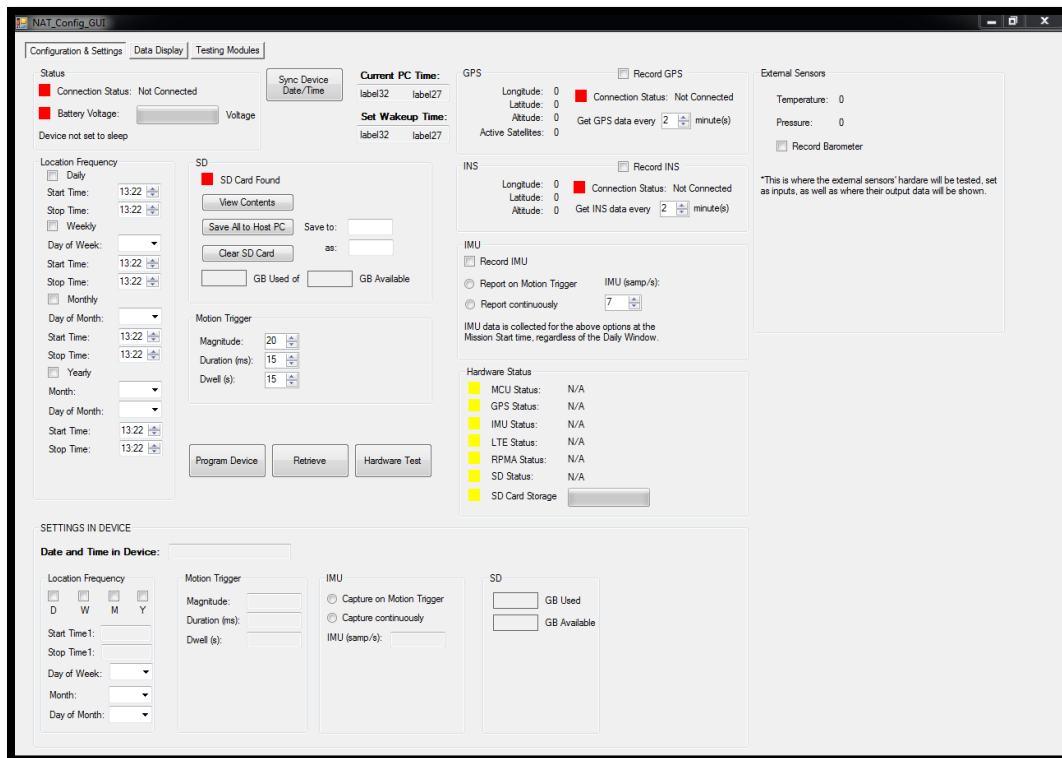


Figure 73 - Configuration GUI - Configuration and Settings Tab

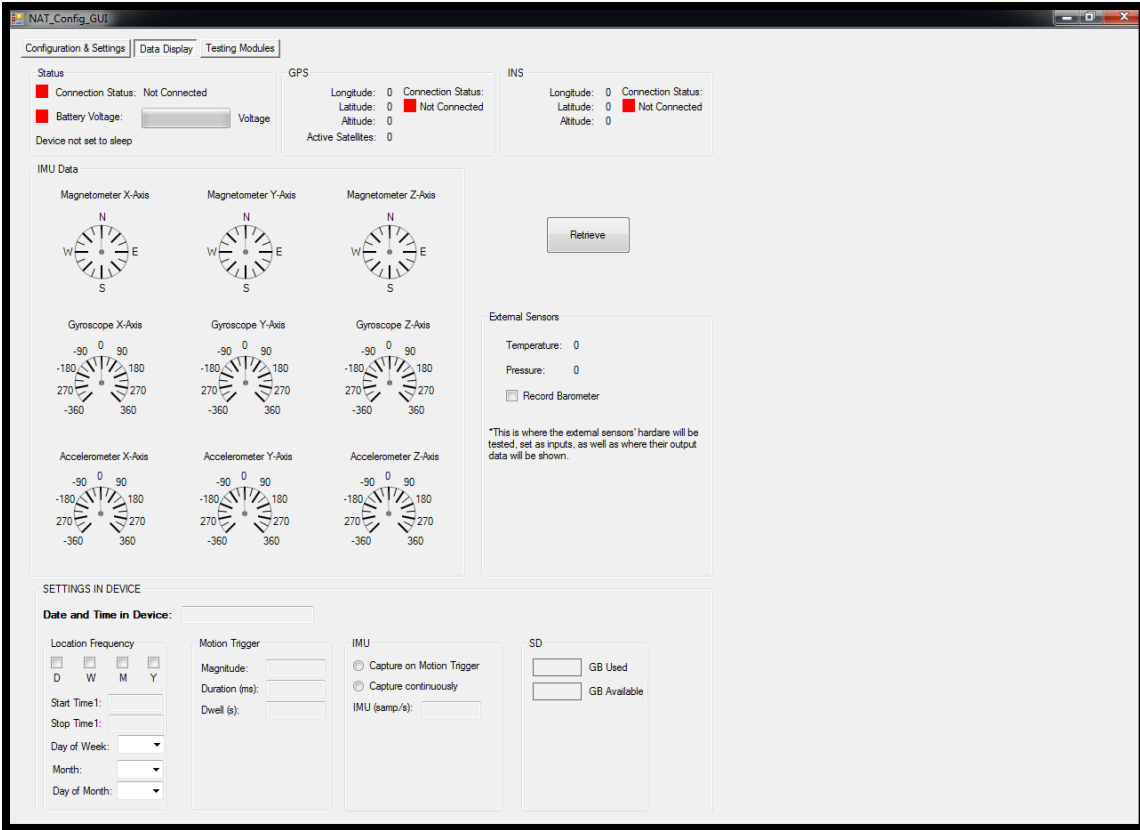


Figure 74 - Configuration GUI - Data Display Tab

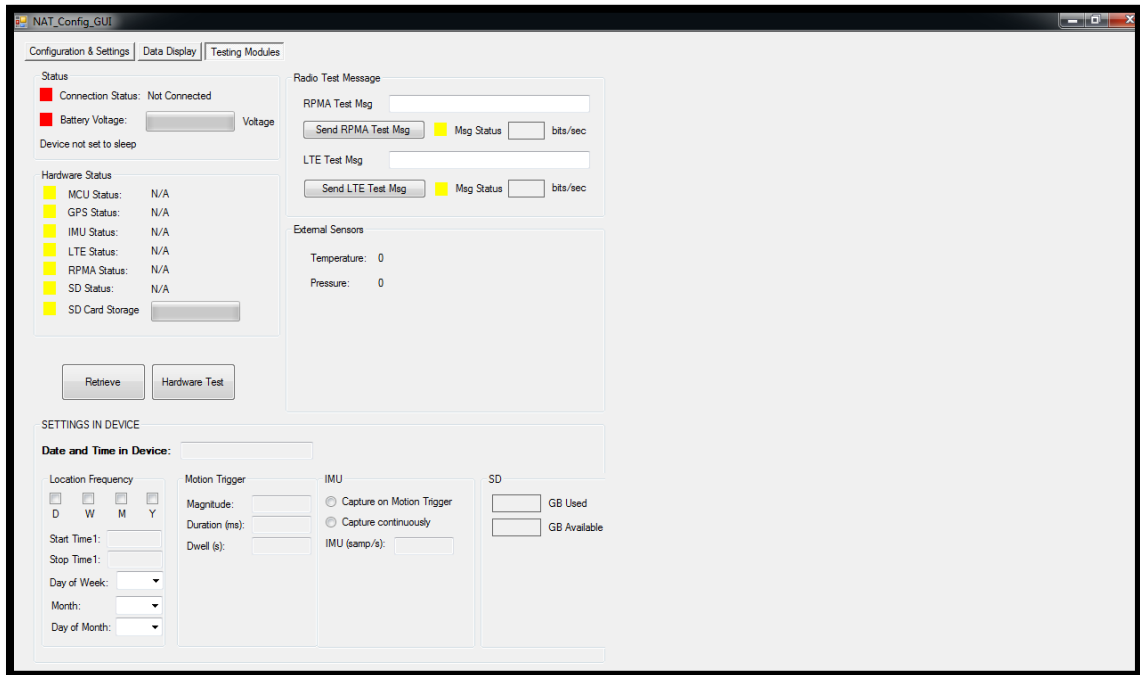


Figure 75 - Configuration GUI - Testing Modules Tab

The first tab is the Configuration and Settings tab, shown in Figure 73. This tab includes some of the settings that can be programmed into our device. Beginning in the upper left side of the figure, the Status group box is shown. This group box is in every one of the tabs. The reasoning behind this is that one of the uses of the configuration GUI is to debug any issues with the modules in our device. Therefore, while we are debugging issues, we do not want to mistake something not working properly with the device itself not being connected properly. Therefore, the status group box is included in every tab. Making sure that the proper problems are being debugged, not a lost connection with the device or a dead battery. The status box will show the connection status of the device, such as whether it is receiving/transmitting messages properly, and the state of the battery, such as whether its voltage is at an acceptable level, or a warning level, or a low level. When it moves between these levels, the value on the progress bar will move accordingly, and will change colors from green, to yellow, to red as it moves between the 3 states of the battery.

The section next to this is the setting of times and date, through the syncing of the current PC time, and setting the wakeup time, for when the device is shipped by the manufacturer. More specifically, syncing the time and date of the device is done here, which is necessary because of a myriad of different setting that will be discussed later. Therefore, the easiest way to set this value is to set it to the current time and date the host PC has. Then, the wakeup time is so that the device can sit in an ultra-low power mode while it is in transit. Thus, the manufacturer will set this value based on how long they expect the device to take shipping to the user. There is also a button that is not shown here, as it is hidden until the device is properly connected, which sets the device into this low-power sleep mode.

The section immediately to the left, is the GPS configuration settings. Here is where the user will see the values currently being read by the GPS module on the device. They can use these values to make sure that the device is reading properly, as the user can determine the GPS location of the host device they are using to compare to the values read of the device. The user can also immediately see if the device is connected at all, to make sure any configuration issues are not the cause of being improperly connected. This box, shown red in Figures 73 and 74, will change in regards to the connection status, red being not connected and green being connected. The next setting here is to record GPS, meaning that if the user should check this box the GPS values will not only be available to be read by the radio modules and sent directly to the user over wireless communication, but also to save these values to the flash memory that is on board the device. There is also the setting that tells the device how frequently to receive GPS data, currently set to vary between 2 and 30 minutes, but in a update of this GUI this may be changed to include smaller intervals. This frequency means that when the device reads a motion trigger, the specifics of such to be discussed later on, how frequently the device should read and report the GPS value to the user. The more frequently this value is read, the more accurate the data, however it will also drain the battery quicker than it would with a less frequent reading. Therefore, this setting has to be set for each specific use, as the user knows which is more of a concern for their application, battery life or real-time accuracy.

The next setting discussed is below the settings for GPS, this is the INS readings which are similar to the settings that were just discussed for the GPS. INS, as discussed in this document, is the Inertial Navigation System which used the values read from the IMU and calculate using a special INS algorithm the new location of the device. These calculated values will be shown in this group box as they are calculated by the device. The record INS is the same as the record GPS, it indicates

to the device that the user wants the INS values that are calculated to be saved to the on-board flash memory. It also shows the connection status and lets you set the frequency of the readings, same as GPS.

Moving back to the left side of Figure 73, there is the location frequency group box. This group of settings will allow the user to determine that, should they not wish to use the motion sensor, how frequently they would like the device to turn on and then output the values of the GPS/INS. This would be convenient if the device is connected to an application that will not trigger the motion sensor very often. Applications such as this may be a shipping container staying idle for days. This application will not trigger the motion sensor on the IMU. Therefore, the device has to have a way to report its location, even when not being triggered by the motion sensor. The options that our device has for a setting such as this is reporting the location daily, thus setting the time of the day when it should turn on and send/save its location data, weekly, setting the day of the week and time of day, monthly, setting the day of the month and time of day (currently only option being days 1-28 as to be inclusive for the months that only have these months), and finally yearly, setting the month and day of month and time of day. Only one choice can be chosen, the programming of the Windows form will guarantee this, and this choice will be set into the device via the configuration message.

The SD configuration settings are enclosed in the group box next to the location frequency. The first item in the group box is another indicator of whether or not the SD card was found, and therefore connected. This checks for both that the SD section of the device is connected properly, and that there is a valid SD card in the SD card mount. Then there are a series of buttons that allows the user to execute some functions with the SD card. First, they will be able to view the contents of the SD card. This will show them all of the values that the device has been saving to the on-board flash memory. The next button will allow the user to simply save everything from the SD card onto the host computer they are using. This will be done with the series of text boxes located next to the “Save all to Host PC” button. The “Save To:” textbox will open a file explorer that will allow the user to designate the file path, and then the “as:” textbox will allow the user to designate what they want the file to be saved as. The button underneath the “Save All to Host PC” is the “Clear SD Card” button, allowing the user to simply clear all of the contents from the SD card. This will give the user functionality to easier reuse our device on a new application of theirs. Then, underneath all of the buttons, a sentence with 2 textboxes are seen. The message reads “___ GB used of ___ GB available”, which will be filled in with the current values the SD card has in the device. These values come from the configuration setting message that the device sends to the GUI, regarding all of the configuration settings that it currently holds.

The motion trigger settings are in the next group box, below the SD card settings. This will allow the user to set which values the IMU will read that will indicate to the device to start outputting GPS data. The magnitude of the motion, the duration of the motion, and the dwell of the motion are all items that the IMU can read, and if they are greater than the values set here, the device will turn on the GPS and report back its values. This will protect the device from being turned on by any minor movement, thus saving battery. A type of application that would benefit from this would be an application where the object is constantly moving, such as swaying on an electrical wire or moving on a boat. This would allow the user to indicate to the device to disregard the constant motion that the application will always experience, and instead look for one of more significance.

Next to these motion trigger settings and below the aforementioned INS settings are the IMU specific settings. The IMU is used for both the INS algorithm and for the motion trigger settings. However, it has a few settings of its own to be set. First, regarding our application specifically, the user can set whether or not to record the IMU values to the SD card. This may benefit the user in some application, but it also may drain the devices battery fairly quickly. Thus, the user must determine if this feature is worth losing a longer battery life. Then, the user is allowed to choose whether the IMU will report on a motion trigger (those settings set in the motion trigger group box explained immediately above this) or continuously. One will significantly drain the battery more than the other, and thus again the user must take this into consideration when determining which settings they prefer. The user can also set the samples/second that the IMU reads, which is limited by the performance of the IMU that was chosen. The device is defaulted to 7 samples/second, but may be changed in a later version of this device.

Underneath IMU settings are a nice view of the various hardware components' status that the NAT will utilize. This grouping will allow the user to easily see in one space which components are and are not connected properly, as well as how much SD storage is left in the SD card.

The final editable settings are the external sensors group box. Currently, there are no external sensors connected to our device. However, the reach goal of this specific project and a final goal of the device is to allow external sensors to be easily connected (essentially plug and play) into our device, and the GUI will allow the user to see the values that are currently being read, and allow a few of those settings to be recorded to the SD card. The small number of values that are shown in this draft of the GUI are just an example of what something like this may look like, and ample space was left so that the GUI could have the additional external sensors added easily, without much formatting edits to the GUI overall.

Before the final group of settings is discussed, the functionality of the 3 buttons that are shown underneath the motion trigger and to the left of the hardware status group box. These buttons read "Program Device", "Retrieve", and "Hardware Test". The "Program Device" button will allow the user to implement the changes to the settings that they have made. Anything that they have changed in the editable boxes will be implemented into the device, via a configuration message that the GUI will send. The "Retrieve" button will request from the NAT device the settings that it currently is holding. Therefore, if the "Program Device" button has worked properly, upon hitting "Retrieve" the Settings in Device values will update and change according to the values the user just set. The final button, "Hardware Test", will allow the user to update the various connection status boxes that are included within this tab, other than the overall connection status indicator which will update regularly so that the user can easily debug if a problem they are having is due to lost connection.

The final group of settings is found at the bottom of this tab, as well as all of the other tabs (shown in Figure 74 and 75). This group box will show the user what settings the device currently holds, and will be updated by the "Retrieve" button shown above it. Upon hitting the retrieve button, the device will send the GUI a configuration settings message with its current settings and the GUI will update the output accordingly. Therefore, it will be un-editable to the user directly. Should the "Program Device" button not work properly, this will be shown by the settings not updating accordingly.

That is the overall look at the first tab, Connection and Settings. This tab's main function is to set and view settings the device has and the specific functionality is described in the explanations above.

The next tab is shown in Figure 74, Data Display. This tab will not necessarily continue onto the final draft of the Configuration GUI for the product as it moves onto market. This tab was specifically made for the use of our team during our Senior Design project. What this tab will show is a few of the data outputs that the device will have, grouped together in one tab so that the team can easily see what values the device is reading.

The first value that is seen is, again, the Status group box. As previously discussed, it will simply show the current connection state of the device and of the status of the battery.

The next values are the GPS and the INS values. Theoretically, if the GPS is connected the INS should not be, and visa versa. These group boxes will show the user what values it is reading. These will change when the user clicks on the "Retrieve" button that is located below the INS group box.

Below the Status, GPS, and INS group boxes is the IMU data. This includes the set of 9 axis that the IMU will read. The IMU module reads magnetometer values for all 3 dimensions, gyroscope values for all 3 axis, and accelerometer values for all 3 axis. These values will be shown on gauges, moving the various needles as the device moves.

There is also the values that are read by the external sensors in this section. These are not being used by our device at this particular moment, but are there for any future update to the device that allows these sensors to be added to our device and read from the GUI.

The final section is the same Settings in Device from the first tab, and are updated upon hitting the "Retrieve" button, though nothing in this tab will update the settings in the device.

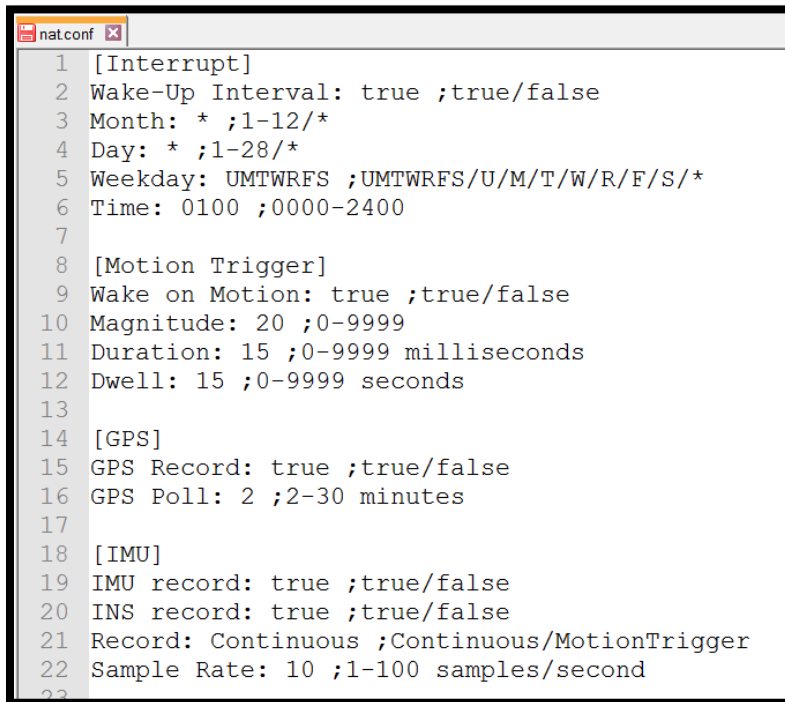
The last tab is the "Testing Modules" tab, as seen in Figure 75. The group boxes that are the same as the previous tabs are the Status group box and the Hardware Status group box. They do the same things in this tab as they did in the last. Also, the Settings in device group box are the same as the previous tabs as well.

The new group boxes are the Radio Test Messages group box. This box allows the user to input a message to be sent over either the RPMA or the LTE modules on the device. The GUI will output whether the messages are being received and what the message bandwidth (bits/sec) it achieves on the path. This will allow the user to select between the RPMA module or the LTE module, as long as both are working properly. The user should be able to choose which module is working better, and then make sure that that is the main module to be used to send messages. Should this module stop working, then the device will try and use the other module in an attempt to still send messages.

Communication with the NAT Device

The NAT device firmware and the Configuration GUI have to be able to speak with each other. The computer engineers have decided that they will do such by sending and receiving a text document between the host computer and the device. The device will store a copy of this text

document and will send its newest version to the Configuration GUI upon the GUI's initial opening and connection to the device. The text document naming convention will be nat.conf, and will be formatted as follows, shown in Figure 76.



```
1 [Interrupt]
2 Wake-Up Interval: true ;true/false
3 Month: * ;1-12/*
4 Day: * ;1-28/*
5 Weekday: UMTWRFES ;UMTWRFES/U/M/T/W/R/F/S/*
6 Time: 0100 ;0000-2400
7
8 [Motion Trigger]
9 Wake on Motion: true ;true/false
10 Magnitude: 20 ;0-9999
11 Duration: 15 ;0-9999 milliseconds
12 Dwell: 15 ;0-9999 seconds
13
14 [GPS]
15 GPS Record: true ;true/false
16 GPS Poll: 2 ;2-30 minutes
17
18 [IMU]
19 IMU record: true ;true/false
20 INS record: true ;true/false
21 Record: Continuous ;Continuous/MotionTrigger
22 Sample Rate: 10 ;1-100 samples/second
23
```

Figure 76 - nat.conf File Contents

The way that the firmware and the software will read this document is that it will look for the '[' character, which will indicate to the code parser that a group of variables is coming next for the section that is stated by the string in between the '[' and ']' characters. Such as for '[Interrupt]' the parser will know the variables that are read between the ']' character and the next '[' character are grouped in this section of variables, related to Interrupts. Then the next string including all of the characters before a ':' character indicate which configuration variable is being set. Such as for "Month" the variable is the month of the interrupt, being some value between 1 and 12 or no number at all. Then the value between the ':' and the ';' characters is the value that configuration setting holds (in a case of a message from the NAT device to the GUI) or a value that the Configuration GUI wants to set in the NAT device (in a case of a message from the GUI to the NAT device). Such as in this example the GPS poll setting is/is being set to 2 minutes. Then finally, solely for the use of a human reading this file, the characters after the ';' character are any possible values the variable could take. This is how the whole file is read, and allows it to be expanded simply and easily, especially in the case of possible external sensors.

Explanation now of the specific sections of the code, the first section being 'Interrupt'. This section is for either the wakeup time of the device overall or the location frequency values of the device. The first configuration setting, 'Wake-Up interval', indicates that the device will be configured to use location frequency to report its location. For a daily location frequency, the value in 'Month' and 'Day' would be '*' indicating that there is no specific month or day included in the frequency,

and the 'Weekday' would be 'UMTWRFS' indicating every day of the week the GPS location will be reported. The "Time" would be at exact what time of day would the GPS be turned on, and the location sent to the user. For a weekly location frequency, the value in 'Month' and 'Day' would be '*' indicating that there is no specific month or day included in the frequency, and the 'Weekday' would be one value such as 'U' for Sunday, 'M' for Monday, 'T' for Tuesday, 'W' for Wednesday, 'R' for Thursday, 'F' for Friday, or 'S' for Saturday. The 'Time' could be any time. For a Monthly location frequency, the value in 'Month' would be '*' indicating every month the location should be taken, and 'Day' would be whichever actual number date of the months that the user wants to see the location, such as every 5th of the month or something of the like. The Weekday would be '*' indicating that there would be no weekly value, as in the day of the week doesn't matter, only that it happens on the 5th. The 'Time' could be any value. For a Yearly location frequency, the 'Month' value would be some number from 1-12 indicating which month of the year January to December the location should be reported on. The 'Day' value would be also some number 1-28 indicating which day of the month the location should be reported on. The 'Weekday' value would be '*', as it doesn't matter what day of the week the yearly date falls on, and the 'Time' value could be any number 0000-2400.

The next section is the 'Motion Trigger' section, meaning what values the IMU has to read for the motion of the device to turn on the location services, like the GPS. This is to prohibit any jerk in motion from turning on our device and wasting precious battery life. First, the setting 'Wake on Motion' indicates the device is configured to report location on a motion triggered basis, not a location frequency. The intensity of the motion that wakes the device is dependent on the next values listed. 'Magnitude' indicates how strong the motion must be to wake the device. 'Duration' is how long that motion must go on for. And 'Dwell' is how long the device must remain in the motion state.

The 'GPS' specific settings are next. 'GPS Record' indicates whether the GPS, in addition to real-time data being sent to the user, should it also record the data it reads and save it to the on-board SD memory. True means it should record to the SD and false means it should not. Then, the 'GPS Poll' means that the device should report data, while the device is on and live, on what time interval. The options range between 2 and 30, meaning the GPS can report its location between every 2 minutes and every 30 minutes.

The final section that we have for configuration data is the 'IMU' settings. These settings are the values the IMU module specifically cares about. 'IMU Record' is the virtual same as 'GPS Record', as it tells the IMU to also save its data in addition to real time reporting it. The 'INS Record' is a place holder for when the project extends into implementing an INS algorithm for when the GPS cannot receive signal. This value will also tell the device to save the value the INS algorithm reads to the SD card. Then, 'Record' can be 'continuous' or 'trigger' meaning we either only want to record data when it is large enough to motion trigger the device, or we want it to be recorded continuously. The 'Sample Rate' value tells the IMU how often to read samples, which is limited by the performance values of the IMU itself. Such as if the IMU can only read 20 samples per second, then the value cannot be set above that number.

This is the configuration message that the GUI will utilize when communicating with the NAT Device and its firmware. The GUI will store a temporary text file when the communication/connection is still active, but it will be discarded once the GUI is closed. The

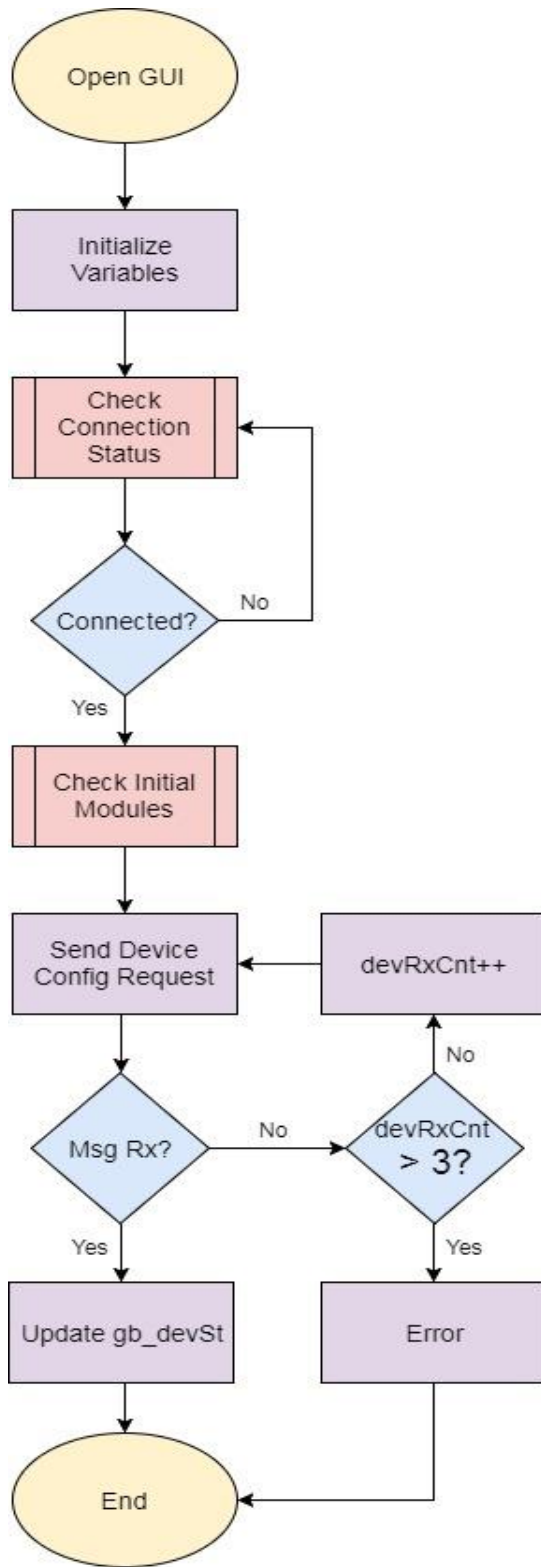


Figure 77 - GUI Connected

not so much a function that will make its way to a distributable Configuration GUI). After a proper

device however, will always have a version of the configuration settings stored in its memory, as it needs these values to configure the device and to communicate with the GUI. It will only send the GUI a copy of this file, and overwrite its file upon receipt of the new configuration file.

The Code

This section will go through the flow charts for some use cases that the GUI will perform during its usage. Each flow chart will be discussed as to when it would be run, and a description describing in more detail what the flow chart illustrates. These flow charts are very important to the computer engineers when developing code, as it allows the engineers to see what the code flow should accomplish before any line of code is written. This also allows the engineers to easily discuss and develop a section of code together, and then easily distribute sections to be written by the individuals.

Initially, the flow chart for Figure 77 will be discussed, as it contains the code that will be run upon the initial opening and running of the Configuration GUI form. The code starts with the opening of the GUI. After it is opened, the GUI will initialize any variables that it may need during its running. This includes variables that are required in other functionality, but also variables such as devRxCnt which is a variable that holds how many times the GUI requests the configuration message from the device to avoid falling into an infinite loop when the code should error and break out. After all of the initial variables are initialize, the code will check to see if the device that we are using is properly connected. If it is not properly connected, the status group box remains colored red (shown in Figures 73, 74, and 75), and it continues waiting for a proper connection to be made. This is what the decision diamond and the 'no' path are indicating. This could end up being an infinite loop, if the connection to the device is never made. However, the engineers decided this is fine, since the point of the Configuration GUI is to connect to the device, configure its various values, and then see some data output (for the benefit of the engineers,

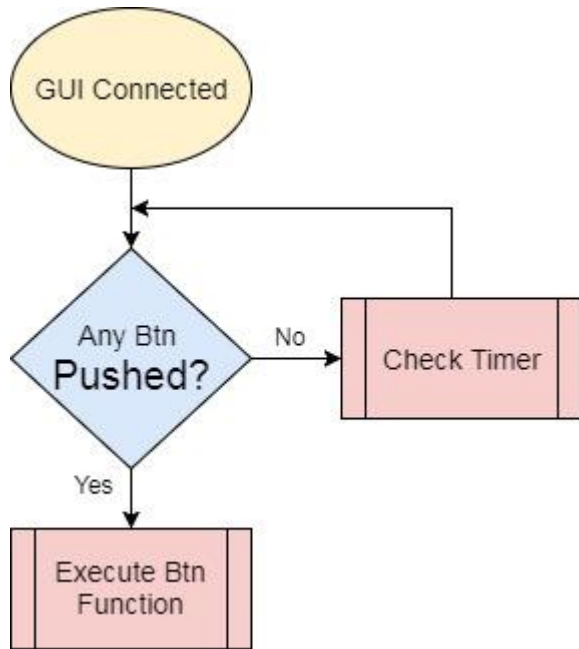


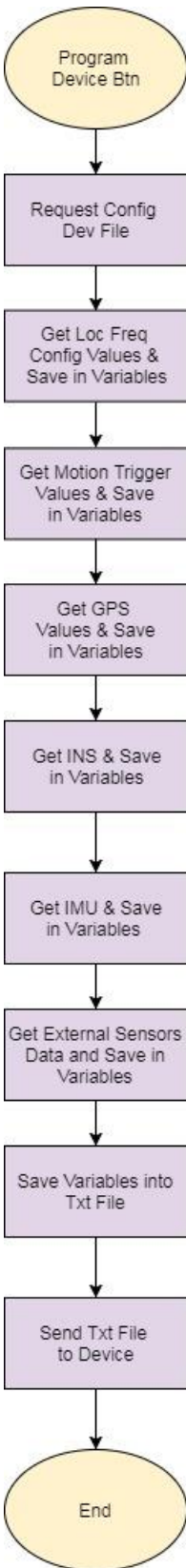
Figure 78 - General Use

from the device, update the objects in the group box ‘Settings in Device’, shown in Figures 73, 74, and 75 above in section ‘Layout and Usage’. Once the group box has been updated, the device ends its opening sequence and move into its general use code flow, shown in Figure 78.

The next flow chart that will be detailed is shown in Figure 78, the general use code flow chart. This is what generally, the GUI is doing while it is running. This is the code that the Windows form will continuously run while the Configuration GUI is open. While the GUI is connected and not in another code flow, the code will wait for a button to be pushed. If the button has not been pushed, it will check to see if the timer has run out and it needs to check to make sure the device is still connected. If the device loses connection, the focus of the code will move from this general use back into the start-up code, waiting for a connection to be made again. Therefore, in the ‘Check Timer’ function, the code will break out of this flow, and move focus back to the flow shown in Figure 77. However, as long as the device remains connected the configuration GUI will continue waiting for a button to be pushed, and occasionally checking to make sure the device is still connected properly. However, once a button is pushed it moves into the appropriate sequence for the button, then moves back to waiting for a button to be pushed after that sequence has been completed.

Moving forward into the sequence for button pushes, the first button that will be discussed is the ‘Program Device Button’. This button will save any settings that the user includes in the Configuration GUI to the nat.conf file that was discussed in an earlier part of this section. It will then send those settings to the NAT device, and update its configuration settings accordingly. This is how the code will go about completing that overall task. This code flow will begin once the button has been pushed. This will break the code from the previous code flow, Figure 78 General Use, and move it into Figure 79 Program Device Button Pressed. The device will request a fresh device Configuration File from the device, so as when it makes changes to the file, it is making

connection is made, the code will move on to checking the initial modules and their connection status’. Then the Configuration GUI will make a request to the NAT device to receive what its current configuration values are. This message will look like Figure 76, described fully in the section above ‘Communication with the NAT Device’. The GUI will wait to see if the message has returned after some appropriate amount of time. If the message has not arrived, then the GUI will check to see if the devRxCnt is above 3, meaning the message has been sent an appropriate amount of time, and if so, error and leave the GUI, closing it and indicating to the user the device has not been connected properly/there is an error communicating with the NAT Device. However, if the message still has more tried to get devRxCnt past 3, then it will resend the message and try again to get a response. If the message comes through, the device will parse the information from the message, getting the current configuration settings



them to the newest version of the file possible. This is necessary because the device is going to save over its old file when it receives a new text file, and parse all of the data in the file as new data. Therefore, if it were to have taken its oldest file and just added in or changed the data that the user changed, and kept the outdated settings from before the NAT device would assume that the outdated data is actually new settings that the user wishes to implement. This would cause errors, so we just take the newest file when we are going to send the device new settings. Once the GUI has the newest file, it will begin to go through the sections of the GUI, locating where values have changed or been added, and then either changing settings that are already in the text file, or adding new values to the file in their appropriate section. The GUI will begin by looking at the location frequency values, which are stored in the ‘Interrupt’ section of the nat.conf file. It will update appropriately, as the ‘Interrupt’ section is a bit more than just directly dropping values into the section. There is an algorithm of sorts, described in the section previous (Communication with NAT Device), and the GUI code will handle settings these values appropriately. The next section that the device will update is the Motion Trigger values. Looking at the appropriate group box in the Configuration GUI’s layout, the code will cycle through the values and update the file if the value the file currently holds is different to the one the user entered. Then the flowchart will move similarly through the GPS, the INS, the IMU, and any external sensors that have been additionally added to the NAT device to increase its functionality. The code will do a similar function for all of these sections as it previously did. It will cycle through the sections and check to see if the values stored in the objects are different from the values read in the file. Whenever they are different, they will update said file. The file, once fully updated, will send the file to the NAT device. The NAT device itself will then parse the files values, update the values where appropriate in the microcontroller (where the firmware will exist and be run from), as well as all of the attached components. Once that is done the NAT device will save the newest version of the file to the memory. The reason why the newest nat.conf file will not be immediately saved to the memory, overriding the last version immediately, is because the engineers want the device to have a chance to flag to the user that the settings are not valid, or that there is an error, without invalidating the main copy of the settings. Should the newest file from the Configuration GUI have been immediately saved and overridden the older version, a version that was known to work, and then did not get implemented because there was some error updating the values, there is no longer an old copy that is known to have been valid, updated the values properly, and then been saved and made the new golden copy. The reason that this file would cause errors are significant. While the GUI does its best to force the user to give values that the components accept, there is always a chance that the programmer missed some setting in a component that doesn’t allow some setting to be implemented. Such as maybe the user wants the GPS polled every 10 minutes, but also the IMU setting at 10 samples/second does not allow the microcontroller to perform both functions at once. And the microcontroller notices this and flags the new configuration settings as faulty. The

Figure 79 - Program Device Button Pressed

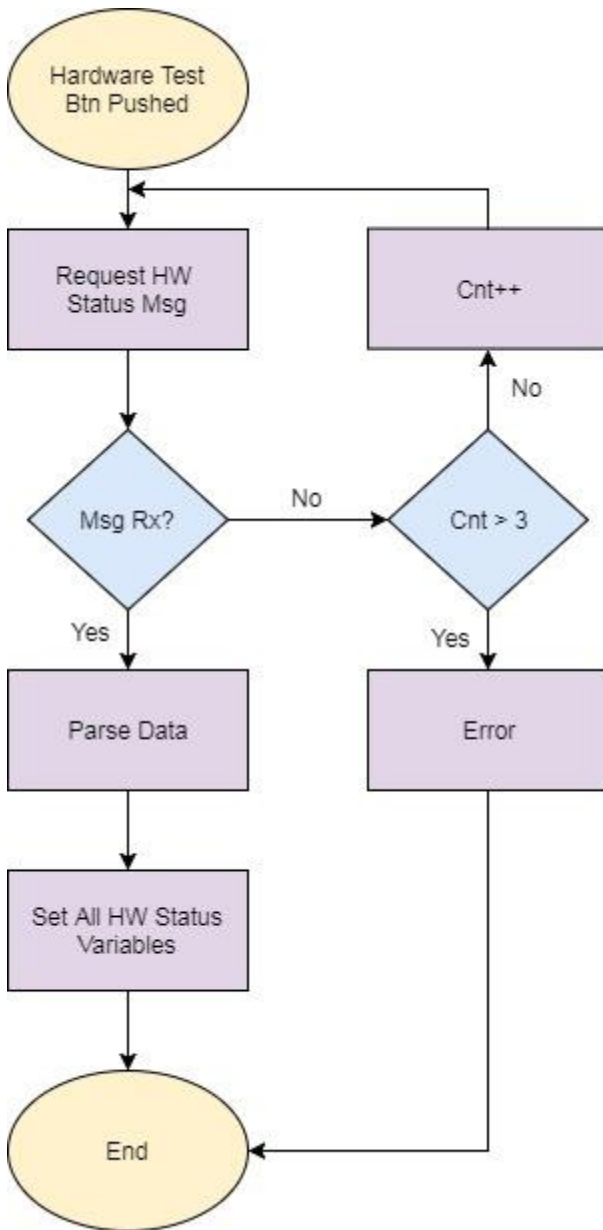


Figure 80 - Hardware Status Button Pushed

message have been lost or dropped, the Configuration GUI will check the counter for the message sending, making sure the device has not sent it more than 3 times. This is to ensure that the Configuration GUI does not enter an infinite loop waiting for the device to receive a message. Should the device not receive a message and had already sent the message 3 times, the code will instead break from the code, throwing an error to the software before doing so. The user will be notified accordingly of the error as well. If the code doesn't go all the way to erroring, and instead at some point a message is received the Configuration GUI will parse the data, set all of the

microcontroller could revert back to the older configuration settings, knowing these have been proven to work.

The next flow chart worth discussing is the 'Hardware Test Button Pushed' flowchart, shown in Figure 80. The hardware test button is pushed when the user wants to see what hardware components are live and connected, but to constantly be polling this data would require information constantly, and would likely not change very frequently. Therefore, to save some bandwidth for when the user needs it, and to reduce the GUI getting bogged down and overloaded in hardware connection check requests and information, the engineers have decided to utilize a button to request this information only when the user wants or needs it. Upon pushing this button, as described in 'Layout and Usage', the user will see the various components connection boxes (the red and yellow squares on the GUI tabs) update accordingly. If there was no change, the device will not give an indication of such during this point in the design. This is how the code will accomplish this. The device will begin with a request to the device to see what the status of the various components in the hardware currently is. As with all communication, the design has to be made to handle lost or dropped packages and frames, so the code will make sure the message has been received after an appropriate amount of time waiting for it. Should the message be proven to have been received by the configuration GUI then the code will proceed forward, however should the

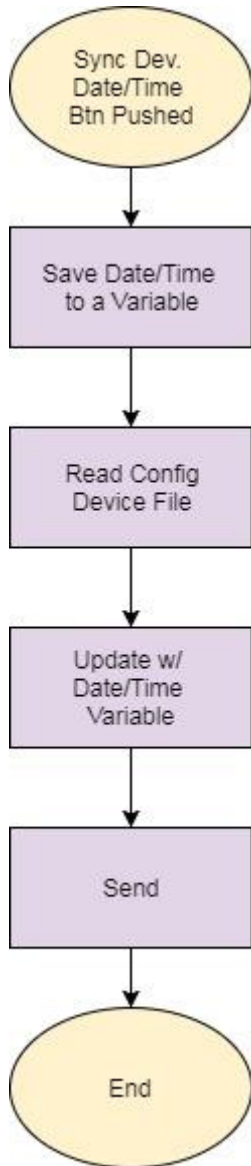


Figure 81 - Sync Device Data and Time Button Pushed

hardware status variables and update the Configuration GUI objects, and then end the code and return to the General Use flow chart in Figure 78.

There are many buttons on the Configuration GUI, as it is a piece of software full of important functionality. One of the seemingly less important buttons is the Sync Data and Time button. This button will essentially sync up the GUI's data and time (based on the host computer that it is running on) with the NAT devices. This will ensure that when the user asks for a location frequency time, or a sleep time that any discrepancies between the Configuration GUI and the NAT device will be handled. This means that if the Configuration GUI is reading the current time as 8:31:52 and the NAT device (via the GPS UTC time in its data, and the clock inside the device) is reading the current time as 8:32:25, this small bit of difference is handled so when the user says they want the location fix at 8:30 daily, the user is actually getting what they want. This flow chart in Figure 81 depicts the Sync Device Date and Time Button Pushed code behind the button. The code will begin when the button is actually pushed, moving the focus of the GUI out of the General Use flow shown in Figure 78, and instead into this code. The first thing the GUI will do is read the Date and Time from the computer's clock and save it to a variable. Then the device will read the configuration file and update the date and time variable within the file. The GUI will then move forward and send the file to the NAT device. While the GUI code will not handle this part, the device will then parse the file within the device, update the date and time variable offset based on the value determined by the Configuration GUI. This offset will be used to determine what needs to be done to the date and time to match the values that the host computer is reading, and thus the values that the user is using to determine what location frequency to set, to the values that the NAT device is actually using so as to ensure a coherent time between the two entities.

The last main button in the Configuration GUI is the 'Retrieve' button. The 'Retrieve' button is used to update the 'Settings in Device' section at the bottom of all of the tabs. This section will hold all of the setting values that the device actually had at the last time the 'Retrieve' button was hit, or if it was never hit then the values the device held upon opening of the GUI. The flow for this button is shown in Figure 82, Retrieve Button Pushed. The pushing of the button, like all of the buttons, moves the focus of the GUI away from the General Use code (shown in Figure 78), and into the flow chart shown in Figure 82. The code begins with a request for the nat.conf file from the NAT device. It implements the same code as before to avoid the function from getting stuck in an infinite loop, but still allowing it to make multiple tries to get a valid message. Once the message has been received the message is parsed and all of the corresponding items in the Configuration GUI are updated with the new information. This is a fairly simple code, however the importance is great. It shows the user that the values that they have set in the GUI intending to update those values then in the NAT device have been either updated properly or that something malfunctioned and the values they wanted to set were not properly set in the device. However, the functionality on how to get the device to update its values within the device and update the nat.conf values is the requirement of the NAT

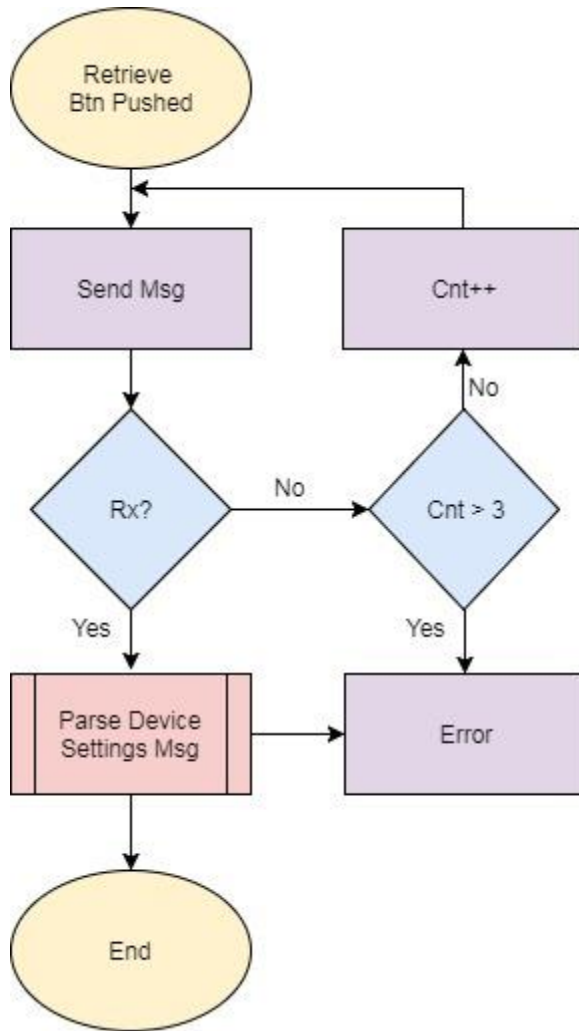


Figure 82 - Retrieve Button Pushed

Use flow chart (Figure 78) is where the Configuration GUI focus will sit while waiting for interrupts to happen and move the focus elsewhere.

The next SD card button is the ‘Save All’ button, the flow chart shown in Figure 83, in the center. Essentially what this button hopes to do is using the values that the user has entered into the ‘Save to:’ textbox and the ‘as:’ textbox to copy all of the contents from the SD card and save it to the location in ‘Save to’ in a folder called the value entered into the ‘as’ textbox. This will allow the contents of the SD card to remain untouched by the user directly, but still let the user see and work with the data that has been saved into the SD card. The user cannot be blocked out of the device completely, as to the engineers’ current knowledge, but keeping it as far away for allowing the user to manipulate it directly is probably for the best.

The final SD card button is the ‘Clear’ button, the flow chart shown in Figure 83, on the left. This button will clear all of the data that it can from the SD card that is currently stored on the device. This may be so that the user can use a device they had already previously used again, since the NAT device is not a disposable device. It is an intelligent piece of hardware, meant to be reused.

device itself.

The final set of Configuration GUI flow charts are the one involving the SD card. There are 3 buttons that are associated with this, the view contents, the save all, and finally the clear button. Each of this will be discussed briefly.

The first of the buttons is the ‘View Contents’ button, the flow chart shown in Figure 83, on the right. The device, when connected to a host computer should be able to be viewed as an HID device, which means that it will show up as something like an E: drive, similar to when a USB flash drive is connected. When an HID device is connected and it works similar to a USB the user can open the SD card on the NAT device and view its contents. Therefore, the view contents should be as easy as opening a file explorer (since the Configuration GUI is a Windows form) and moving the directory path to the HID NAT Device. However, the engineers don’t want the user to be able to manipulate the files from there, thus the Configuration GUI will copy the file contents (essentially just the names and any other major data) and show it to the user in a pop-up window. The code will then wait for the user to close the window before returning control to the General Use case flow chart. Everything returns to the General Use case flow chart after its done with its flow, since the General

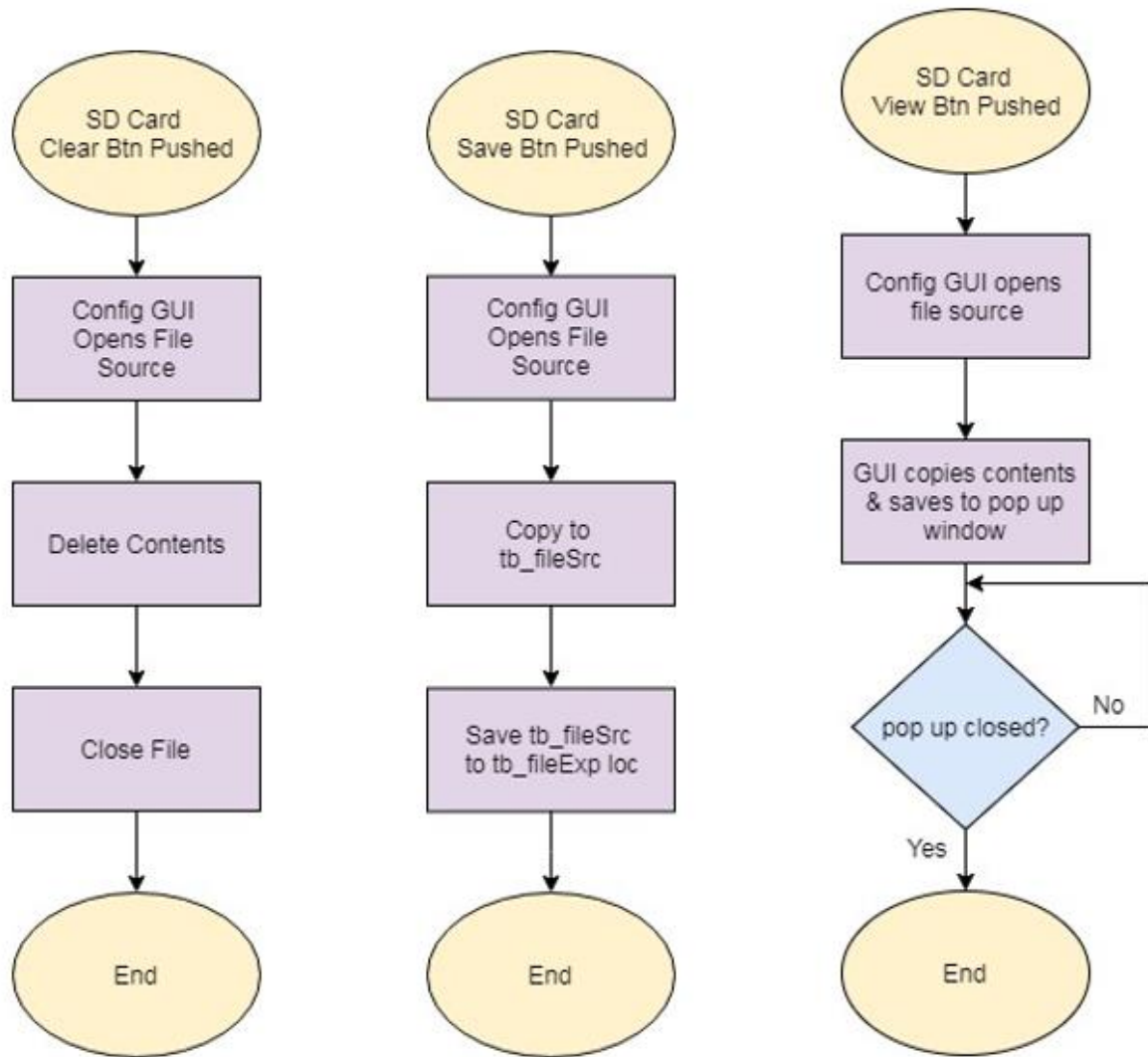


Figure 83 - SD Card Flowcharts (Left) Clear, (Center) Save All, (Right) View Contents

5.5.3 UART

The UART module will handle the communication between the MCU and the GPS unit, and is one of three potential modules that may be used for the RPMA and cellular communication modules.

The input that the UART module will receive is for transmission string that originates in the MCU via nonblocking function call and any byte received via the UART interrupt handler.

The output from the UART module mirrors the inputs, one byte at a time from the MCU to the transmission buffer and data from the receive buffer to the MCU.

Helper modules include the board initialization module, which will set parameters such as baud rate, stop bit, and parity, and the wakeup interrupt handler, which will coordinate the restart all communication. The UART module itself will serve as a helper module to provide a simple interface between the MCU and the device so that they communicate with each other.

5.5.4 MPU

The MPU module provides a software interface for the MCU to control the operation of the onboard Inertial Measurement Unit (MPU-9250) and obtain its readings. Specific memory addresses and control constants are defined as macros in its header file to enhance readability of the calling code.

The main program sends an MPU I²C address, a memory register address, a pointer to data, and a data length. If the memory register is for addressed for a write operation, the pointer is dereferenced for the given length and taken as further input. If the address is a read operation, the input is taken from the I²C bus after the necessary delay.

Output from the MPU module begins by writing the MPU-9250 address to the I²C bus. When control returns to the MPU module the memory register address is then written to the bus. Write operations then write the given data, while read operations await a response to store the returned values in the given pointer location.

The MPU module depends heavily on the I²C module, which was provided by the microcontroller vendor, Microchip. This module organizes read and write requests in a queue and reports to the calling function the current status of each transaction. In monitoring the status, the MPU module is prevented from overloading the I²C bus, which operates at 400 kHz.

5.5.5 Telecommunications

Each of the two wireless telecommunications peripherals (RPMA and Cellular) will be managed through highly similar software modules, to present the same interface to the MCU. Two separate modules are planned, to handle any implementation differences and so that the calling code explicitly invokes each module as configured.

The input to the telecommunications modules will be the software-defined server IP address and the payload information of NAT device identifier, GPS location data, and timestamp for storage in the remote server. Other payloads from additional sensing equipment may be included to meet the optional specification in later designs.

The output from the RPMA and Cellular modules will be positive acknowledgements, timeouts, or error status after communication is attempted. The main controller unit will take this output to the diagnostic log.

The SPI helper module provided by Microchip will be used for communication between the MCU and the telecommunications peripherals. Unlike the I²C module described above, the SPI module does not use queues and instead uses a blocking data exchange system, where data is read from a buffer, transmitted over SPI, then the buffer is overwritten with the response.

5.5.6 Web GUI Model-View-Controller

The customer-facing web application will consist of three classes of components according to the Model-View-Controller paradigm. The homepage on the end-user's browser will request the customer data view from the main controller component, which will obtain the data from the appropriate model component, and format the results using the correct view module, which is then presented to the user. Thus, all three classes of components are tightly interconnected in operation.

The models contain the core business logic, and are accessed indirectly by the users. Their main inputs are all of the possible commands from the controllers and the unformatted database query results from the database-management system. The models' outputs are the precise database queries sent to the back-end, including both update and data access requests, and the formatted database results toward the front-end back through the controller.

The view modules will contain the user-visible interface definitions: what data to display, how to display it, and what interactive elements to provide. The user login view will take in customer credentials from the browser and pass them to the controller for authentication. The device tracking views will take in map images, device location tracks, and uploaded sensor data from the controller and output the results as a web page for the browser. Operational controls on the browser will be sent to the controller and the results received using Asynchronous Javascript (AJAX), allowing the user to remain on the same page as information is refreshed.

5.5.7 IMU

The IMU allows a GPS receiver to work when GPS signals are not accessible. It is an electronic device that measures a body's exact force, angular rate and occasionally the magnetic field surrounding the body. It uses a combination of accelerometers, gyroscopes and magnetometers. The NAT device will be programmed to record the IMU data either continuously or when motion triggered. The input and output will be to the microcontroller because that is what it communicates with. The IMU does not depend on any other module or function in order to do its job.

5.5.8 HID / USB

HID has built in software and therefore there is no programming to be done.

5.5.9 INS Skeleton

The INS portion of this device is going to be executed in a future project because it is not in the scope of the current project at hand. The design team is going to implement a skeleton where the INS code will go in the program during its future development. It will take values from the SD and the IMU.

5.6 Flow Charts

The main software that the NAT will run will be the Power On/Reset module. This is shown to the left in Figure 84. Upon power on the module will initialize the variables that the device needs throughout operation. Then it moves down and tests every module so to show that the system is

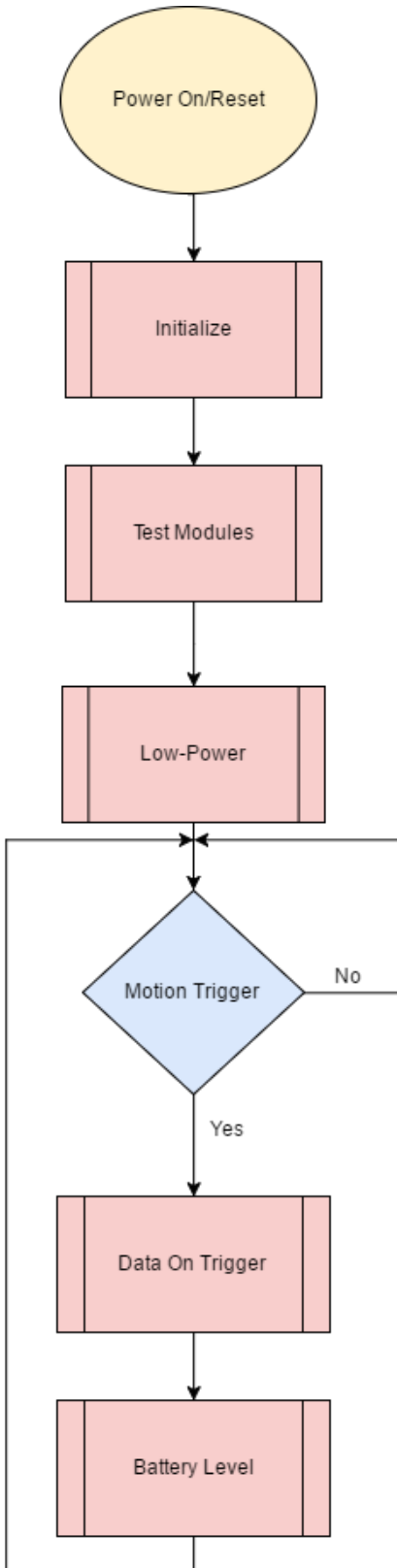


Figure 84 - Main Device Software Flow Chart

both chips, which included operational example code from other projects that provided much guidance for the software prototypes so far developed.

completely up and running, no broken or faulty modules. The system then moves into low power mode, conserving as much energy as it can as to prolong the battery life as long as possible. The only thing it does while in this stage is look for a motion trigger. The motion trigger is given by the IMU module. Until it recognizes this trigger, the device sits and waits in low power mode. However, upon the trigger it moves into the function Data on Trigger. This function will utilize the other modules (mainly the GPS, flash memory, and the radio modules if the GPS is working and has a signal otherwise it will also utilize the IMU and the microcontroller to calculate the current location) to output to the user the important data (such as the devices location and power status). While in the function it will continue to loop and send data until a timeout shows the object has stopped moving. Then, since the trigger has stopped, the focus returns to the loop shown in Figure 84, leaving the data on trigger function block. The device then checks the battery level to determine if it should move to an emergency battery conservation effort. If not, it returns to the infinite motion trigger check loop and waits for another trigger.

This is the main functionality. The device will output the its location after a motion trigger wakens it from low power mode, then output that location as well as other user designated relevant data (such as level of battery) over the radio modules continuously until it stops moving or the battery dies. However, decisions like whether to use GPS or INS, or transmit data through RPMA or Cellular are determined elsewhere.

5.7 Summary of Design

Overall, primary design decisions were made in order to produce a system that provides the desired asset tracking functionality with the minimum level of implementation uncertainty. As a required part of their curriculum, the team has programmed microcontroller units to communicate over a serial UART. This is reflected in the use of a microcontroller-based design and the selection of the OriginGPS Nano Hornet module and the XBee cellular modem, both of which communicate with the microcontroller using UART. Along the same lines, the choice of the particular PIC24FJ128GA006 microcontroller and MPU-9250 IMU reduced uncertainty through the sponsor's familiarity with

To reduce uncertainty in the software modules development, design wizards and existing example code are being used and extended wherever feasible. Current feature prototypes for the NAT device firmware are currently using library code provided by the Microchip MPLAB Code Configurator wizard, which generated initialization routines for each of the UART modules, timers, and I2C and SPI interfaces, and APIs well documented in included header files. Similarly, the forms in the Configuration GUI were generated by the Visual Studio form editor, allowing form elements to be dragged and dropped into position and have their properties set from a programmer-friendly set of on-screen menus.

While the available libraries provide interfacing code for each of the software modules, the endpoint software must still be developed by the team. A complete functional breakdown of one software module per major component in the NAT device firmware and a layered approach for the both the configuration and customer GUIs allow the modules to be developed with high cohesion and minimal coupling, with detailed documentation such as execution flowcharts produced for every module under development.

The supporting circuitry for the major components on the printed circuit board was also designed using a principle of simplicity, as requiring fewer components reduces the complexity of hardware troubleshooting issues. A voltage regulator and voltage reference provide the needed stability for microcontroller operation, a high frequency and low frequency oscillator drive the execution and timer clocks, and p-channel MOSFETS allow for external switching directly from active-low logical outputs from the microcontroller.

6.0 Project Prototype Construction and Coding

The team needs to utilize the research, the standards and constraints, as well as the design to create a prototype of the device and the code. The sections below will aid in creating said prototype. First, the electrical engineers will design the schematic for our device in the OrCAD software. From there, they will research potential Printed Circuit Board Technology and then detail how the component pieces will be stuffed into the device. From the computer engineers, a prototype of the GUI and the Firmware technology will be created, and then a final coding plan will be advised.

6.1 Integrated Schematics

6.1.1 OrCAD

OrCAD is a software suite used primarily for electronic design automation (EDA). This is the software used often in industry, and therefore not only beneficial for the design and production of the NAT device, but also to get experience with an industry used software. Within this software, the engineer can capture a schematic, the NAT device schematic shown in Figure 85 below, as well as allow immediate creation of a build of materials. The team utilized OrCAD for the electrical design of our NAT device.

6.1.2 The Schematic

Below in Figure 85 is the integrated schematic of our project which fully describes the architecture of our device and is the realization of the block diagrams shown above. This schematic was made using Cadence OrCad circuit capture and once all devices are described along with their package footprints it can be exported to Cadence's PCB design software.

The schematic was drawn using Cadence's OrCad schematic capture tool which came with all of the passive components shown: the resistors, capacitors and inductors, as well as some of the active components like the PMOS's. However, when it came to the IC's we had to model them ourselves and comb over datasheets to ensure that all of the pin numbering was correct and that external components were connected properly.

The same goes for device footprints, many of the common ones came pre made such as the 0805X footprint used for the resistors and capacitors and the SOT-23-X footprints used for the MOSFETs and the voltage regulators. However, some footprints had to be custom made for our purposes such as the QFN-54 footprint for the PIC microcontroller that includes a reference to one extra pin that represents the thermal pad.

A schematic is not meant to represent the actual scale of components or their actual placement on the eventual electrical device. A schematic is meant to show what components are used in a device and the connections needed to facilitate the operation of the device, but above all a schematic is meant to be readable. Much of the complexities of a full, to scale, circuit diagram are removed so that the details of the circuit can be easily analyzed and appreciated.

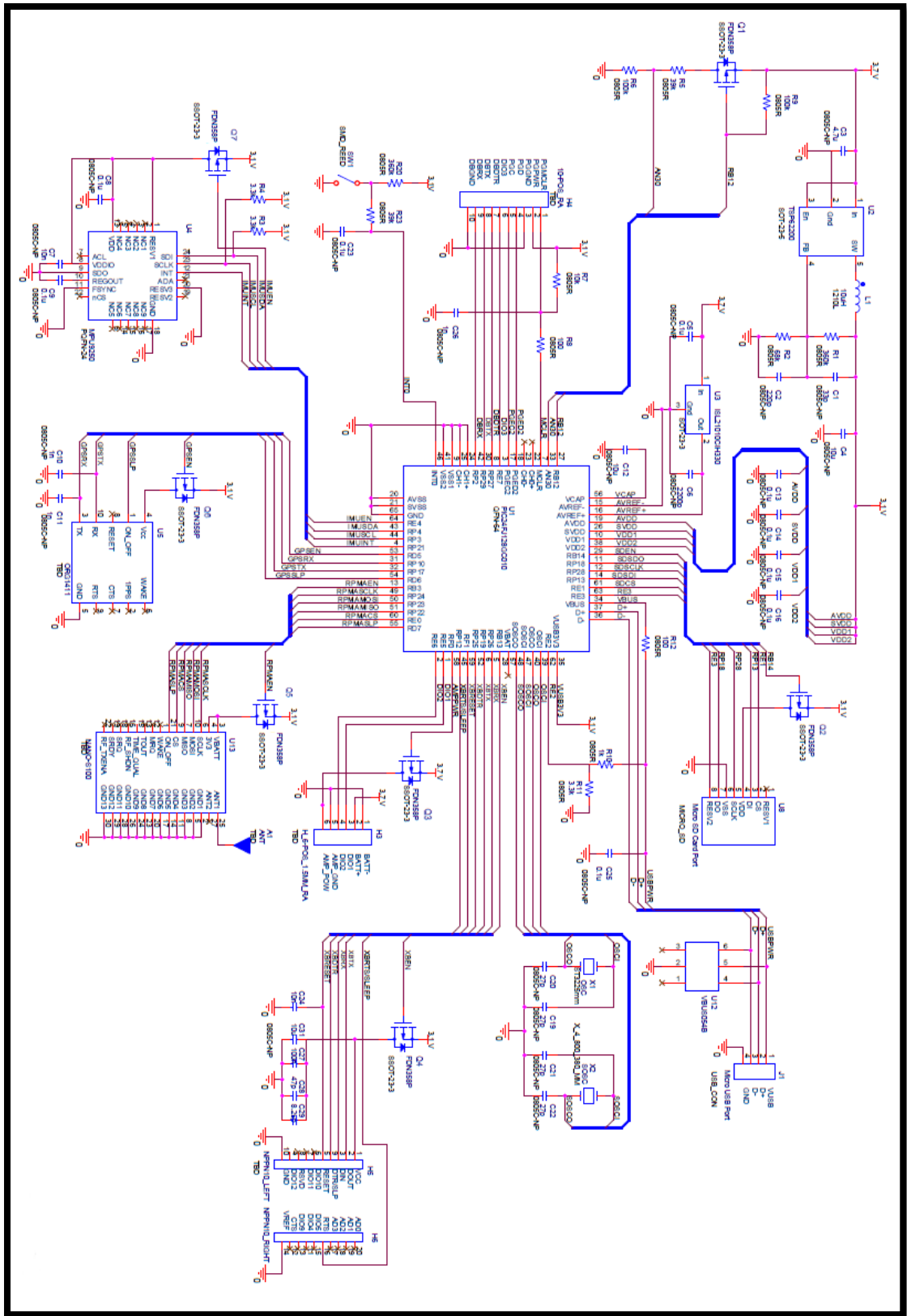


Figure 85 - Integrated Schematic-Semester 1

Make note of the pin numbering on the PIC microcontroller, notice that the pin numbers are out of order. The pin numbers on the actual PIC device start at 1 on the top of the left-hand side (pin 27 in the figure above) and increase going counter clock-wise around the package. In our schematic, the pin positions have been modified so that the connections can be easily managed and neatly organized to help with analysis.

Update: The only design changes that the NAT device had was the removal of RPMA and the addition of a charging circuit, both of which can be seen in the updated schematic, Figure 86.

As discussed in section 3.3.7, the RPMA module was removed from the design due to issues within the company that was mainly responsible for deploying RPMA technology, Ingenu. RPMA, as previously discussed, was a great choice for communication within our design. It was low power, and was free to use because it operated on the unlicensed frequency band. However, when the engineers worked with Ingenu to order a module to integrate, as well as some base stations to test with, they were told that the company was changing its direction from building out its network. The sponsor advised us that this was what happens when a new company/technology is failing. Therefore, we decided to remove the module entirely, and rely solely on the LTE module. This allowed for the pins that were configured for RPMA in the previous schematic to be allocated to other modules, which helped with flow control.

The other major change was the addition of a charging circuit. A charging circuit is necessary on devices such as ours, even if the NAT device is meant to use so little power that it can stay in the field for months to years on a single battery charge. However, in the current design the LTE module draws a significant amount of power when in use. Therefore, for use in the current design as well as for market use, a charging circuit was integrated. When the device is plugged in via the USB connection, a red LED will turn on indicating that the device has been connected and is charging. There is another LED to indicate that the battery is fully charged, which is a green LED.

A smaller change that can be seen from the schematics is that there was an addition of a TLV70018DDCR 1.8V linear regulator above the GPS. This was added because it was found during testing of the first version of the PCB that one version of the OriginGPS NanoHornet needed to operate at 1.8V. Therefore, it was added so that the proper voltage was sent to the GPS unit.

Other, smaller additions that can be immediately seen are the addition of many test points. These test points are necessary to allow easy testing of certain lines on the PCB. Therefore, there are many test points throughout the schematic.

The final schematic for this scope of the NAT device is shown in Figure 86. It has all of the additions that are discussed above, as well as all of the major components that were in the original schematic, shown in Figure 86. This schematic uses almost every possible pin that the PIC allows. These pins are configured to the settings that are shown in the schematic through the IDE that is provided with the microcontroller. These pins will be set in the pin manager file that is included in the firmware. Once the firmware is loaded, the configuration of the pins is as well, so the microcontroller knows what to set each pin to do, as they are very configurable.

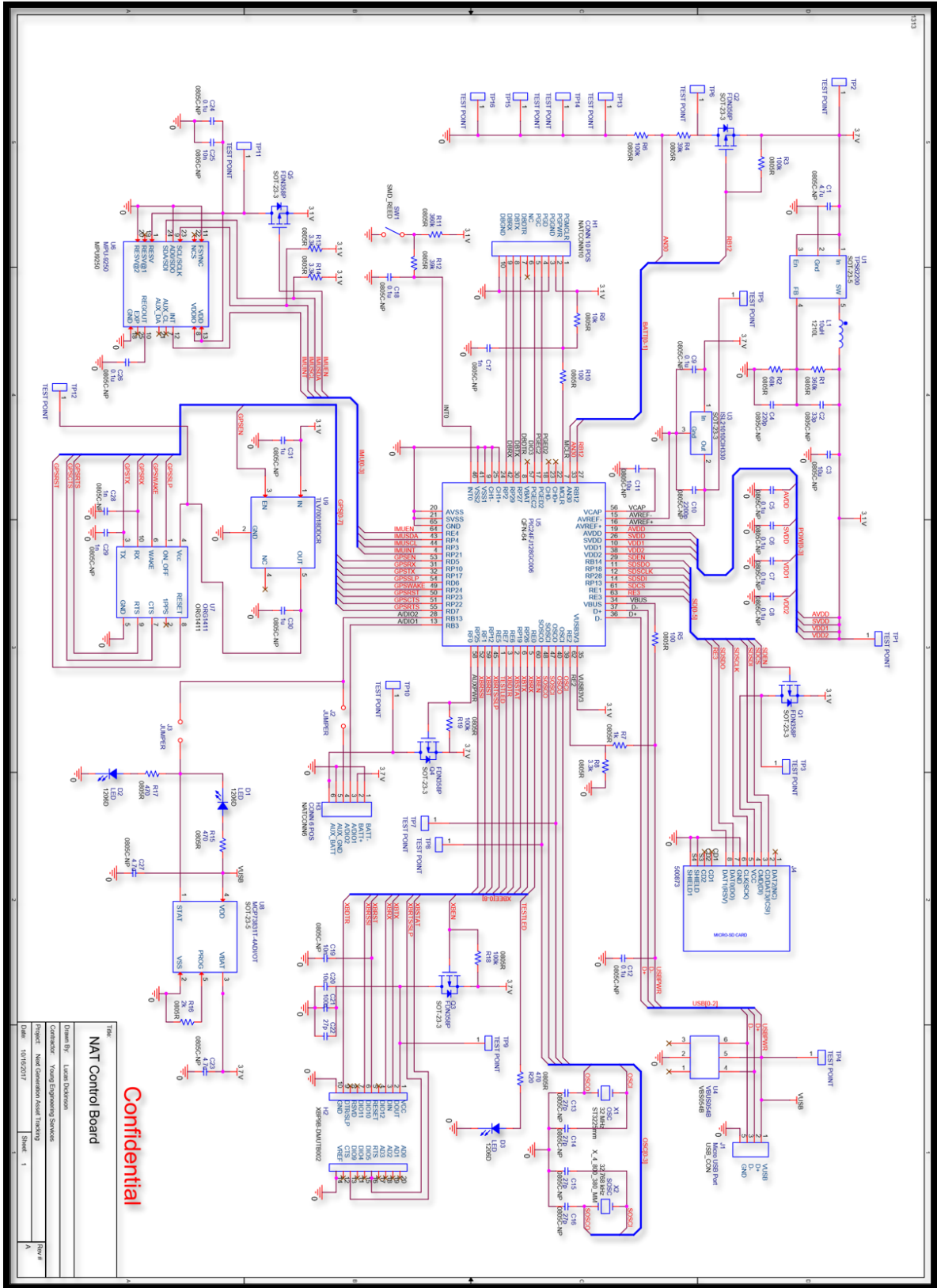


Figure 86 - Final Schematic

6.2 PCB Vendor and Assembly

Once the board is laid out in the PCB design software and the physical constraints of the desired physical board are met it can be exported as a Gerber file ".gbr" and sent to a PCB manufacture and they will send back the routed PCB board. However, the job of placing the components on the board, or stuffing it, either lies with the project engineers or a pick and place machine. Once components get small enough a human can no longer manually solder them to the PCB so a pick and place machine must be used to stuff the PCB board. For this purpose, we have chosen to use component sizes large enough to solder manually as we cannot justify spending the large amount of money required to program and operate a pick and place machine.

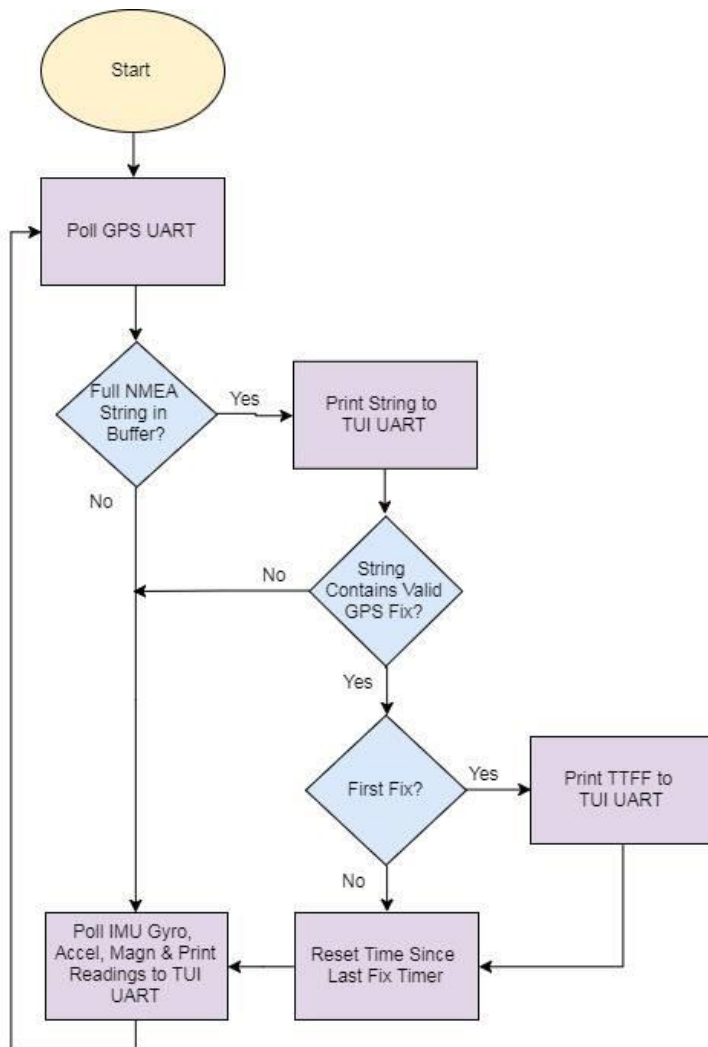


Figure 87 - Software Prototype Flowchart

Association (NMEA) specification. The UART code maintains its own GPS string buffer using a state machine, flushing invalid strings and writing full strings to the workstation UART upon completion and validation. Header file definitions specify what rows of the serial terminal to use for NMEA strings, and a helper function rotates between the available rows, using ANSI line erase

The team's current design requires a 4-layer PCB. Research into the possible PCB vendors brought the team to decide on using BasicPCB.com. They guarantee the best deal on PCB printing, and will aid in keeping the teams budget. Therefore, when the times come to print the PCB, the team will utilize this service.

6.3 Software Prototype

Initial prototyping involved a PIC24FJ128GC006 development board connected to the GPS development board through a UART serial connection and to the IMU using an I2C interface. A rudimentary test program was developed, which cycles between servicing the UART receive buffer and polling the IMU, sending results to a test workstation using a VT100-compatible text-mode interface on a second UART connection. A execution flowchart of this process is given in Figure 87 and a screenshot of the output in Figure 88.

The GPS communicates to the PIC24 by sending strings formatted according to the National Marine Electronics

commands to wrap the display area at the last line. Additionally, the strings are parsed to determine if they contain a valid GPS fix, and if so, the top GPS line is updated with the latest fix data (UTC time, longitude, and latitude). The time to the first such valid fix is shown on a dedicated console line, measured using a 32.768 kHz oscillator, whose 1-second interrupt updates the timer value between hardware power-on and the first fix.

The IMU provides nine readable registers of interest, namely the x, y, and z values for each of the gyroscope, accelerometer, and magnetometer readings. When polling the IMU, the software prototype first reads the configuration register to determine the current ranging factor for the sensors, which are then displayed above each set of readings. This is done using I²C read commands, which append a 1 as the new least significant bit to the 7-bit I²C address. The data is transferred one byte at a time, but the readings are given in a 16-bit range, so first the high half register is read, its contents bitwise shifted to the left, then added to the low half register reading. The raw data is displayed as a string, but to aid visualization, a 53-column bar is displayed, with pipe characters at the far left, center, and far right, allowing a range of 25 columns in the positive and negative direction. Block characters are then displayed in the appropriate direction representing one 25th of the extreme range of the sensor reading

```
PIC24 UART Demo
Last updated 7\23\2017
22:54:50 UTC 28°35.682' N 81°23.132' W
TIME TO FIRST FIX 0:52
TIME SINCE LAST FIX 0:06
$GPGGA,, , , , 0,00, , , M,0.0,M, , 0000*50
$GPGGA,225417.882,2835.6859,N,08123.1323,W,1,03,19.5,31.0,M,-31.0,M, , 0000*6D
$GPGGA,225439.882,2835.6825,N,08123.1325,W,1,03,20.7,31.0,M,-31.0,M, , 0000*64
$GPGGA,225450.882,2835.6820,N,08123.1326,W,1,03,21.3,31.0,M,-31.0,M, , 0000*68

0 0
GYROSCOPE +/- 250 dps
X: -215 -2 dps |
Y: 16868 129 dps |
Z: -2020 -15 dps |
ACCELEROMETER +/- 2.0g
X: -14264 -0.871 g |
Y: 7472 0.456 g |
Z: -2576 -0.157 g |
MAGNETOMETER +/- 1200 µT
X: 12812 469 µT |
Y: -1710 -63 µT |
Z: -962 -35 µT |
I2C1_MESSAGE_COMPLETE
```

Figure 88 - Software Prototype Terminal Output

6.4 Final Coding Plan

All project coding activities, namely the full development of NAT device firmware, Configuration GUI, and Customer Web Application sub-projects, will be the joint responsibility of each

Computer Engineer in the product group. The team will coordinate through weekly face-to-face design meetings at the laboratory site with all agreed specifications saved to a shared folder on Dropbox according to the documentation standards given in section 4.1.1.4.

Although every team member is responsible for every line of code in the project, primary responsibility for each will be assigned as follows: Mr. Baird will coordinate the development for each software module in the firmware, Ms. Fry will head the Configuration GUI in addition to her general project management duties, and Ms. Thomason will manage the web application. During the intersession between semesters, each sub-project leader will decompose the development activities based on the descriptions and flowcharts of each module into a work breakdown structure. The team will then collectively assign weekly due dates and effort approximations to each item over the execution phase of the project, which will be added as an additional level of detail to the Gantt chart in Figure 100 in section 8.1.

To facilitate collaboration between team members, all code must be checked into a Git version control repository nightly. Version control provides a timestamped record of coding efforts and allows modules to be reverted to a good branching point if more efficient solutions present themselves or a particular approach needs to be abandoned. Git is a distributed version control system that provides operational atomicity, efficient performance, and SHA-1 security. Although Git is a distributed system, Somasundaram portrays it as a hybrid system, with centralized elements such as a main repository server used to host the authoritative master branch [UU]. One such server will be provisioned for the team during the intersession.

Integrated development environments will be used to boost productivity through features such as syntax highlighting, autocompletion, style linting, VCS integration, and built-in debugging tools. Syntax errors will be caught as they are made, eliminating the more tedious aspects of software debugging, and the integration of the entire toolchain will allow code to be built and deployed with a single click. The choice of IDE is the community standard for each development ecosystem, minimizing the number of external libraries that will have to be tracked down and acquired. The NAT device firmware will be coded using the free MPLAB X suite provided by Microchip Technology, providing many peripheral libraries for the chosen PIC24 MCU. The Configuration GUI is specified to be a .NET Windows Desktop application, which selects any recent version of Visual Studio, including several editions available from Microsoft free of charge. Finally, the Customer Tracking GUI will be coded using the PHP application plugin for NetBeans, freely available from Oracle.

Updated:

The MCU software will be a structured program, with execution parameters defined in a configuration file on the mass storage device. In operation, a loop will execute, reading the values from the GPS and IMU, logging the sensor data to the mass storage device and transmitting the data over the cellular links, logging success or failure. Mature peripheral drivers provided by the MCU vendor will be used wherever possible to reduce the likelihood of faults in the bulk of the code. The operational loop will be controlled by the power-saving profile defined in the configuration file, either waking up based on a timing interval, or via motion trigger.

The NAT device firmware components are defined as a sequence of modules, each servicing a separate peripheral according to its specific interface. The USB library being used requires a cooperative multitasking environment, and so each module will execute under the expectation that it will cede control to the next module when it has nothing to do at the present time in an infinite loop, broken only by transitions to low power mode and returned to by configured timer or peripheral interrupts. Care will be taken in implantation that no long-term blocking code is used, deferring to the next loop as often as necessary for the smooth operation of all firmware modules.

The main NAT device firmware runs when power is first applied and whenever the device is awoken by motion or real-time-clock interrupts. The microcontroller hardware is initialized, with each peripheral power enable pin configured for output and asserted. Each piece of hardware is queried, and any errors or failed responses are indicated using up to three independent debugging indicators: lines appended to the debug log on the SD card, descriptive text transmitted over the debugging UART, and predetermined blink sequences on an LED. The device configuration is then read from the SD card and applied. After configuration, data is read from the GPS and IMU and logged to the SD card. The current firmware can only obtain position data from the GPS, but later expansion is expected to use an inertial navigation system(INS)to deduce position from IMU data. Once the position is known, a packet of data is sent to the cellular data module, which then transmits the device ID and position to a centralized server. Once this is completed, the firmware switches to low-power mode, de-asserting the external peripheral power pins. If voltage is sensed on the USB power line at any time, the device switches to USB Mass Storage Device(MSD)operation, allowing a new configuration file to be written, and device logs read on the configuration workstation. Normal operation is then resumed once the USB power line is disconnected.

7.0 Project Prototype Testing Plan

Once the components have been acquired, they will go through breadboard testing. Then, once the device has been printed and stuffed, the overall prototype will then be tested as a whole. The software test plan will also be created to test the software, once it's been written.

7.1 Hardware Test Environment

The Lab

Working with a sponsor to design a project may have some perks depending on what resources are available to you by the sponsor. For our project, the majority of our testing and implementing is done in a fully functional lab provided to us by our sponsor. The facility is equipped with various test equipment including digital multimeter, soldering station, microscopes, and power supplies. This can be advantageous in the case where the university laboratory equipment is all being occupied. Further discussion into this lab can be found in section 8.3 The Winter Park Laboratory.

7.1.1 GPS Module

For the GPS used in our project, we acquired a development board that we use to help us in our testing and development of our final project. Along with the GPS development board, there was documentation and schematics that was used in the development of our schematic and also in the testing of the GPS with our microcontroller. Performing accurate tests on the GPS module required the creation of a special cable and connector created by our Electrical Engineering team. Since our software and hardware testing is done indoors, which will diminish GPS signal strength, we had to create a cable long enough to extend from our test bench to a window that the GPS could hang out of to receive better signal. Connectors were made to easily connect and disconnect this cable from our microcontroller board and the GPS development board. This setup can be seen in Figures 55 and 57 of section 3.5 and also appendix testing image 5.

7.1.2 Microcontroller Development board

The microcontroller development board includes the MPU, the DC-DC converter, and the voltage reference generator. Testing on the microcontroller and the MPU will be done with the components on the development board. However, the DC-DC converter and the reference generator will be tested individually after they are ordered and received. A test procedure will be designed for these two components that will be discussed in section 7.2. This development board and test setup can be seen in Figure 55 and 57 of section 3.5.

7.1.3 Wireless Communication Modules

The development kit from Ingenu will be a good enough test setup for our group to test the functionality of the RPMA module. This will be a similar case for the Digi XBee LTE CAT-1 cellular module. Testing these modules on their respective development boards is the most efficient and easiest way for us because these modules are surface mount components. This means their pins will be hard to access for testing unless they are soldered onto board with pins that break out from the board. The specific testing of these modules will also be covered in section 7.2. The

Digi XBee LTE CAT-1 Module testing setup can be seen in Figure 55, in section 3.5 Parts Selection Summary.

The LTE Software: XBTU

XBTU is the software made by Digi to configure and use their radio modules. The software will connect the module via a UART connection with the development/experimenter's board, and allow the user to configure its settings, such as seeing the devices phone number (as it is connected to Verizon's network), seeing its IP number, setting the phone number to send SMS messages to, and then telling the device what messages to send that number in addition to a myriad of other settings. The software allows the users to use a GUI to configure these settings or to use a serial communication port to write the settings manually.

7.2 Hardware Specific Testing

When going through the design process of a project, time must be set aside for testing in each phase. This may seem obvious, but component and module testing should be done in the early stages of the design to ensure proper functionality of each component, and to potentially ease troubleshooting process should there be an issue later in the design. Individual component testing can also be useful to verify results of specific test parameters that may be critical to a design. For instance, the datasheet for a piece of hardware may suggest that it will only draw a few micro amperes of current in a certain low power state. If your design is based on low power consumption, this data is worth the time of designing a test plan for verification. At the very least, it is good practice to test each component or module to validate proper operation before combining everything into a single project to find out you have a defective part.

7.2.1 GPS

As technology progresses, there becomes an increasing reliance on GPS technology. With this in mind, it is important for designers to understand its limitations and how to test them. Along with general hardware operating parameters like current draw in different operating modes, there are a series of fundamental performance parameters that apply to these systems that will be discussed and tested.

7.2.1.1 Cold Start

A cold start occurs when the GPS has to track its location beginning with no information about its position, time, or the ephemeris data of satellites. The case of a cold start generally includes initial power on. With the help of a software test program that records the TTFF, we were able to perform a Cold Start test. On initial power on of the device, we recorded a cold start time of 1 minute and 50 seconds. This is quite off from the datasheet and could be a result of our position of the GPS.

7.2.1.2 Warm Start

A warm start takes place when the user initializes the GPS with valid position and time data or if there is continuous RTC operation with an accurate last known position stored in RAM. Although position and time is known, TTFF is usually longer for this state because there is no satellite ephemeris data due to its expiration.

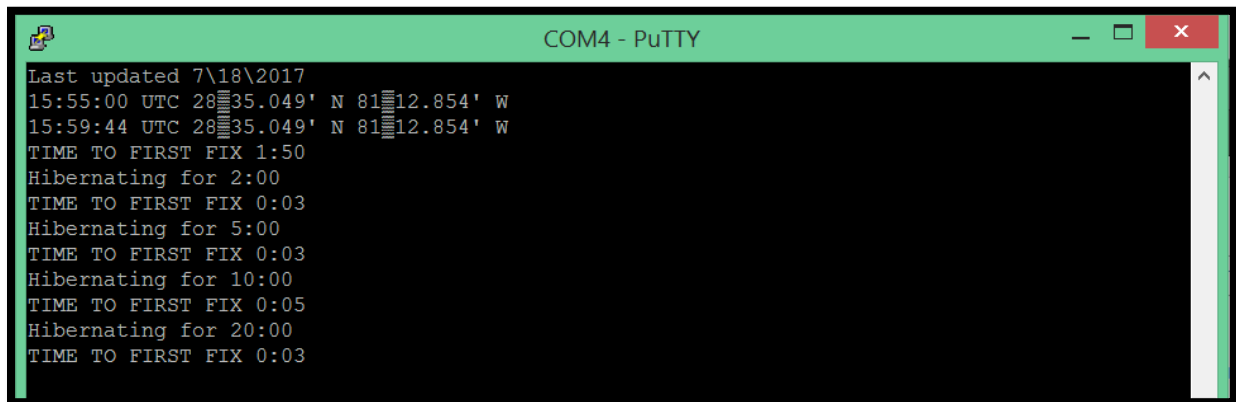
7.2.1.3 Hot Start

The first performance parameter we will discuss is the Hot Start time. This parameter is a result of a software reset after a period of continuous navigation, or on a return from a short idle period that came after a period of continuous navigation. During Hot Start all critical data (position, velocity, time, and satellite ephemeris) is valid to the specified accuracy and available in RAM [T].

7.2.1.4 Aided Start

The TTFF can effectively be reduced by the application of this method. The difference between Aided Start and Warm Start is that in Aided Start state the GPS is being supplied with valid satellite ephemeris data.

Figures 89, 90, and 91 below show actual test results displayed in the putty serial communication software and, in a table that compares our results with results from the datasheet of the GPS module.



```
COM4 - PuTTY
Last updated 7\18\2017
15:55:00 UTC 28 35.049' N 81 12.854' W
15:59:44 UTC 28 35.049' N 81 12.854' W
TIME TO FIRST FIX 1:50
Hibernating for 2:00
TIME TO FIRST FIX 0:03
Hibernating for 5:00
TIME TO FIRST FIX 0:03
Hibernating for 10:00
TIME TO FIRST FIX 0:05
Hibernating for 20:00
TIME TO FIRST FIX 0:03
```

Figure 89 - GPS Acquisition times test.

Operation	Datasheet Value	Test Results	Unit
Hot Start	<1	3	S
Signal Reacquisition	<1	n/a	s
Aided Start	<10	n/a	s
Warm Start	<32	n/a	s
Cold Start	<35	110	s

Figure 90 - GPS Acquisition Times.

7.2.1.5 Power Supply current

When taking any current measurements, care must be taken to protect the device and the person taking the measurement especially in high power situations. Since our project operates with low power we do not have to worry too much but will still take safety precautions just in case. Even though our project will be low power, we must still take necessary safety measures to prevent any damage to our components. Current measurements taken on the GPS was done in two different

states; Tracking and Hibernation. After using software to drive the module into each state, we measured the current draw of the GPS module. In Figure 91 we can see that when the GPS module is actively tracking its location, the power supply current can be anywhere from 6 to 40 mA. Putting a small resistor in series with the power supply and measuring the voltage across it we calculate the power supply current (using Ohm's Law) to be 39 mA during tracking. We can see that this value is within the range specified by the datasheet. Putting the GPS into a Hibernation state should reduce the power consumption. The 20 μ A power supply current that was measured shows this reduction in power and is also within the specified range.

Parameter	State	Min	Type	Max	Measured	Unit
Power supply current	Acquisition	40	43	47	46	mA
	Hibernation	16	18	22	20	μ A
	Tracking	6		40	39	mA

Figure 91 - GPS power supply currents.

7.2.2 Microcontroller

Our project will use a 16-bit PIC24 microcontroller. It is a general-purpose microcontroller with an operating voltage range of 2.0V to 3.6V. According to the datasheet, it can sink a maximum of 300 mA and source a maximum current of 250 mA.

For the microcontroller, our main concern was that the device is operational and is able to execute the commands given to it. With the help of our microcontroller development board, we were able to test the microcontroller designing a test program that returns positional data using the provided MPU. We also interfaced the GPS module to the PIC24 and integrated it into our test program so that it could return data about its location to the microcontroller. We verified the results gathered from the communication with the PIC24 and the GPS by looking at our exact coordinates on google maps.

7.2.3 MPU

Our software prototype was used to test the MPU for functionality. Our main concern for this module is that we can get valid data that appropriately changes based on the movement of the device. With our software prototype, we were able to verify that the MPU is working, and reacted by providing the right data each time we moved it. The results of our test can be seen in Figure 88 of section 6.3. In this figure we can clearly see the gyroscope, accelerometer, and the magnetometer are all providing useful data on each axis.

7.2.4 RPMA

The RPMA NANO-S10 Evaluation Board will be the main platform for the testing of our RPMA module. We will verify that we can communicate over the network using the evaluation board. To

do this, we can send small amounts of test data to the RPMA base station to see if the module is working properly. We will also test the dc characteristics of the device to see how much our results deviate from the results of the manufacture.

Current Consumption

To be considered a LPWA device, one of the criteria is that the device must consume very low power. When the RPMA module is in a deep sleep state, it will wake up at given different intervals to try and keep an accurate signal. This periodic wake and brief waking up from the deep sleep state is called the oscillator calibration state, and is done to ensure quicker synchronization to the network. The current consumed during these periods of calibration is between 1.4 mA and 1.6 mA and can be seen in Figure 92. Looking at the figure we notice how short these periods are, so even though they consume a lot of current, it's for a very small amount of time which compensates for the high current draw.

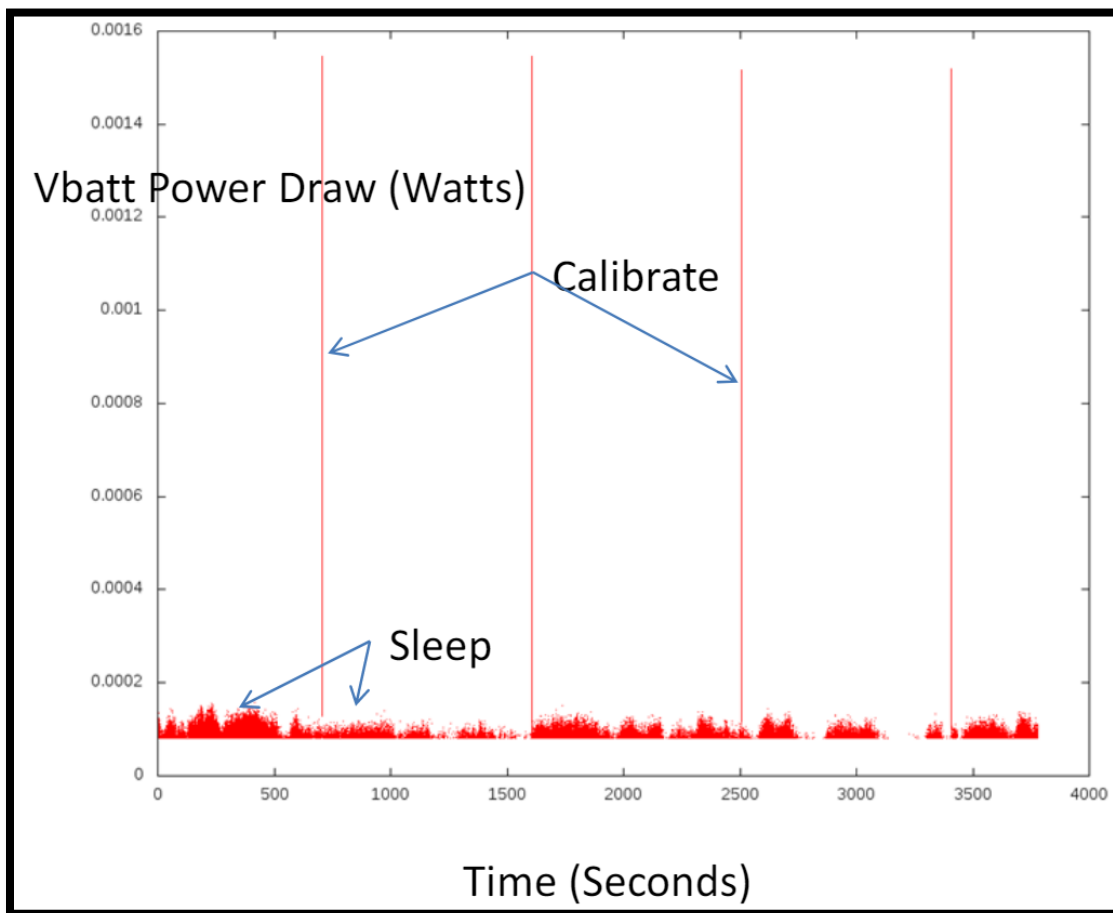


Figure 92 - RPMA Oscillator Calibration State

*Permission Requested [P11]

During operation, we know that the RPMA will consume different amounts of current depending on what operation it is carrying out. Figure 93 below displays these different current consumptions

in a graphical fashion. We can see right away that the most power-hungry task is transmitting data over the network. The transmit power in the graph represents about 23.3 dBm, which is the absolute maximum according to the datasheet. The transmit power actually depends on the RSSI received. When the device is receiving data, it is a little less taxing on the power consumed. A quick evaluation of this graph shows that by minimizing the amount of times the device transmit and receives data, we can reduce the amount of power consumed and get the battery life that the manufacturers claim the device is capable of.

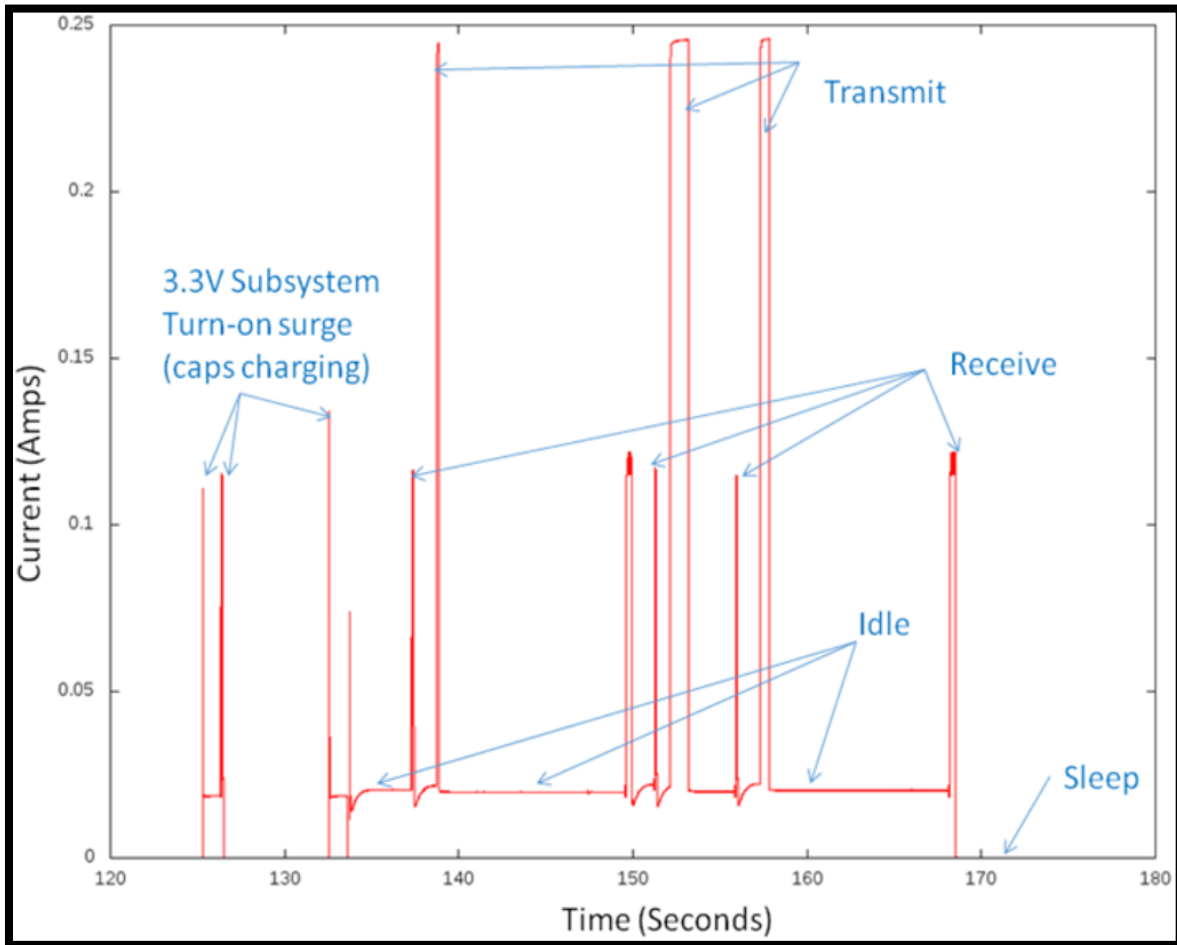


Figure 93 - RPMA current consumption during various states.
*Permission Requested [P11]

7.2.5 LTE Module

The XBee cellular LTE Cat 1 embedded modem was tested on the evaluation board from Digi International.

Configuring the LTE Module

The team utilized the XBTU software (described in Section 7.1.3) to configure the LTE module, shown in Figure 94 below.

From the figure, it can be seen that the device was tested to be connected to the network, as it shows that the network it is connected to is Verizon and it has a valid phone number. Other various values that were verified in this GUI output is that it has a solid signal strength (6B is a hexadecimal value for 107dB) well within the values the module specify, that the device is configured to send SMS messages to a set phone number (which is proven to work further on in this section). The configuration settings that are not also included in this image, but in a lower section of the settings you can also configure the device to move into sleep mode, where it drains significantly less power, or airplane mode where it should also drain less power than when it is in full power mode. These values are also tested and shown below in Figure 97.

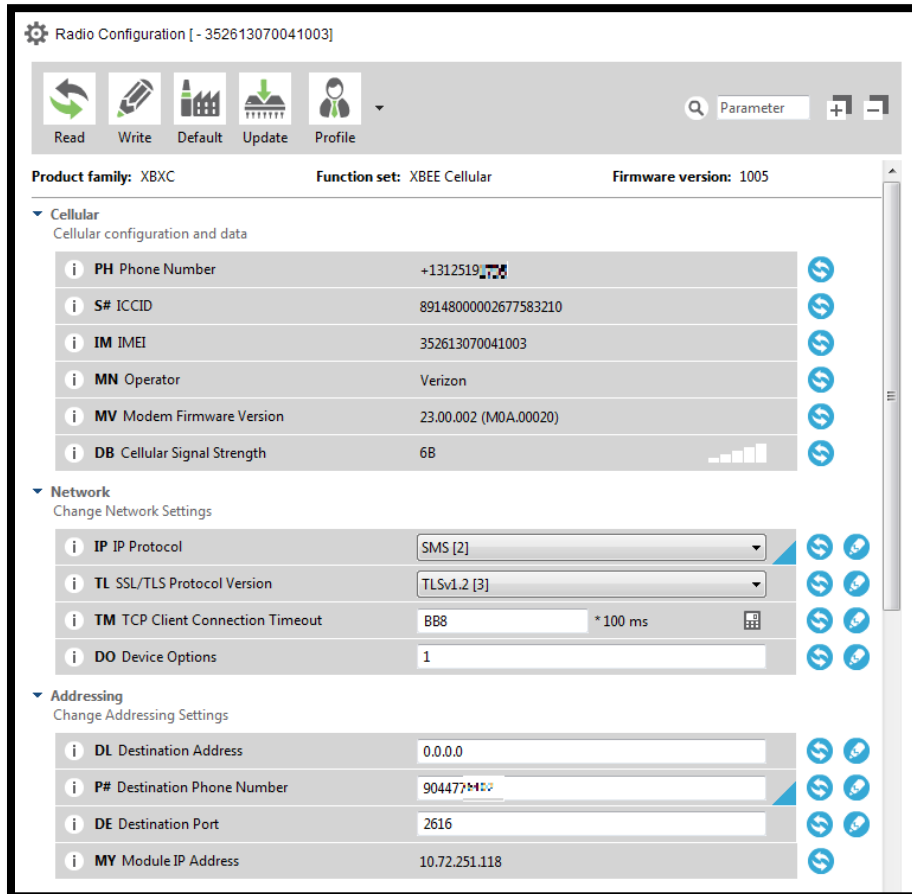


Figure 94 - XBTU Configuration Settings for LTE Module

SMS Messaging

The first specification of the LTE module that was tested was its ability to send SMS messages, as it should be connected to Verizon's cellular network. This is a very important feature of the LTE CAT-1 device, and it could even be argued may be the most important one of our specific application. Utilizing both the XCTU software and PuTTY's serial COM port, the device was configured and told to send the engineers text messages to their phones. The device was able to both send and receive messages to and from other cellular devices on various cellular carriers, with minimum to no effect of the success of the message. This output is shown below in Figure 97.

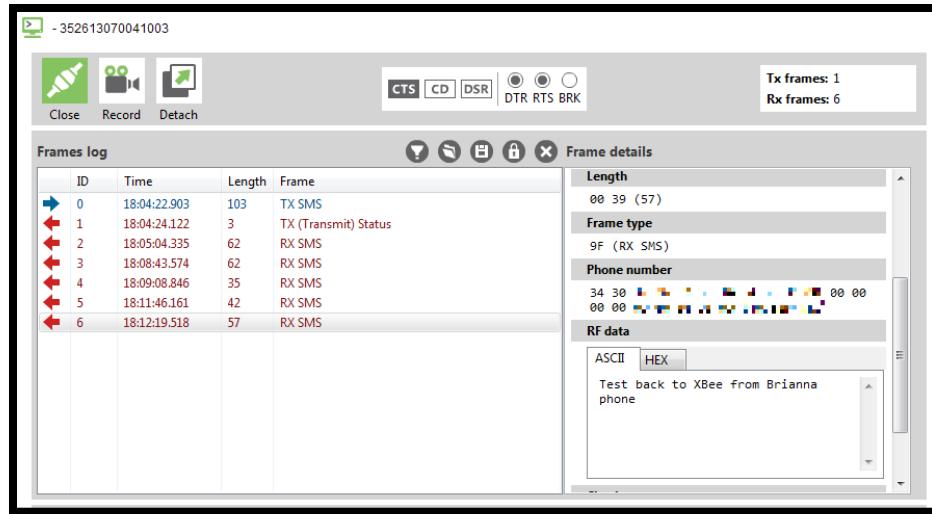


Figure 95 - XBTU Sending SMS Messages

Figure 95 shows the output that XCTU gives the user. What can be seen from this output is that in the first message under Frames log (where the arrow is pointing to the right of the screen), there was a message sent from the LTE module to one of the engineers' phones to prove that it could send SMS messages that the engineers write and tell it to send. The second message is the response that the cellular device that the message was sent to receive the message, which is explicitly stated in the Frame details as a successful transmission (the frame TX (Transmit) Status). The other messages are all messages that the LTE module has received (where the arrow is pointing to the left of the screen). Any of these messages can be selected and the details seen on the righthand side of the screen under Frame details. The last message has been selected to open some of its information. This message was sent to the LTE Module from one of the engineer's phone (Brianna as stated by the message content shown under RF data). Other information that may be of use when receiving and sending messages is also shown in the Frame details section.

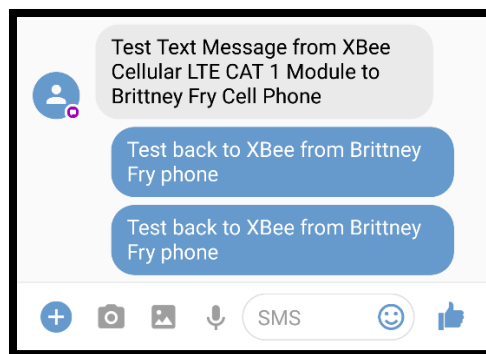


Figure 96 - SMS Messages from LTE Module on Engineer's Phone

The Figure 96 above shows that the user actually does receive the messages. However, during testing the Engineers found that unless the user was on the Verizon network as well, these messages came in as fully hexadecimal messages. Moving forward with designing the software

that handles sending and receiving messages with this LTE module, we will have to build in an algorithm to return these messages back into their regular text format.

Current Consumption

The team also tested the device in various settings, specifically the power modes that the device specifies will save us power. The values that we read are shown below in Figure 97.

Parameter	Power Mode	Voltage Current Read at	Typical	Measured	Unit
Power supply current	Idle	3.3V	860	90	mA
	Deep Sleep		143	6.2	μA
	Airplane		-	90	mA

Figure 97 - LTE Module Power Supply Current Measurements

The chart above in Figure 97 shows that the device draws very little current, even in its full idle mode. This is very good for our application as we need the device's battery to last for a very long time, hopefully meeting our requirement that the battery will last longer than 3 months, as specified in Section 2.5 the Quality of House.

7.2.6 DC-DC Converter

DC-DC converters are devices used to convert from one voltage level to another. Functionality tests can determine if the device has the performance characteristics that the manufacturer specifies in the datasheet. Some of these tests will be discussed below.

Output Line Regulation

The output line regulation is a measure how well the DC-DC converter maintains its output voltage while the input voltage varies from its minimum to maximum operating voltage. The equation below is used to calculate the percent of deviation from the nominal output voltage as the input voltage changes. The TPS6220 can keep its output within 0.26%/V of the nominal value which is pretty accurate.

$$R_o = \frac{|V_{omax} - V_{omin}|}{V_{onom}} \times 100\%$$

- V_{omax} = Output voltage at maximum input voltage
- V_{omin} = Output voltage at minimum input voltage
- V_{onom} = Nominal output voltage

Output Load Regulation

The output load regulation is similar to the output line regulation. Both specifications tell how much the output deviates from its nominal value when there is a change to the circuit. Load regulation refers to how much the output voltage deviates from its nominal value specified in the

datasheet, while the load is varied from minimum to maximum. The equation below gives this value as a percentage. The load regulation for the TPS6220 is specified to be 0.0014 %/mA.

$$L_R = \frac{V_{oio} - V_{oim}}{V_{onom}} \times 100\%$$

- V_{oio} = Output voltage at maximum output current
- V_{oim} = Output voltage at minimum output current
- V_{onom} = Nominal output voltage

Efficiency

Efficiency determines the internal power dissipation and how efficiently the input power is transferred to the output power. The equation that specifies how efficient the device is as a percentage is given by:

$$Ep = \frac{|V_{out} - I_{out}|}{V_{in} \times I_{in}} \times 100\%$$

- V_{out} and I_{out} = Output voltage and current of converter
- V_{in} and I_{in} = Input voltage and current of converter

The test also has the capability of capturing the efficiency at different power levels and can be plotted into a chart to illustrate the efficiency versus the output current. For our device, a maximum of 95% efficiency can be achieved depending on the load current.

7.2.7 Voltage Reference Generator

Testing for our voltage reference generator will not be as extensive as other modules. For our purpose, making sure the device generates the desired output voltage is the main functionality of interest. Of course, this output voltage is constant only for a given range of input voltages, which according to the datasheet is 3.2V to 5.5V. Staying in this range on the input should ensure a constant 3.0V at the output. Another important thing to note about this device is that the output voltage is relatively independent of the load resistance on the output. Therefore, the output should stay constant even when the load is changing.

To perform the proper tests on the device, we will use a variable DC power supply to apply power to the input of the reference generator and vary the voltage within the specified operating range. This will verify that the device operates within the range defined by the manufacturer. Starting within the input voltage range, we can slowly decrease until the output starts to drop off. Figure 98 shows the dropout curve for our device.

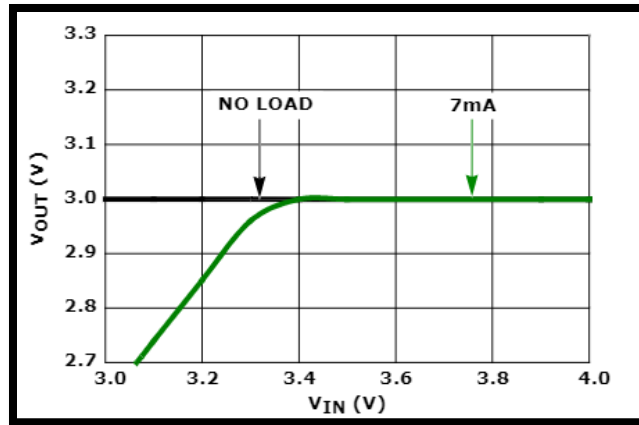


Figure 98 - Voltage Reference Generator Dropout voltage
 *Permission Requested [P12]

7.3 Software Test Environment

Firmware:

Software testing will proceed in three stages, with a different environment for each. First, each module will be tested in isolation on a lab bench using an Integrated Circuit Debugger (ICD). Any problems that arise will be investigated using software breakpoints, single-stepping, and diagnostic messages printed to the screen. Both “tests-to-pass” of core functionality and “tests-to-fail” of exception states that may cause undesired behavior will be conducted. Integration and system testing will follow in the same manner with every combination of modules and full-up testing. The second test environment will see the NAT device operating wirelessly, but locally. Without the ICD, debugging will depend on diagnostic logs written to the external flash storage. The last stage of testing will be field testing, with one engineer operating the NAT in cell phone contact with another engineer in the lab monitoring telemetry through the web application interface.

Configuration GUI:

The .NET configuration GUI will be tested using the debugging features of Visual Studio. Individual front-end modules will be tested with dummy data, to include both valid and invalid inputs. Next, integration testing with a software approximation of the NAT device will be conducted, to ensure the correct configuration file is generated by the system. Finally, the GUI will be tested with a live NAT device prior to its full-up testing phase, with every combination of settings tested both on the GUI and the NAT devices itself.

Database and Customer GUI

The initial development will take place on virtual machines at the developer location. A database server will be implemented first, with dummy data and queries used to verify data is correctly sorted and can be retrieved by the web server. Once that basic functionality is achieved testing will proceed each component of the Model-View-Controller, starting with live View components first with dummy controllers then live controllers and dummy modules, and then finally with live

modules integrated with the database. Completion of this phase will begin the provisioning of a web-facing server to repeat these steps in a staging environment, and then the full system will be tested with a live MVC and real NAT device data.

7.4 Software Specific Testing

Firmware Testing

Four main scenarios will test each of firmware software modules by validation of the correct operation of the hardware modules under its control: Interval Data Capture, the Motion-Triggered Data capture, RPMA transmission, and Cellular transmission tests. All four tests will involve the GPS unit, while the remaining tests verify one each of the Inertial Measurement Unit, RPMA transmission radio, and cellular radio.

Interval Data Capture Test

First, the NAT device will be configured to take position samples every minute and to store the results to external flash memory. The device will then be powered on and the main program will be left to run for ten minutes. The device will then be connected to the configuration workstation and the diagnostic log will be downloaded. The test is successful if the diagnostic log shows ten correct GPS positions have been recorded with the correct timestamps.

Motion-Triggered Data Capture Test

In the second test, the NAT device will be configured to take position samples after the Inertial Measurement raises a motion interrupt and store the results to external flash memory. The device will be powered on then allowed to enter sleep mode. The device will then be relocated ten times, with an increasing number of minutes between motions (1-20) in order to test the wakeup handlers. The current time during each movement will be recorded in paper notebooks for later comparison. The device will then be connected to the configuration workstation as in the Interval Data Capture test and the pass criteria is also the observation of ten correct GPS recordings with the correct timestamps.

RPMA/Cellular Transmission Tests

The third test follows the initial procedure for each of the previous two tests, with the difference that the device be configured to report its location to the remote database using its RPMA radio, with the cellular radio disabled. The Interval Data Capture/RPMA and Motion-Triggered Data Capture tests are both successful if ten correct GPS positions and timestamps are observed in the database for each, and they match the data stored on the external flash memory. The same procedure and criteria is followed for the Cellular Transmission test, except that the NAT device is configured to use the cellular radio rather than the RPMA radio.

Configuration GUI Testing

The two main functionalities of the configuration GUI are programming the NAT tracker with its operational parameters, and displaying diagnostic data at both a high level and raw diagnostic

level. The operational parameters can be tested in isolation by manual inspection of the output then in conjunction with a live NAT unit, while high-level testing depends on the cooperation of self-test routines built into the NAT firmware and raw diagnostics depend on the implementation of a debug event log on the external flash storage.

Configuration File Generation Test

The isolated test case will consist of using the GUI to select several wide combinations of configurations, so that every possible enumerated setting and a representative distribution of range-based settings are covered. The configuration file, which is downloaded to the NAT tracker in normal operation, will be intercepted in software and the settings will be inspected using a hex editing tool and cross-checked with the configuration file specification.

Configuration File Download and Integration Tests

Once the configuration files are known to conform to format expected by the NAT device firmware, both the ability of the GUI to download the configuration file to the NAT device and then the expected behavior of the NAT device under that configuration will need to be tested. This creates an early integration test requirement between the firmware and the configuration GUI. The GUI will pass the unit test if inspection shows the configuration file in place in the correct format on the NAT flash storage device, and integration passes if the diagnostic file on the NAT device records the intended operational parameters are indeed set as configured.

NAT Self-Test Reporting Tests

The second functionality of the configuration GUI, modules related to testing and debugging the NAT device will also be tested at the integration level. The GUI will need to indicate if any of the four hardware modules are in a failure state, and so self-test testing will involve fault-seeded versions of the NAT firmware, each disabling one combination of the GPS unit, IMU, and each of the radio modules. The GUI will pass this test if the appropriate peripherals are listed in a failure state on the workstation for every combination.

Database and Customer GUI Testing

The remote web-server consists of four layers that build on each successive layer: the backend database, and each part of the Model-View-Controller. Individual unit tests will be carried out using behavioral mock-ups of the components not under test, followed by a two integration tests, first one without user authentication, then one with defined user-based data visibility.

Database Storage Test

The back-end storage setup will be tested by provisioning a relational database server accessible from a web-server based database administration tool in a local area network. The web server will be used to transmit an INSERT query to the database, testing the database connection parameters allow a valid connection to the backend server. This test passes if the database administration tool shows the data is correctly returned using a SELECT query.

Customer Application Model-View-Controller Tests

To test each unit of the customer application, three behavioral stand-ins will need to be created. A view stand-in will simply dump all returned data in the format received. The controller stand-in will pass placeholder data between the tested view and model, one of which will also be a stand-in. Finally, the model stand-in will provide the same placeholder data as the controller stand-in, and will simulate updates to the data by responding with operation success without any actual modifications. The unit tests pass if the expected data is output by the web server to the testing browsers using one real component at a time.

Integration testing of the customer Model-View-Controller will proceed incrementally, starting with the View unit test with Controller and Model stand-ins, then the Controller will be replaced with the actual component and the View will be tested again, proceeding to test with the actual Model, and finally the live backend database will be used, with multiple updates made to the data to ensure the live system is tested and not just the placeholder content. The pass criteria are again that the expected data is passed to the web browser View component, and updates from edits to the data propagate correctly.

Multiple-User Test

The final test of the Customer GUI web application will invoke the user authentication faculties of Model-View-Controller, with the test data marked in the database as belonging to a specific user. The test will pass if the View component shows only the data belonging to the authenticated user and does not display any other data and displays no data if no user is authenticated.

Full System Testing

The overall goal of the project is for all the disparate software components to work in conjunction with the full system. Once each sub-module unit test and full-module integration test is complete, all available NAT devices will be configured to report their locations every minute, with remote reporting to the web server. The devices will then be transported to separate locations in the Central Florida vicinity. The test passes if the correct module locations are displayed on the Customer GUI web application in real time for the duration of the test.

Beta Testing

Once the system is verified to work, additional testing efforts will be made to find the limitations of each component to perhaps extend them with revision. This will include testing the effects of low battery, extended range, garbled communication, and long duration operation to document each failure state. As this is a process rather than a goal, no end of test criteria can be specified, but success will be measured based on the density of issues identified and resolved.

8.0 Administrative Content

This administrative content discussed the part of this project that is not directly related to the engineering of the actual device, yet still very important to the overall production and productivity of the group. The following information will detail how the team worked with setting a timeline, overall goals, and setting and following a budget. These are necessary for a project to function well, because the majority of projects fail due to poor time management, poor leadership and overall project management, as well as poor budgeting. When the project doesn't finish by its deadlines, or when it goes way over budget to the point where it is not worth continuing funding its production or the group runs out of money to fund the project, then the project dies.

8.1 Milestone Discussion

Initial Project Milestone for Both Semesters

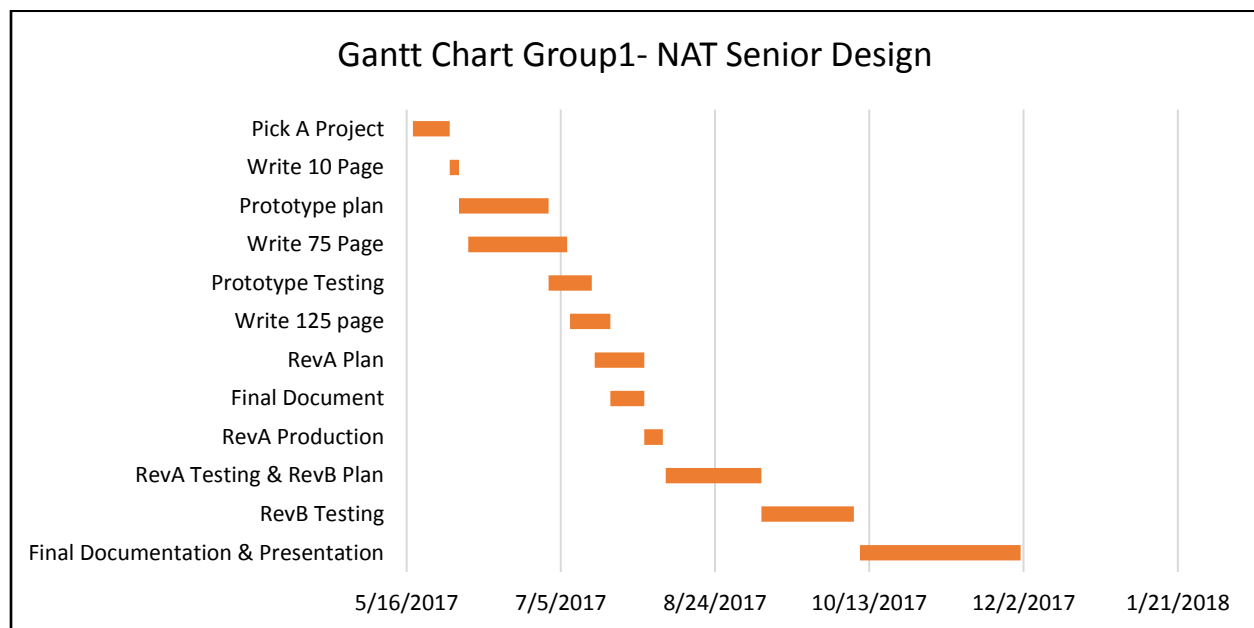


Figure 99 - Initial NAT Project Gantt Chart

The Gantt chart in Figure 99 above was the initial idea of what deadlines the project should function by. This was an initial overview and was then expanded to Figure 100 below to include some more specific project deadlines as the team worked through what work and goals were necessary for the completion of the project. The Project Manager was to use this Gantt chart as a starting point from which to make deadlines for tasks given to team members.

The initial milestones for the end of Senior Design 1

- GUI with basic functionality working
- Record on SD card and display 9 – axis Motion on GUI
- Read preliminary IMU data
- Send data through Ingenu network AND/OR Cellular network

- Understand INS and work has begun on algorithm
- Wiring diagram and schematic finished
- Tested with external wiring to produce rev0 board early next semester

Initial milestones for Senior Design 2

- Complete RevA PCB board early (1-2 weeks in semester)
- RevB PCB board tested past the point of confidence in functionality and have multiple deployed to demonstrate the covered range of device
- Develop a web browser to give real time location of device
- Device has basic INS functionality
 - Biggest issue with this will be power draw from processing power of IMU
- Get price point of device down to ~\$40/unit when producing 1000units

Modified and Expanded Project Milestones

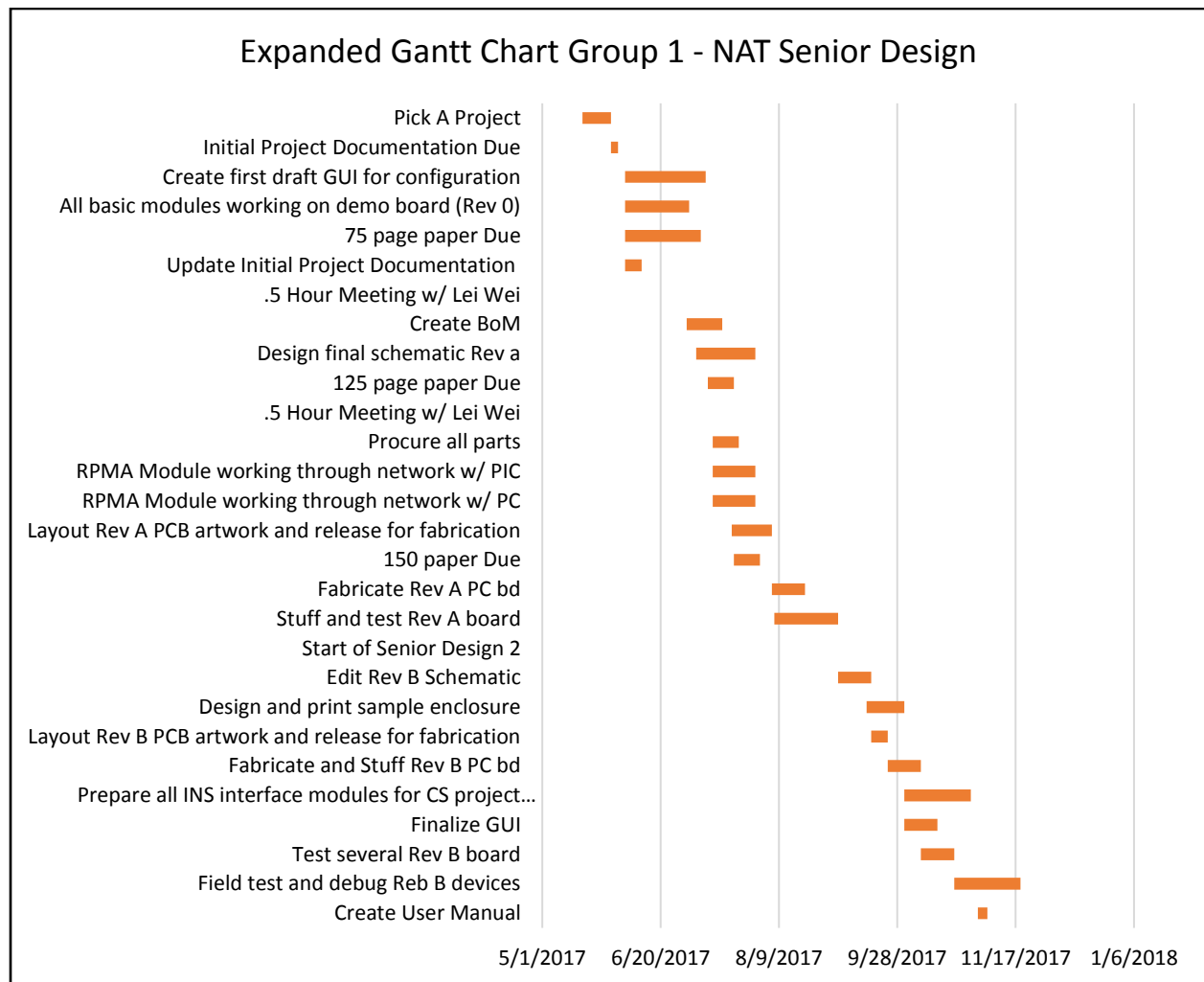


Figure 100 - Expanded NAT Project Gantt Chart

The Gantt Chart shown in Figure 100 shows the overall outlook of the project and the teams development and progression deadlines. These are important to the overall productivity of the team as well as a strong factor in whether or not our project will succeed. In the chart, both the Senior Design class deadlines are shown as well as the project specific deadlines that were discussed and agreed upon by the team and the sponsor/advisor as appropriate deadlines for the goals that were made. These are the initial deadlines, and the team and sponsor are aware that these may change as the project progresses. Therefore, what is seen above may change by the time this document is read.

The project manager utilized this expanded Gantt Chart to set deadlines on tasks given to team members and to decide whether anything needed to be done should the team get off track or behind schedule.

8.2 Budget and Finance Discussion

Initial General Estimated Budget

Description	Estimated Price / Unit
Resistors/Capacitors/Inductors	\$0.10
Pin Connectors	\$0.75
Reed Switch	\$1.00
GPS Modules	\$25.00
Volt/Switch Regulator	\$1.50
MICROSD Socket/Card Reader	\$15.00
Voltage Reference	\$1.50
IC MCU	\$5.00
Motion Sensor IMU	\$10.00
Crystals	\$1.00
NB-IoT	\$15.00
ESTIMATED TOTAL \$95 / board	

Figure 101 - Initial NAT Project General Budget

The above is the estimated budget for one board to be designed and built. This project is sponsored by Young Engineering Services, LLC. The design group was given a budget of \$1200 total. This budget includes the final design of 10 boards. Any part that is to be purchased by one of the group members has to first be signed and approved by Young Engineering Services, LLC, which is

agreed upon in writing by each member. This will ensure that the group stays on budget and all the parts can be purchased.

Final Budget and Financing

Description	Qty	RevA x 3	RevB x 7	Need	Price	Total \$
6 Positions Header Connector 1.5mm	1	3	7	10	0.77	7.70
10 Positions Header Connector 1.5mm	1	3	7	10	0.89	8.90
USB – Micro AB USB 2.0 Receptacle Connector 5 Position	1	3	7	10	0.74	7.40
PMOS 30V 1.5A	7	21	49	70	0.27	18.90
Reed Switch	1	3	7	10	1.06	10.60
GPS Module	1	3	7	10	20.50	205.00
DC-DC Converter	1	3	7	10	1.34	13.40
MICROSD (Socket)	1	3	7	10	10.30	103.00
Voltage Reference	1	3	7	10	1.45	14.50
USB ESD Protection Unit	1	3	7	10	0.58	5.80
IC MCU 16-bit	1	3	7	10	4.55	45.50
Motion Sensor – IMU	1	3	7	10	8.39	83.90
Crystal 32MHz	1	3	7	10	0.76	7.60
Crystal 32.768KHz	1	3	7	10	1.18	11.80
SD Card Reader	1	3	7	10	5.95	59.50
RPMA	1	3	7	10	15.00	150.00
LTE Module	1	3	7	10	99.00	990.00

Figure 102 - Detailed NAT Project Budget

Total Cost = \$ 1,743.50

Total cost is over the initial budget because the team changed from the NB-IoT to the LTE module. The sponsor has approved the use of the LTE module and is reevaluating the budget for the project, as the LTE module is more expensive than the original plan of using the NB-IoT.

8.3 The Winter Park Laboratory

Our group was fortunate enough to work with Young Engineering LLC, a company that is owned and operated by Michael Young. He has provided our engineering group with access to his company's work laboratory, located in Winter Park. This lab is complete with a lot of the hardware and testing equipment that our engineering group may need during the design, development, and production of the NAT device, as well as some additional services.

The Computers

The lab is complete with a set of computers, configured to be used with both the electrical schematic and PCB software for the electrical engineers to utilize, as well as with the programming and software testing software for the computer engineers to utilize. From these computers, the engineers have developed the schematic, the block diagrams, and the various software components such as the TUI, and the configuration GUI.

The Electrical Bench

The lab also has various hardware and testing components that the team may need to utilize during the development and testing of our devices. Within the lab there are various testing hardware including digital multimeters, a soldering station, microscopes, and power supplies that the engineers will use during the scope of this project. In addition to this, within the work station there are various minor components, such as resistors and capacitors, which the team may need during development, debugging, and testing. This station has allowed the team to develop the testing environment of the NAT development board, the setup shown in Figure 57. All of the connectors were developed by our team to develop and test various parts of our project.

Michael Young's Office

This lab is also where our sponsor, Professor Michael Young, has his main office. Since our engineering team works alongside Professor Young, whenever we run into trouble or need help debugging or deciding something it is invaluable to have such an industry professional right there with us to aid in our project.

The Conference Room

Part of the agreement between Professor Young and the students in our group was a contract stating what was expected of the students, of which all of us signed. Part of this contract states that we need to meet once a week to discuss the development of this project. With experience in industry, the students know that this is much like how an engineering project at a company will also run. Therefore, we gain more industry experience working with this type of schedule. The point of stating this, however, is there is a conference room in the lab that allows our team to group together and work on different aspects of our project, or just give us a nice space to conduct our weekly

meeting. There is a white board in this room, as well as another portable one, that allows the team to work together. There is also a projector in the room, should we need to make more formal type presentations.

9.0 Appendixes

9.1 References

- A. Texas Instruments, “TPS6220x High-Efficiency, STO23 Step-Down, DC-DC Converter,” TPS62200 Datasheet, Mar. 2002 [Revised June. 2015].
- B. Texas Instruments, Application Report SNVA559A.
- C. Linear Technology, “1 MHz Step-Down DC/DC Converters in SOT-23,” LTC1701 Datasheet, No Date.
- D. Fairchild Semiconductor, “High-Efficiency Step-Down DC-DC Converter,” FAN5307 Datasheet, Aug. 2009.
- E. Analog Devices, “Chapter 14: Voltage References,” *wiki.analog.com*, sect. 14.1-14.3, June 6, 2017. [Online]. Available: <https://wiki.analog.com/university/courses/electronics/text/chapter-14>. [Accessed May 31, 2017].
- F. Intersil, “300nA NanoPower Voltage References,” ISL21080 Datasheet, May. 2010.
- G. Linear Technology, “1 μ A Precision Series Voltage Reference,” LT6656 Datasheet, July. 2010 [Revised Nov. 2013].
- H. Texas Instruments, “REF30xx 50-ppm/ $^{\circ}$ C Max, 50- μ A, CMOS Voltage Reference in SOT-23,” REF3030 Datasheet, Mar. 2002 [Revised Nov. 2015].
- I. “Quartz Crystal Oscillators,” Internet: www.electronicstutorials.ws/oscillator/crystal.html, [Accessed June 18, 2017].
- J. “A Closer Look at Ingenu RPMA Alternative to LoRa or Sigfox LPWAN Standards & RPMA Development Kit,” Internet: <http://www.cnx-software.com/2016/11/20/a-closer-look-at-ingenu-rpma-alternative-to-lora-or-sigfox-lpwan-standards-rpma-development-kit/>, Nov. 20, 2016 [Accessed June 18, 2017].
- K. “What is LPWAN?,” Internet: <http://dgmatics.com/technology/what-is-lpwan>, [Accessed June 22, 2017].
- L. “On-Ramp Wireless becomes Ingenu, touts purpose-built network for Internet of Things,” Internet: <http://www.fiercewireless.com/wireless/ramp-wireless-becomes-ingenu-touts-purpose-built-network-for-internet-things>, Sep. 8, 2015 [Accessed June 20, 2017].
- M. “Energate Announces Its LC2100, a New, Cost Effective, Easy-to-Deploy Load Control Switch for Utilities,” Internet: <http://globenewswire.com/news-release/2015/04/07/722182/10127711/en/Energate-Announces-Its-LC2100-a-New-Cost-Effective-Easy-to-Deploy-Load-Control-Switch-for-Utilities.html>, April. 7, 2015 [Accessed July 10, 2017].
- N. Dregvaite, Giedre. *Information and Software Technologies*. Switzerland: Springer International Publishing, 2016, pp. 665-670.
- O. “Sigfox publie ses cartes de couverture pour couper l'herbe sous le pied de ses concurrents,” Internet: <http://www.usine-digitale.fr/article/sigfox-publie-ses-cartes-de-couverture-pour-couper-l-herbe-sous-le-pied-de-ses-concurrents.N372467>, Jan. 8, 2016 [Accessed June. 18, 2017].
- P. Lakshmi, Nair A. “Performance Evaluation of Star Topology in Fiber Optic Communication”. *International Journal of Science and Research*, vol. 4 iss. 2, pp 245. Feb. 2015.

- Q. "LoRa Alliance Technology," Internet: <https://www.lora-alliance.org/what-is-lora>, [Accessed July 23, 2017].
- R. "What is LoRa," Internet: <http://www.semtech.com/wireless-rf/internet-of-things/what-is-lora/>, [Accessed June 30, 2017].
- S. "Countries – LoRaWAN™ Networks," Internet: <https://www.lora-alliance.org/>, [Accessed Jun 30, 2017].
- T. 2017. [Online]. Available: <https://www.origingps.com/wp-content/uploads/2017/06/Nano-Hornet-ORG1411-Datasheet-Rev-3.4.pdf>. [Accessed: 05-Jun- 2017].
- U. S. Qualcomm, "SiRFstar IV 4e | Qualcomm", Qualcomm, 2017. [Online]. Available: <https://www.qualcomm.com/products/sirfstar-iv-4e>. [Accessed: 15- Jun- 2017].
- V. U-Blox, 2017. [Online]. Available: https://www.u-blox.com/sites/default/files/products/documents/GPS-Compendium_Book_%28GPS-X-02007%29.pdf. [Accessed: 18- Jun- 2017].
- W. 2017. [Online]. Available: <https://www.navcen.uscg.gov/pubs/gps/gpsuser/gpsuser.pdf>. [Accessed: 18- Jun- 2017].
- X. "A-GPS", Google Books, 2017. [Online]. Available: <https://books.google.com/books?hl=en&lr=&id=stTSHdFhrFUC&oi=fnd&pg=PR13&dq=Assisted+GPS&ots=z8Rh3BscTk&sig=0sDM-GqTbHoDRrIgG-BoxA36ihw#v=onepage&q=Assisted%20GPS&f=false>. [Accessed: 18- Jun- 2017].
- Y. "UCF EZproxy Login", Sciencedirect.com.ezproxy.net.ucf.edu, 2017. [Online]. Available: <http://www.sciencedirect.com.ezproxy.net.ucf.edu/science/article/pii/S0098300411001063?via%3Dihub>. [Accessed: 22- Jun- 2017].
- Z. "Garmin | What is WAAS?", Www8.garmin.com, 2017. [Online]. Available: <http://www8.garmin.com/aboutGPS/waas.html>. [Accessed: 22- Jun- 2017].
- AA. 2017. [Online]. Available: <https://cdn-shop.adafruit.com/datasheets/GlobalTop-FGPMMPA6H-Datasheet-V0A.pdf>. [Accessed: 22- Jun- 2017].
- BB. "Understanding the Difference, and Debunking the Myths, Between Active and Passive Antennas", Blog.rfvenue.com, 2017. [Online]. Available: <http://blog.rfvenue.com/2014/12/15/active-v-passive-anntennas>. [Accessed: 29- Jun- 2017].
- CC. OriginGPS, 2017. [Online]. Available: https://www.origingps.com/wp-content/uploads/2016/04/Origin_brochure_Spider_WEB_21april16-3.pdf. [Accessed: 29- Jun- 2017].
- DD. OriginGPS, 2017. [Online]. Available: <https://www.origingps.com/wp-content/uploads/2017/06/Multi-Micro-Spider-ORG4033-mk05-Datasheet-rev-1.3.pdf>. [Accessed: 22- Jun- 2017].
- EE.A. Nawrat, K. Jedrasiak, K. Daniec and R. Koterak, "Inertial Navigation Systems and Its Practical Applications", 2017. [Online]. Available: <https://www.intechopen.com/books/new-approach-of-indoor-and-outdoor-localization-systems/inertial-navigation-systems-and-its-practical-applications>. [Accessed: 16- Jul- 2017].
- FF. "How it works | Bluetooth Technology Website", Bluetooth.com, 2017. [Online]. Available: <https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works>. [Accessed: 23- Jul- 2017].

- GG. Microchip, 2017. [Online]. Available: <http://ww1.microchip.com/downloads/en/DeviceDoc/50002081B.pdf>. [Accessed: 24- Jul- 2017].
- HH. E. Stinson, P. Sarconi, J. Ray, D. Pierce, D. Pierce and J. McMahon, "Why Everything Wireless Is 2.4 GHz", WIRELESS, 2017. [Online]. Available: <https://www.wired.com/2010/09/wireless-explainer/>. [Accessed: 24- Jul- 2017].
- II. "eCFR — Code of Federal Regulations", Ecf.gov, 2017. [Online]. Available: <https://www.ecfr.gov/cgi-bin/text-idx?SID=8a3ac876eaaef1b5c7e45a09dd15596&mc=true&node=pt47.2.22&rgn=div5>. [Accessed: 24- Jul- 2017].
- JJ. "Custom ASIC Cost Calculator - Sigenics", Sigenics.com, 2017. [Online]. Available: <http://www.sigenics.com/page/custom-asic-cost-calculator>. [Accessed: 02- Jul- 2017].
- KK. R. Reese, J. Bruce and B. Jones, Microcontrollers. Boston, MA: Course Technology, 2009, p. 54.
- LL. "MSP430FR599x, MSP430FR596x Mixed-Signal Microcontrollers (Rev. B) Online Datasheet", Texas Instruments, 2017. [Online]. Available: <http://www.ti.com/product/msp430fr5962/datasheet>. [Accessed: 03- Jul- 2017].
- MM. "ATUCL3U/L4U Series - Complete Datasheet", Microchip, 2013. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-32142-32-bit-Flash-MCU-ATUCL3U-L4U_datasheet.pdf. [Accessed: 05- Jul- 2017].
- NN. "PIC24FJ128GA006 - Microcontrollers and Processors", Microchip, 2012. [Online]. Available: <http://www.microchip.com/wwwproducts/en/PIC24FJ128GA006>. [Accessed: 05- Jul- 2017].
- OO. J. Cooper and A. James, "Challenges for Database Management in the Internet of Things", IETE Technical Review, vol. 26, no. 5, p. 324, 2009.
- PP. C. Pitt, Pro PHP MVC. Berkeley, CA: Apress, 2012, pp. 1-2.
- QQ. "C# Coding Conventions (C# Programming Guide)", Docs.microsoft.com, 2017. [Online]. Available: <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions>. [Accessed: 16- Jul- 2017].
- RR. "GNU Coding Standards: Writing C", Gnu.org, 2017. [Online]. Available: https://www.gnu.org/prep/standards/html_node/Writing-C.html. [Accessed: 16- Jul- 2017].
- SS. "JavaScript Standard Style", Standardjs.com, 2017. [Online]. Available: <https://standardjs.com/rules.html>. [Accessed: 16- Jul- 2017].
- TT. "Manual :: Coding Standards", Pear.php.net, 2017. [Online]. Available: <http://pear.php.net/manual/en/standards.php>. [Accessed: 20- Jul- 2017].
- UU. R. Somasundaram, Git: Version Control for Everyone Beginner's Guide. Birmingham UK: Packt Publishing, 2013, pp. 28-42.
- VV. R. Frey, "File:MVC-Process.svg -", Commons.wikimedia.org, 2010. [Online]. Available: <https://commons.wikimedia.org/wiki/File:MVC-Process.svg>. [Accessed: 29- Jul- 2017].
- WW. "LTE Cat M1", u-blox, 2017. [Online]. Available: <https://www.u-blox.com/en/lte-cat-m1>. [Accessed: 15- Jul- 2017].
- XX. "LTE Cat M1", u-blox, 2017. [Online]. Available: <https://www.u-blox.com/en/lte-cat-m1>. [Accessed: 15- Jul- 2017].

- YY. B. Ray, "LTE eDRX and PSM Explained for LTE-M1", *Link-labs.com*, 2017. [Online]. Available: <https://www.link-labs.com/blog/lte-e-drx-psm-explained-for-lte-m1>.
- ZZ. "Long Term Evolution for Machines: LTE-M | Internet of Things", *Internet of Things*, 2017. [Online]. Available: <https://www.gsma.com/iot/long-term-evolution-machine-type-communication-lte-mtc-cat-m1/>. [Accessed: 01- Jul- 2017].
- AAA. Y. Hwang, "Cellular IoT Explained - NB-IoT vs. LTE-M vs. 5G and More | IoT For All", *IoT For All*, 2017. [Online]. Available: <https://iot-for-all.com/cellular-iot-explained-nb-iot-vs-lte-m/>. [Accessed: 01- Jul- 2017].
- BBB. "Skywire™ 4G LTE CAT M1 Embedded Modem - NimbeLink : Build Your IoT Device Faster", *NimbeLink : Build Your IoT Device Faster*, 2017. [Online]. Available: <http://nimbelink.com/skywire-4g-lte-cat-m1/>. [Accessed: 01- Jul- 2017].
- CCC. M. DeGrasse, "LTE for IoT: Cat 1 vs. Cat M1", *Enterprise IoT Insights*, 2017. [Online]. Available: <http://enterpriseiotinsights.com/20170522/chipsets/lte-for-iot-cat-1-vs-cat-m1-tag4>. [Accessed: 01- Jul- 2017].
- DDD. "Cellular IoT alphabet soup | Ericsson Research Blog", *Ericsson Research Blog*, 2017. [Online]. Available: <https://www.ericsson.com/research-blog/internet-of-things/cellular-iot-alphabet-soup/>. [Accessed: 01- Jul- 2017].
- EEE. "LTE Cat-M1", *YouTube*, 2017. [Online]. Available: <https://www.youtube.com/watch?v=XsL0Q3Htf1Y>. [Accessed: 01- Jul- 2017].
- FFF. 2017. [Online]. Available: http://www.sequans.com/wp-content/uploads/2016/06/PI-Monarch-4-20160617_Web.pdf. [Accessed: 01- Jul- 2017].
- GGG. "GPS Tracking Devices | Real-Time Personal & Car GPS Trackers", *Spytecinc.com*, 2017. [Online]. Available: <http://www.spytecinc.com/gps-satellite-tracking.html>. [Accessed: 01- Jul- 2017].
- HHH. 2017. [Online]. Available: https://ipc.kavi.com/higherlogic/ws/public/download/4721/IPC-HDBK-001F_Final%20Industry%20Review_Version%202_FINAL.pdf. [Accessed: 01- Jul- 2017].
- III. 2017. [Online]. Available: http://www.huawei.com/minisite/iot/img/nb_iot_whitepaper_en.pdf. [Accessed: 01- Jul- 2017].
- JJJ.5. guides and W. IoT?, "What is Narrowband IOT?", *5g.co.uk*, 2017. [Online]. Available: <https://5g.co.uk/guides/what-is-narrowband-iot/>. [Accessed: 01- Jul- 2017].
- KKK. "How Big of a Problem is Employee Theft and Fraud?", *Incorp.com*, 2017. [Online]. Available: <https://www.incorp.com/help-center/business-articles/employee-theft-and-fraud-part1>. [Accessed: 15- Jun- 2017].
- LLL. 2017. [Online]. Available: http://www.eecs.ucf.edu/seniordesign/sp2013su2013/g08/HTS_ConferencePaperLion.pdf. [Accessed: 01- Jul- 2017].
- MMM. 2017. [Online]. Available: <http://www.eecs.ucf.edu/seniordesign/su2013fa2013/g14/documents/CP.pdf>. [Accessed: 01- Jul- 2017].
- NNN. "Composition of a Printed Circuit Board - ALLPCB.com", *Allpcb.com*, 2017. [Online]. Available: http://www.allpcb.com/pcb_composition.htm. [Accessed: 01- Jul- 2017].

- OOO. [Online]. Available: <https://www.silabs.com/documents/public/application-notes/AN249.pdf>. [Accessed: 29- Jul- 2017].
- PPP. [Online]. Available: https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU925_0REV1.0.pdf. [Accessed: 29- Jul- 2017].
- QQQ. "Inertial measurement unit", *En.wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Inertial_measurement_unit. [Accessed: 29- Jul- 2017].
- RRR. R. Journal, "Frequently Asked Questions - RFID Journal", *Rfidjournal.com*, 2017. [Online]. Available: <http://www.rfidjournal.com/site/faqs>. [Accessed: 29- Jul- 2017].
- SSS. "Radio-frequency identification", *En.wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Radio-frequency_identification. [Accessed: 29- Jul- 2017].
- TTT. "What is RFID? - EPC-RFID", EPC-RFID, 2017. [Online]. Available: <http://www.epc-rfid.info/rfid>. [Accessed: 29- Jul- 2017].
- UUU. "RFID Tags Information | Engineering360", *Globalspec.com*, 2017. [Online]. Available: http://www.globalspec.com/learnmore/data_acquisition_signal_conditioning/data_input_devices/rfid_tags. [Accessed: 29- Jul- 2017].
- VVV. S. ADXL335, S. Kit, B. Straight, L. ADXL335, B. Angle, S. ADXL337, S. ADXL362, S. ADXL377, S. LIS3DH and T. ADXL335, "SparkFun Triple Axis Accelerometer Breakout - ADXL335 - SEN-09269 - SparkFun Electronics", *Sparkfun.com*, 2017. [Online]. Available: <https://www.sparkfun.com/products/9269>. [Accessed: 29- Jul- 2017].
- WWW. S. ADXL345, A. R3, B. Straight, G. SM-24, S. ADXL337, S. ADXL362, S. Headers), S. ADXL377, S. LIS3DH and S. LIS331, "SparkFun Triple Axis Accelerometer Breakout - ADXL345 - SEN-09836 - SparkFun Electronics", *Sparkfun.com*, 2017. [Online]. Available: <https://www.sparkfun.com/products/9836>. [Accessed: 29- Jul- 2017].
- XXX. S. LIS331, B. Straight, S. ADXL345, S. ADXL337, S. ADXL362, S. ADXL377, S. LIS3DH, 3. MMA8452Q, T. LIS331HH and S. Bi-Directional, "SparkFun Triple Axis Accelerometer Breakout - LIS331 - SEN-10345 - SparkFun Electronics", *Sparkfun.com*, 2017. [Online]. Available: <https://www.sparkfun.com/products/10345>. [Accessed: 29- Jul- 2017].
- YYY. S. L3G4200D, A. R3, S. ITG-3200, L. v3, S. USB, S. ADXL345, F. 4.5" and F. 2.2" and A. 3.3V/8MHz, H. One and A. R3, "SparkFun Tri-Axis Gyro Breakout - L3G4200D - SEN-10612 - SparkFun Electronics", *Sparkfun.com*, 2017. [Online]. Available: <https://www.sparkfun.com/products/10612>. [Accessed: 29- Jul- 2017].
- ZZZ. S. ITG3200/ADXL345, S. MPU-9250, B. Straight, S. MPU-6050, S. MPL3115A2, S. LSM303C, S. LSM6DS3, L. v3, A. R3 and S. Bi-Directional, "SparkFun 6 Degrees of Freedom IMU Digital Combo Board - ITG3200/ADXL345 - SEN-10121 - SparkFun Electronics", *Sparkfun.com*, 2017. [Online]. Available: <https://www.sparkfun.com/products/10121>. [Accessed: 29- Jul- 2017].
- AAAA. S. MPU-6050, S. MPU-9250, S. ITG3200/ADXL345, 3. MPU-6050, A. R3, R. 3, B. Straight, S. Bi-Directional, L. v3 and A. R3, "SparkFun Triple Axis Accelerometer and Gyro Breakout - MPU-6050 - SEN-11028 - SparkFun Electronics", *Sparkfun.com*, 2017. [Online]. Available: <https://www.sparkfun.com/products/11028>. [Accessed: 29- Jul- 2017].

- BBBB. *Webshop.atlantikelektronik.de*, 2017. [Online]. Available: http://webshop.atlantikelektronik.de/Webpage/CSR_Human%20Interface%20Device%20Profile.html. [Accessed: 29- Jul- 2017].
- CCCC. "Accelerometer, Gyro and IMU Buying Guide - SparkFun Electronics", Sparkfun.com,2017.[Online].Available:https://www.sparkfun.com/pages/accel_gyro_guide. [Accessed: 29- Jul- 2017].
- DDDD. "Machine Network Nationwide Coverage, Internet: <https://www.ingenu.com/technology/machine-network/coverage-tracker/>, [Accessed: Jul. 29, 2017].

9.2 Copyright Permissions

Every image used that was not of the engineering teams own work, permission has been requested by the images' owners. Proof of these requests is shown in the images below.

P1 Reprinted with requested permission from SpyTech

Contact us

< [Spy Tec Products](#)

Sales Questions

Technical Support

Other Issues

Copyright Image Use

Please describe your question.

Hello,

I am a student at the University of Central Florida and would like to request permission to use (http://www.spytecinc.com/media/catalog/product/cache/1/small_image/200x200/9df78eab33525d08d6e5fb8d27136e95/3/8/381708123282014.qbmccp3g4bdwwradayje_height640.png) in my senior design paper.

Thanks,
Brittney Fry

Your Name
Brittney Fry

Your Email
brittney.fry.2013@knights.ucf.edu

Phone Number (optional)

Submit

Image Request 1 - Figure 16 (GX350)

P2 Reprinted with requested permission from Ingenu

Brittney Fry
Fry
Student
University of Central Florida
brittney.fry.2013@knights.ucf.edu
9044776402
United States
Florida
Hello,
I am a student at the University of Central Florida and wanted to request permission to use your image (<http://www.ingenu.com/wp-content/uploads/2015/09/rACM-Ingenu-Transparent-Medium-1024x749.png>) in my senior design paper.
Thank you,
Brittney Fry

Image Request 2 - Ingenu RPMA

P3 Reprinted with requested permission from EvolutionSide

EvolutionSide
About CONTACT US PRIVACY POLICY
CONTACT US
Your Name (required)
Brittney Fry
Your Email (required)
brittney.fry.2013@knights.ucf.edu
Subject
Using Image from Site
Your Message
I am a student at the University of Central Florida and I would like permission to use an image I found on your site (<http://sahajinfosys.com/wp-content/uploads/2016/01/Sahaj-GPS-Tracker-Banner-Map-1024x1024.jpg>) in my Senior Design Paper.
Thanks,
Brittney Fry
Send

Image Request 3 - Figure 1 (Introduction GPS Map)

P4 Reprinted with requested permission from UnknownFieldsDivision

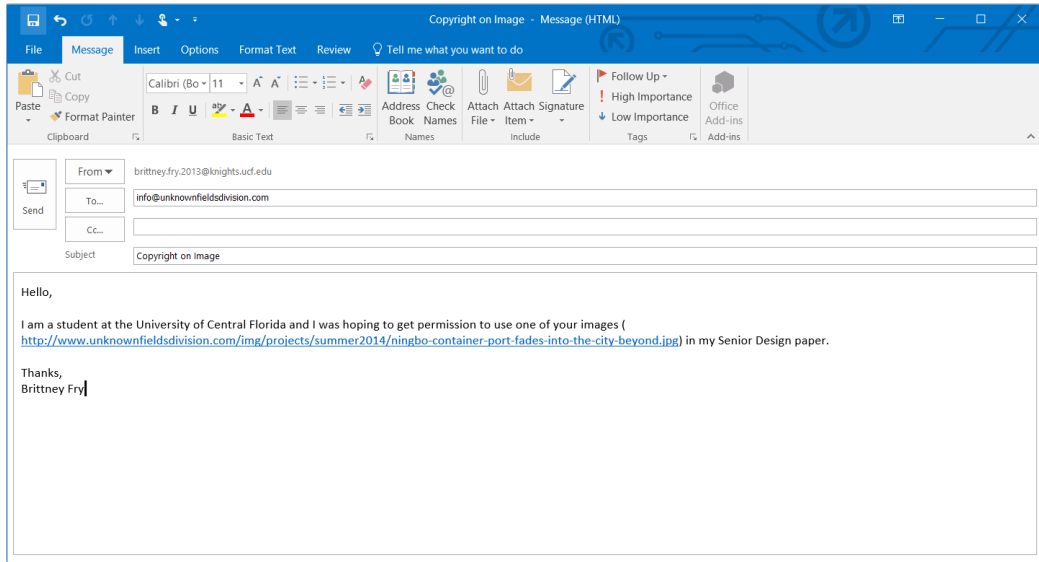


Image Request 4 - Figure 2 (Introduction Shipping Containers)

P5 Reprinted with requested permission from IntelTrack

Name *
Brittney Fry

Email *
brittney.fry.2013@knights.ucf.edu

Phone
9042776402


Company
University of Central Florida - Student

Which topic best describes your question?
 Product support
 Sales Inquire
 Product Inquire
 Image Copyrights

Details
I am a student at the University of Central Florida and would like to request permission to use https://www.inteltrack.com/wp-content/uploads/2017/01/STI_Bolt.jpg in my senior design paper.

Thanks,
Brittney Fry

Devices needed
 1-20
 21-50
 50 plus

Captcha


Submit

Image Request 5 - Figure 15 (STI_Bolt)

P6 Reprinted with requested permission from SpyTec

< Spy Tec Products

Sales Questions	▼
Technical Support	▼
Other Issues	>

Your inquiry was submitted and will be responded to as soon as possible. Thank you for contacting us.

Contact reason
Copyright Image Use

Please describe your question.
Hello,
I am a student at the University of Central Florida and would like to request permission to use (<http://www.spytecinc.com/media/catalog/product/cache/1/image/390x390/9df78eab33525d08d6e5fb8d27136e95/g/q/300.jpg>) in my senior design paper.
Thanks,
Brittney Fry

Your Name
Brittney Fry

Your Email
brittney.fry.2013@knights.ucf.edu

Phone Number (optional)
|

Submit

Image Request 6 - Figure 17 (STI_GL300)

P7 Reprinted with requested permission from SpyTec

< Spy Tec Products

Sales Questions	▼
Technical Support	▼
Other Issues	>

Please describe your question.
I am a student at the University of Central Florida and would like to request permission to use (http://www.spytecinc.com/media/catalog/product/cache/1/image/390x390/9df78eab33525d08d6e5fb8d27136e95/x/t/xt-2000_obd_tracker.png) in my senior design paper.
Thanks,
Brittney Fry

Your Name
Brittney Fry

Your Email
brittney.fry.2013@knights.ucf.edu

Phone Number (optional)
|

Submit

Image Request 7 - Figure 14 (XT_2000)

P8 Reprinted with requested permission from Ingenu

Brittney

Fry

Student

University of Central Florida

brittney.fry.2013@knights.ucf.edu

9044776402

United States

Florida

Hello,

I am a student at the University of Central Florida and I would like to request permission to use your image (https://www.ingenu.com/wp-content/uploads/2016/11/ublog_NANO_small.jpg) in my Senior Design paper.

Thank you,
Brittney Fry

Image Request 8 - Figure 54 (RPMA Module)

P9 Reprinted with requested permission from Unity

Send Attach Discard ...

To legal@unity3d.com

Cc

Copyright Request

Hello,

I am a student at the University of Central Florida and I would like to request permission to use your image (<http://answers.unity3d.com/storage/temp/89563-hid-interface-embadded.jpg>) in my Senior Design paper.

Thank you,
Brittney Fry

Send Discard Attach ...

Draft saved at 11:05 PM

Image Request 9 - Figure 47 (HID)

P10 Reprinted with requested permission from Link Labs

GET IN TOUCH

Purpose of inquiry*

Miscellaneous/Other ▼

First Name*

Wayne

Last Name*

Marshall

Email*

waynemarshall@knights.ucf.edu

Company Name*

University of Central Florida

Phone Number

What country will you be deploying your solution in?*

United States ▼

State/Region*

Florida ▼

Role or Title*

Other ▼

Other

Student

Notes

To whom it may concern,

My name is Wayne Marshall, and I am an Electrical Engineering student at the University of Central Florida. my senior design group is working on a project and we would like to use images displaying PSM and eDRX of the LTE module in our documentation. may we please have your permission to do so?

Image Request 10 - Figure 24 and Figure 25 (Power Profile for LTE-M1)

P11 Reprinted with requested permission from Ingenu

The screenshot shows an email request form from Ingenu. The form includes a search bar with the text "Wayne", a dropdown menu for "United States", and a text area containing the following text:

To whom it may concern,

My name is Wayne Marshall, and I am an Electrical Engineering student at the University of Central Florida. My senior design group is working on a project and we would like to add graphs from your [NanoNode](#) Integration Specifications document into our final document. The images of interest are the ones displaying oscillator calibration state and Current Consumption During Deep Sleep, Idle, RX, and TX of the [RPMA](#) module. May we please have your permission to use these images?

Image Request 11 - Figure 89 and Figure 90 (RPMA Oscillator Calibration State & RPMA current consumption in different states)

P12 Reprinted with requested permission from Intersil

The screenshot shows a contact form from Intersil. The form includes the following fields:

- Topic: General Inquiry
- First Name: Wayne
- Last Name: Marshall
- Email Address: waynemarshall@knights.ucf.edu
- Company Name: University of Central Florida
- Job Title: Student
- Address 1:
- Address 2:
- Postal Code: 32816
- Country: USA
- State: Florida
- Phone:
- Preferred Distributor: None
- Comments: Engineering student at the University of Central Florida. My senior design group and I would like to use an image from the datasheet of the Intersil ISL21080 as a visual aid

Image Request 12 - Figure 95 (Voltage Reference Generator Dropout voltage)

P13 Reprinted with requested permission from Huawei

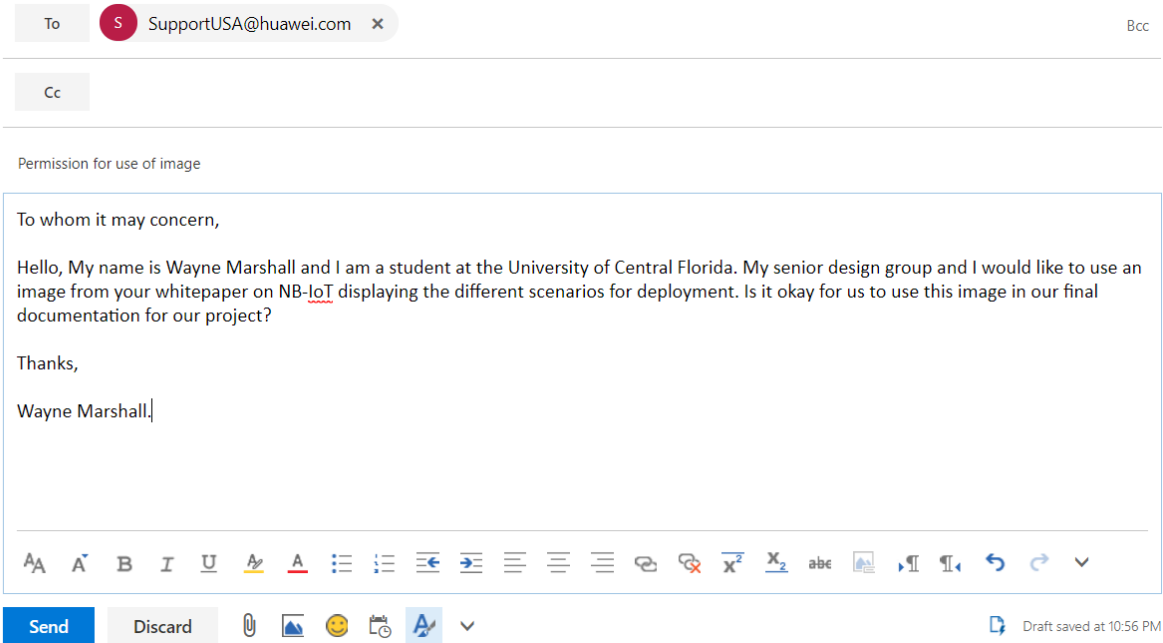


Image Request 13 - Figure 20 (Deployment scenarios of NB-IoT)

P14 Reprinted with requested permission from AllPCB

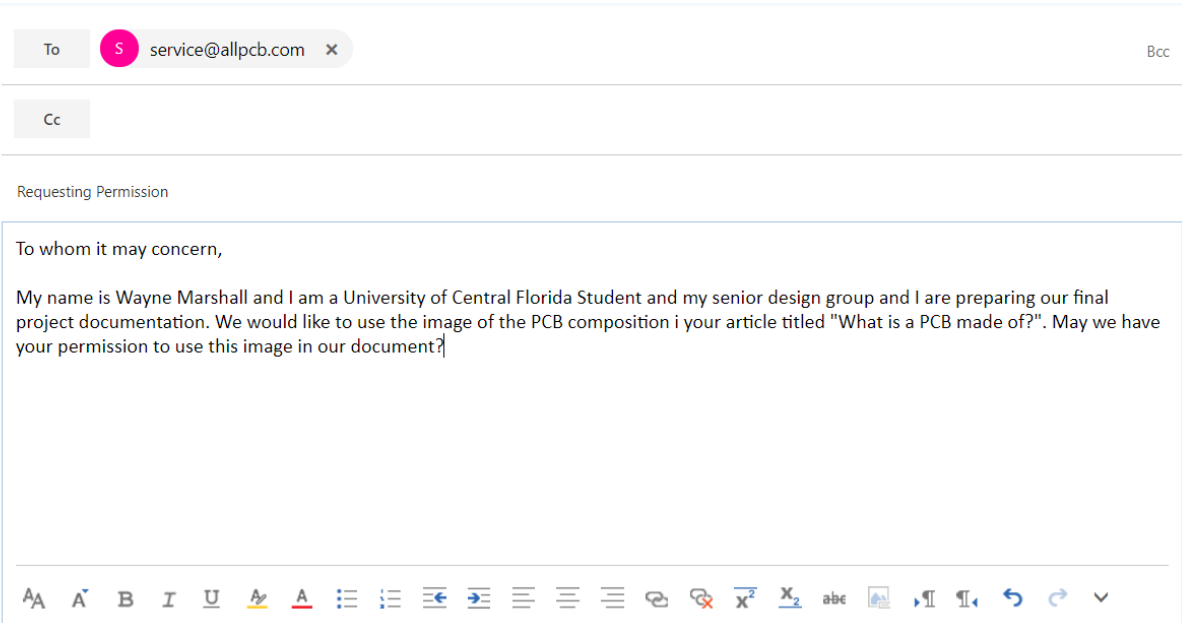


Image Request 14 – Figure 59 (Basic PCB Composition)

P15 Reprinted with requested permission from Andres from HelmetTracker UCF Senior Design Project

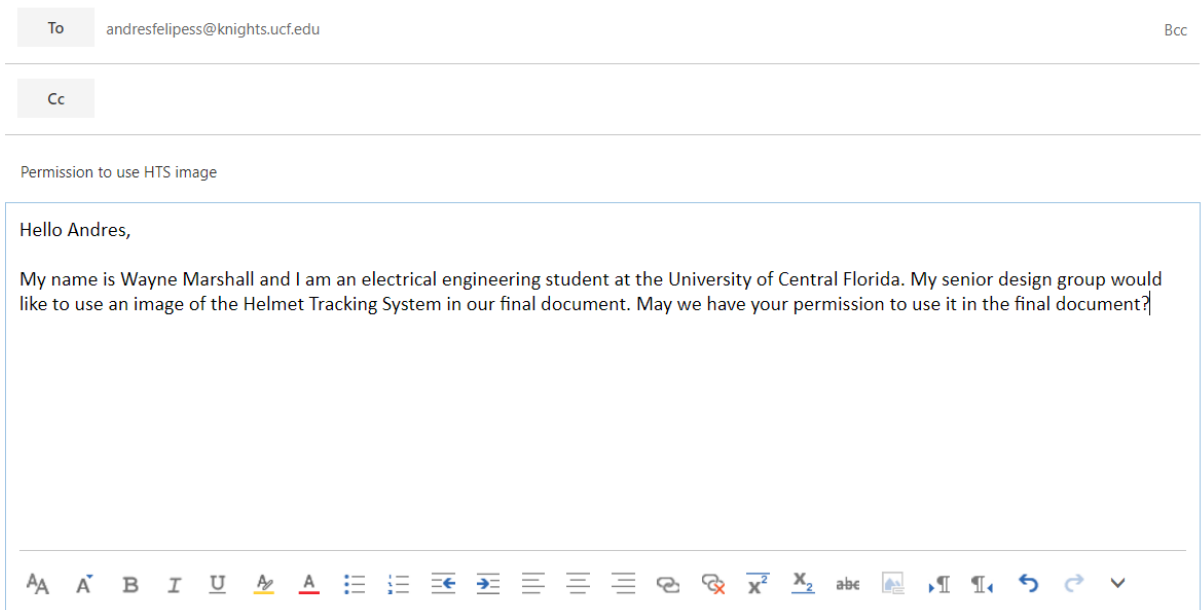


Image Request 15 – Figure 13 (Similar projects - Helmet Tracking System)

P16 Reprinted with requested permission from InvenSense

MPU-9250 Permission Request



Image Request 16 – Figures 33, 34, 36, 37, 38 (All MPU 9250 related images)

P17 Reprinted with requested permission from InvenSense

Permission Request - UCF Senior Design



Brianna Thomason
Today, 2:50 AM
TechSupport@sparkfun.com

Reply all

To Whom it may Concern,

I am a Computer Engineering student at the University of Central Florida in Orlando, Florida. My senior design group is requesting to reuse some pictures and cite some information found on your website.


More specifically the images found at:

1. <https://www.sparkfun.com/products/9269>
2. <https://www.sparkfun.com/products/9836>
3. <https://www.sparkfun.com/products/10345>
4. <https://www.sparkfun.com/products/10612>
5. <https://www.sparkfun.com/products/10121>
6. <https://www.sparkfun.com/products/11028><https://www.sparkfun.com/products/11028>

Sincerely,
Brianna Thomason
UCF CpE Senior

Image Request 17 – Figure 39, 40, 41, 42, 43, 44, 45 (All related IMU not MPU 9250)

P18 Reprinted with requested permission from Bar Code Graphics

To  support@barcode-us.com x Bcc

Cc

Copyright Permissions

Hello I am student at UCF, and I wanted to request permission to use your image (<http://www.epc-rfid.info/wp-content/themes/gintinfo/images/how%20rfid%20works.png>) in my Senior Design paper.

Thanks,
Brittney Fry




A A B I U   Discard 

Image Request 18 - Figure 27 (RFID Communication)

P19 Reprinted with requested permission from InTechOpen

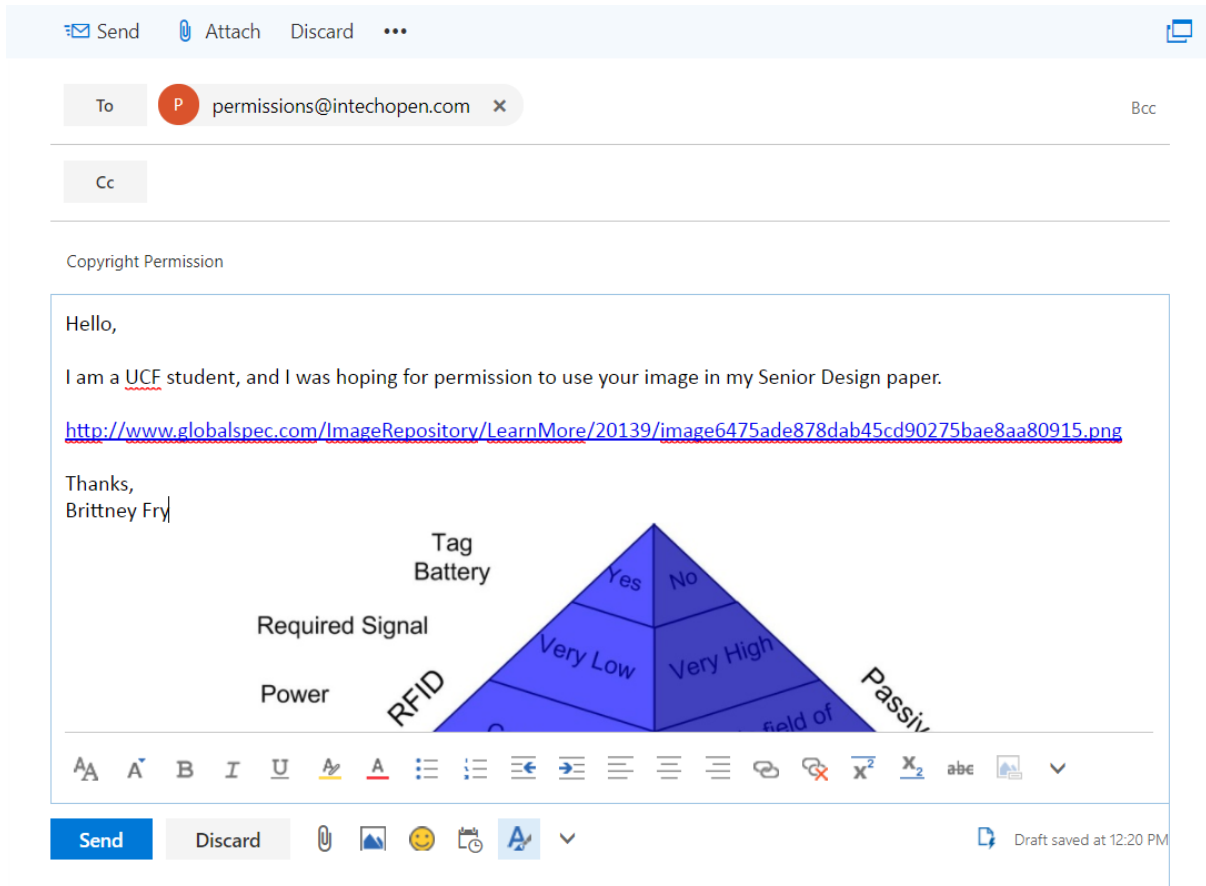
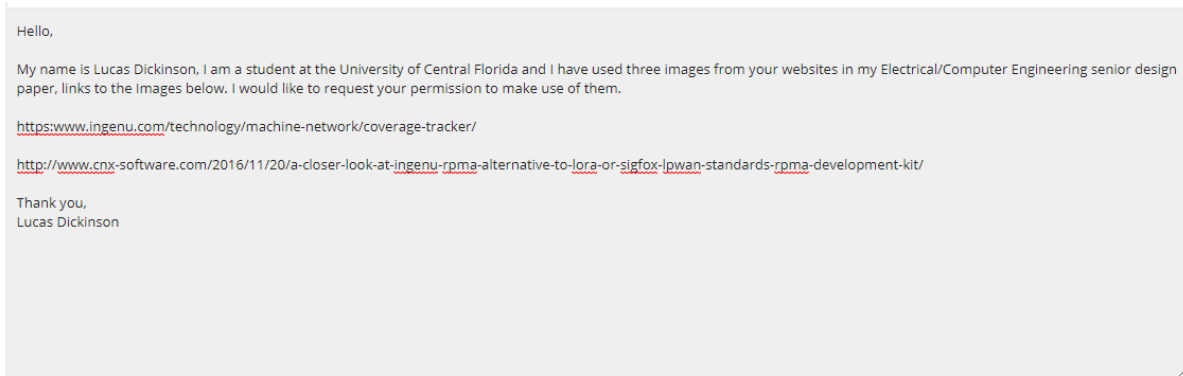


Image Request 19 - Figure 28 (RFID Passive vs Active)

P19 Reprinted with requested permission from Ingenu



P20 Reprinted with requested permission from Atlanktik Elektronik

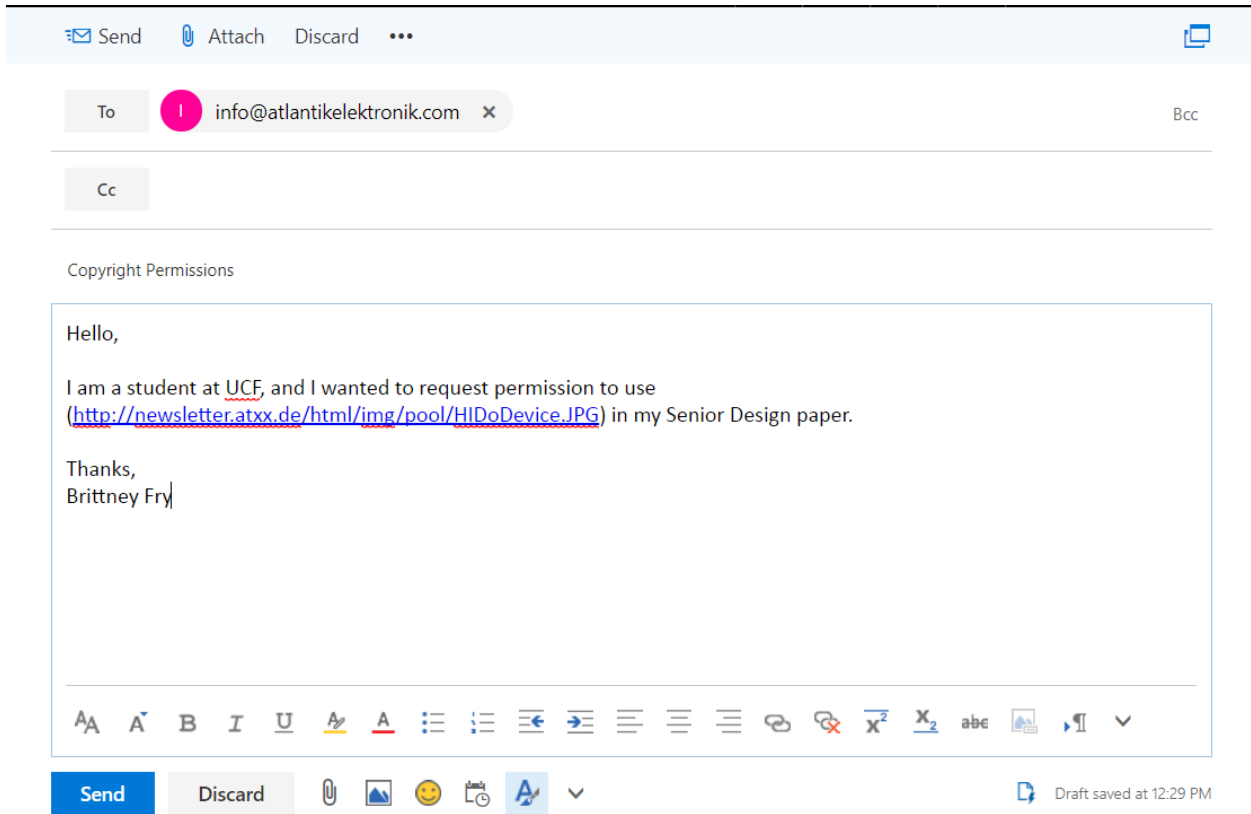


Image Request 20 - Figure 46 (HID Profile)

P21 Reprinted with requested permission from The LoRa Alliance

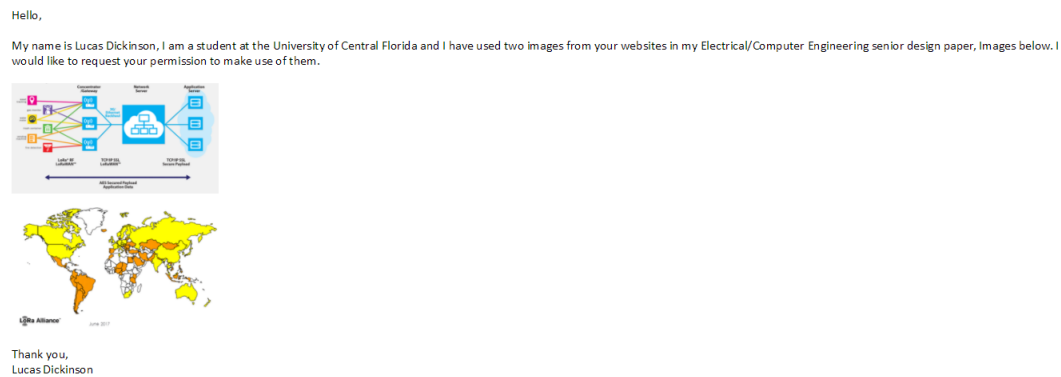
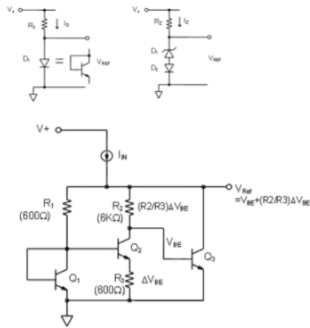


Image Request 21 - Figure 19, 22 (LoRa)

P22 Reprinted with requested permission from wikiAnalog

Hello,

My name is Lucas Dickinson, I am a student at the University of Central Florida and I have used two images from your websites in my Electrical/Computer Engineering senior design paper, Images below. I would like to request your permission to make use of them.



Thank you,
Lucas Dickinson

Image Request 22 - Figure 65, 66 (Voltage References)

P23 Reprinted with requested permission from Electronics Tutorials

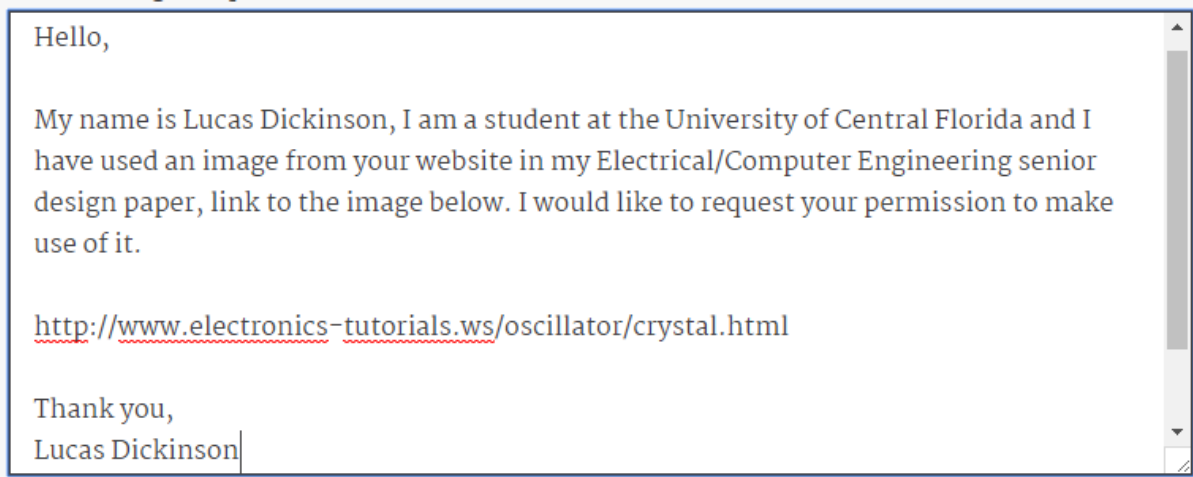


Image Request 23 - Figure 69, 70 (Oscillators)

P24 Reprinted with requested permission from CNX-software

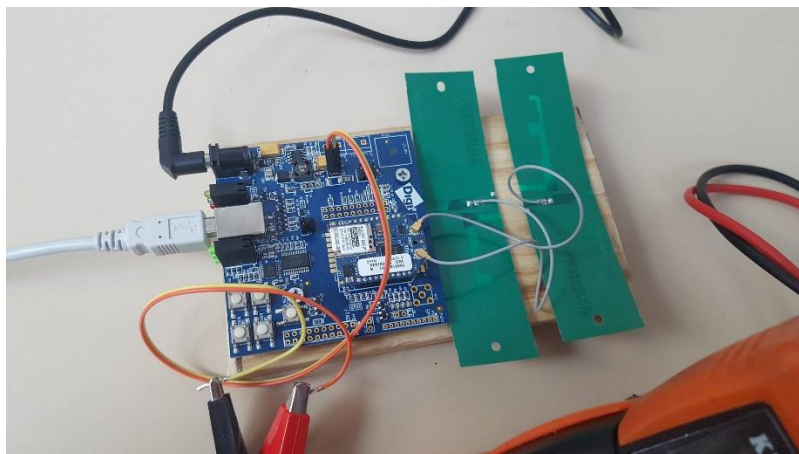
If you have a new great product or news related to embedded systems, let us know in the form below. You can also use this form for other queries. However, please do NOT use this form asking for free technical help, as we probably just delete your email.

Your Name (required)	<input type="text" value="Brittney Fry"/>
Your Email (required)	<input type="text" value="brittney.fry.2013@knights.ucf.edu"/>
Subject	<input type="text" value="Copyright Permissions"/>
Your Message	<div style="border: 1px solid #ccc; padding: 5px;"><p>I am a student at UCF and would like to request permission to use your image (http://www.cnx-software.com/wp-content/uploads/2016/11/Comparison-Sigfox-Lora-GSM-NB-IoT-LTE-Cat-M1-RPMA.jpg) in my senior design paper.</p><p>Thanks, Brittney Fry</p></div>
	3 8 D X
Enter the text you see above	<input type="text" value="38DX"/>
	<input type="button" value="Send"/>

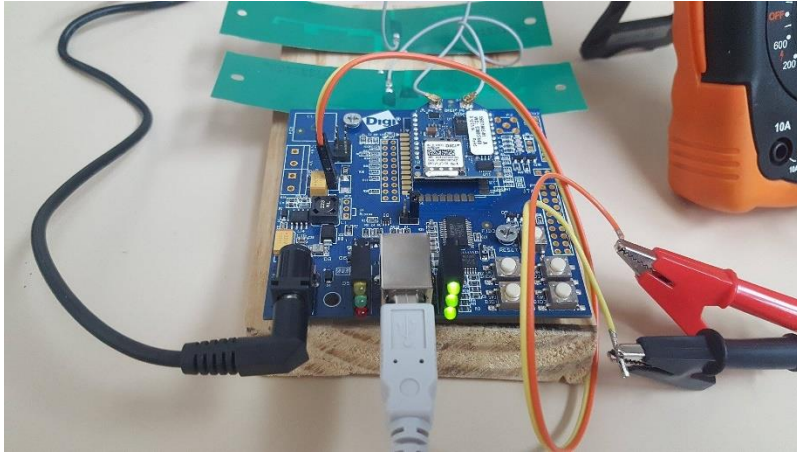
Image Request 24 - Figure 23 (LPWAN Comparison)

9.3 Additional Testing Images

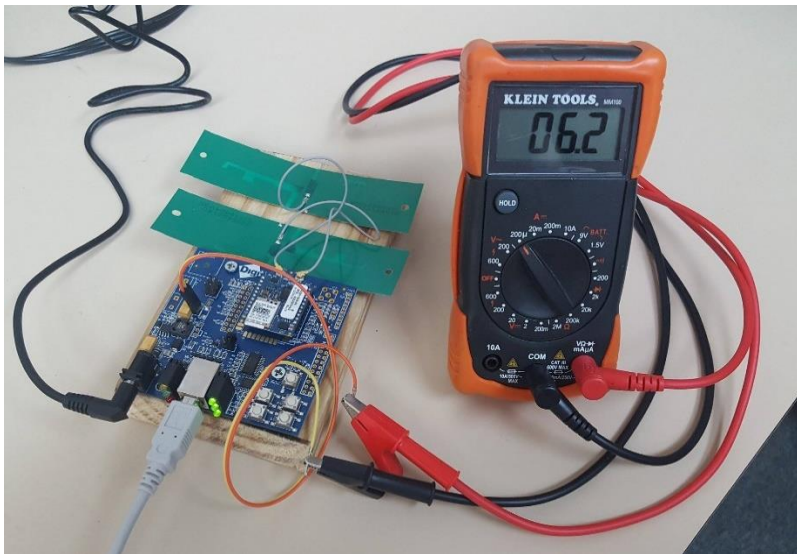
Testing the Digi XBee Cellular LTE CAT-1 Module



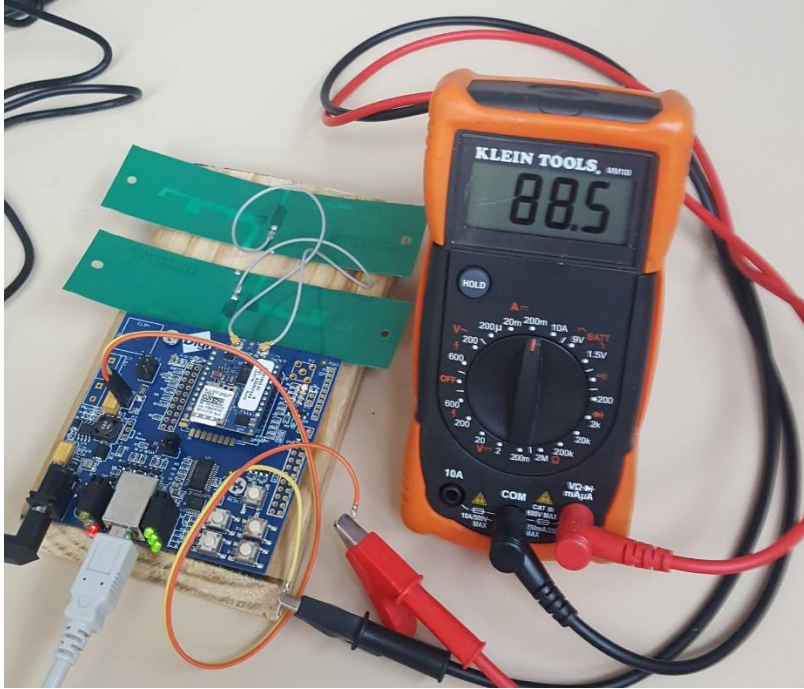
Appendix Testing Image 1 - Test Setup for LTE Module Testing



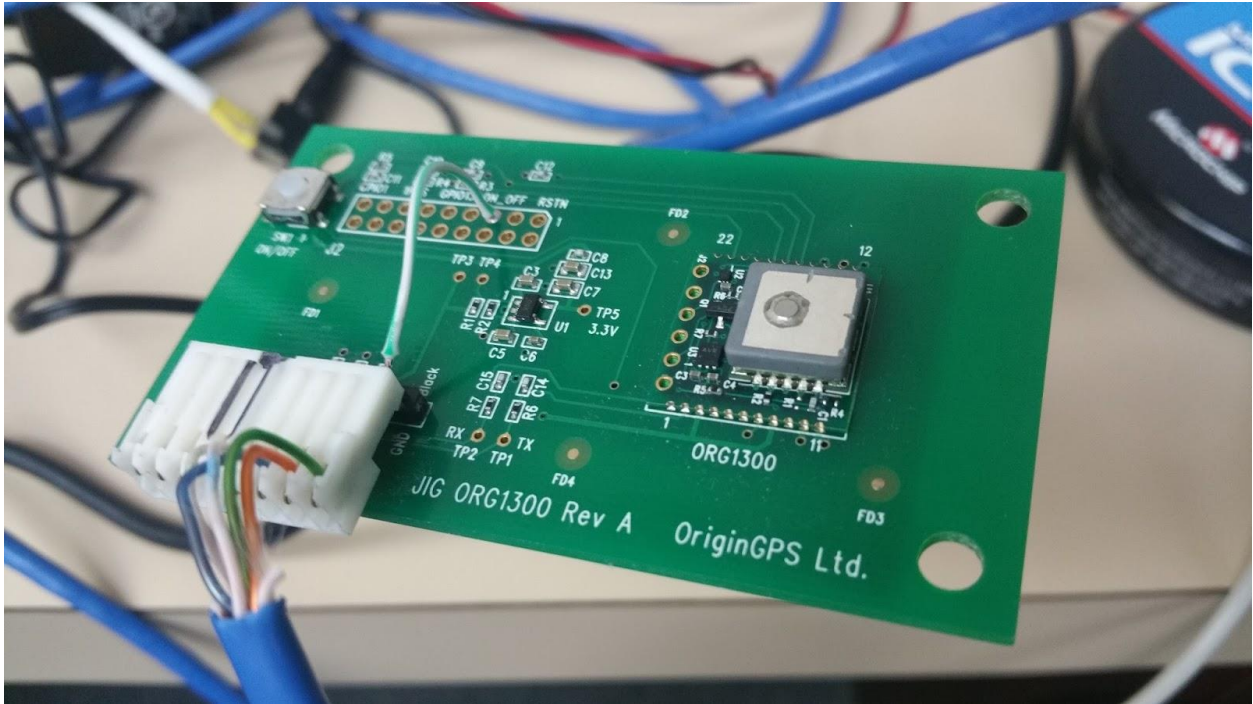
Appendix Testing Image 2 - Testing LTE Module - Alternate View



Appendix Testing Image 3 - Testing the LTE Module - Sleep Mode



Appendix Testing Image 4 - Testing LTE Module - Idle Mode



Appendix Testing Image 5 - GPS Module