PRELIMINARY DESIGN REVIEW FOR A GENERIC CANBUS ACTUATOR MODULE FOR MOBILITY ANTENNAS

Shane Fontaine  Christopher Anderson  Gonzalo Albaladejo  Ryan Mailszewski

Project Sponsors:
ViaSat
ELEC 491
Electrical Engineering Design and Practice
University of San Diego
November 26, 2013
I. Abstract

A CAN bus actuator module for mobile receivers allows for a modular system that has the ability rotate the receiver freely. Given different circumstances, one may need to set coordinates for an antenna to point to a specific location, or to continually adjust in order to stay in line with an object. While this module is intended for an application rotating a mobile receiver, the device is more applicable for vehicular use. CAN buses are commonly used in the automotive industry, so this module can be widely integrated.

This module allows receivers on top of vehicles to constantly be facing the same satellite. This constant communication with the satellite allows for Wi-Fi access inside the vehicle. By constantly updating GPS coordinates and reevaluating its position, the actuator module can continuously exchange information with the satellite and provide an unbroken WiFi signal to the user. The small size and affordability of the module gives it functionality for many users.

A servo motor functions as the motor for the actuator module, and uses an auxiliary power source. Embedded logic within the system provides sufficient instruction for the actuator to perform correctly, and servo drivers allow the software to properly communicate with the hardware used in the system. The parts have been received, part of the code has been written, and the design has been finalized. The remainder of the project will be dedicated to integrating all of these parts into a final product.
# II. Table of Contents

## Contents

I. Abstract ................................................................................................................................................. ii

II. Table of Contents ................................................................................................................................. iii

III. List of Figures .................................................................................................................................... v

IV. List of Tables .................................................................................................................................... vi

I. Introduction .......................................................................................................................................... 1  
   a. Background ....................................................................................................................................... 1  
   b. Purpose ............................................................................................................................................. 1  
   c. Literature Review .............................................................................................................................. 2  
      i. Prior Work ..................................................................................................................................... 2  
      ii. Patents .......................................................................................................................................... 2  
      iii. Codes and Standards .................................................................................................................... 4  

II. Problem Definition ................................................................................................................................ 5  
   a. Project Requirements ....................................................................................................................... 5  
   b. Constraints ........................................................................................................................................ 6  

III. Design Specification .......................................................................................................................... 6  
   a. Design Overview and Deliverables .................................................................................................... 6  
   b. Functional Specifications .................................................................................................................. 8  
   c. Physical Specifications ...................................................................................................................... 8  

IV. Design Results ................................................................................................................................... 8  
   a. System Design ................................................................................................................................... 8  
   b. Positioning Subsystem Design ......................................................................................................... 12  
   c. CAN Bus Subsystem Design ............................................................................................................. 13  
   d. Servo Subsystem Design ................................................................................................................. 16  

V. Design Plan ...................................................................................................................................... 18  
   a. Stage 1 – Research ........................................................................................................................... 18  
   b. Stage 2 – Design .............................................................................................................................. 18  
   c. Stage 3 – Prototype Construction .................................................................................................. 19  
   d. Stage 4 – Testing .............................................................................................................................. 19
e. Stage 5 – Documentation ................................................................................................................ 19
f. Schedule .......................................................................................................................................... 20
g. Budget ............................................................................................................................................. 21
h. Personnel ........................................................................................................................................ 22
VI. References ...................................................................................................................................... 25
VII. Appendices ...................................................................................................................................... 26
    Appendix A: Stand-Alone CAN Controller with SPI Interface .............................................................. 26
    Appendix B: High-Speed CAN Transceiver ........................................................................................... 27
    Appendix C: Quadruple Half-H Drivers .............................................................................................. 28
    Appendix D: PIC16 Datasheet ................................................................................................................. 29
    APPENDIX E: Microchip High-Speed CAN Transceiver Datasheet .......................................................... 30
    APPENDIX F: NXP High-Speed CAN transceiver Datasheet ................................................................. 31
    APPENDIX G: TI 3.3V CAN Transceiver Datasheet .............................................................................. 32
    Appendix H: Servomotor Datasheet ....................................................................................................... 33
    Appendix I: Sample Code 1 ..................................................................................................................... 34
    Appendix J: Sample Code 2 ..................................................................................................................... 40
    Appendix K: CAN Bus Codes and Standards ....................................................................................... 42
    Appendix L: Team Resumes .................................................................................................................... 49
III. List of Figures

Figure 1 - Ku-Band Antennas ........................................................................................................................ 2
Figure 2 - Actuator Module.............................................................................................................................. 3
Figure 3 - Antenna Positioning ....................................................................................................................... 4
Figure 4 - Concept Diagram ........................................................................................................................... 7
Figure 5 - Block Diagram ............................................................................................................................... 8
Figure 6 - System Figure ................................................................................................................................. 9
Figure 7 - Component Sizes .......................................................................................................................... 11
Figure 8 - Hardware Design .......................................................................................................................... Error! Bookmark not defined.
Figure 9 - Positioning Data Flow .................................................................................................................. 12
Figure 10 - Positioning Logic Flow .............................................................................................................. 13
Figure 11 - CAN Bus Data Flow ................................................................................................................... 14
Figure 12 - CAN Bus Logic Flow .................................................................................................................. 15
Figure 13 - Servo Logic Flow ....................................................................................................................... 17
Figure 14 - Butterworth Filter ....................................................................................................................... 18
Figure 15 - Gantt Chart 1 .............................................................................................................................. 20
Figure 16 - Gantt Chart 2 ............................................................................................................................... 20
Figure 17 - Personnel .................................................................................................................................. 22
IV. List of Tables

Table 1 - Requirements and Constraints....................................................................................................... 5
Table 2- Budget ........................................................................................................................................... 21
I. Introduction
   a. Background

   ViaSat specializes in satellites and other forms of digital communication for commercial and government use. An actuator module for mobility antennas gives users many options for vehicular communication with satellites. Since the launch of the first satellite in 1957, satellite and antenna technology has grown exponentially. With these recent advancements, people can experience more convenience in communication from the inside of their own car. ViaSat produces digital communication products that ideally ensure communication from any location. Different projects within ViaSat use mobile antenna systems, and a modular system can help combine these efforts and make production of different products easier. With both government and commercial work, ViaSat needs a readily available module that can function in both environments. With the recent launch of ViaSat-1, the company looks to expand its communications network to a vast array of users.

   As a vehicle moves about, any antenna it carries would need to be moved in order to correctly point to ViaSat-1. This antenna movement would therefore need to be synchronized to vehicle movement. In their present state, these types of mobility antennas are large and cumbersome, and do not fit in with the aesthetic design of modern consumer vehicles. With newer, smaller technology continuously being developed, more convenient systems can be created in order to accommodate all types of situations.

   Different modules for mobility antennas have been proposed and created by other parties. These were created with goals set by each individual person or company. ViaSat has a distinct set of functions they want this module to accomplish. These functions differ from project to project and are unique to their company.

   ViaSat seeks a solution to the unavailability of a small, functional mobile antenna on vehicles. Current systems with similar functions tend to be bulky, have a heavy need for maintenance, and are subject to varying physical forces. The actuator module proposed here shall attempt to eliminate these problems as much as possible. It will communicate with a sensor module and controller via CAN bus. CAN bus is a standard automotive communications protocol already present in many vehicles. Additionally, it has proven to be an effective method of intra-vehicle communications for digital devices. The system will rely on a sensor module that will send sensor data to a master device on the CAN bus, in this case it is a BeagleBone Black embedded system. The BeagleBone will then interpret the sensor data and use the results to control the actuator module over the CAN bus in order to drive the servo motors to move an antenna. In this way, the antenna will be positioned using the sensor data to keep it in line with the satellite at all times.

   b. Purpose

   Government vehicles need the ability to constantly communicate with their allies all around the globe. Commercial vehicles can be upgraded to provide constant communication with a satellite in order to be more functional to the user. The purpose of this actuator module is to accommodate customers in these and other situations. Addressing these needs can be a matter of life or death on the battlefield or in everyday life. Given a vehicular disaster in which communication is limited, a mobility antenna could save a life. There is a need for an efficient
module for a mobile antenna, and the demand will continue to grow as long as technology continues to advance.

**c. Literature Review**

Designs of systems like this have many common components. These components vary anywhere from servos motors to microprocessors. Common designs of the CAN Bus setup is typical in many vehicular applications, such as RVs, military vehicles, as well as commercial cars. Servos are generally used in many motor applications due to their accuracy and ease of use.

Mobile receivers have been used to point to geostationary satellites, just as this project requires. It is a common practice to use servos to move mobile antennas. Ku-Band applications are used in many different satellite applications. The band is convenient for satellite communication with applications.

**i. Prior Work**

**Hi-Sat – Ku-Band Antennas**

The European Space Agency (ESA) and other industries have designed, created, and implemented a Small Aperture Antenna (SAA), a compact antenna designed to be mounted on a vehicle and receive a Ku-Band signal from a geostationary satellite. For this project, they faced a couple challenges that are similar to ones that we will face as well. Like our project, the SAA had to be small enough to fit onto the top of a car without being too intrusive, which it achieved to some degree with the dimensions of a radius of 3.93” and a height of 1.58”. The SAA team also had to consider cost of mass producing their design which they attempted by minimizing electronic components. The SAA in shown in Figure 1 below.

![Figure 1 - Ku-Band Antennas](image)

At ViaSat, the team was able to see an early prototype of a mobile receiver. The device has an actuator module that has the exact same function as our project, but it did not have the size and production cost limitation that we have to comply to.

**ii. Patents**
There are many patents relating to actuator modules, so our team’s research focused on actuator modules that also communicates by a CAN bus. Satellite receiver positioning was also researched because it would be helpful in setting up and programming the servos.

Patent No: US 20110196553 A1
- Publication date: 2011-08-11
- Inventors: Pierre Garon, Neil Garfield Allyn, James Steven Moncynski, Thomas Samuel Martin
- This patent discloses a servo module that controls the direction of a motor of a boat. It is similar to our project because the servo and servo module are connected to the master processor via a CAN bus interface. In our project we will be utilizing a servo module to control the direction of an antenna. The servo module we are designing connects to a CAN master device.
- While this patent is a great hardware example for us, it does not meet the size limitations required of us. (seen below in Figure 2)

![Figure 2 - Actuator Module](image)

Patent No: EP 0246635 A2
- Publication date: 1987-11-25
- Inventor: Markoto Nakayama
- This patent discloses a system for directing the position of a satellites receiver. This coincides with our project because that is the function of the servos connected to our actuator module
• While the rotation method is the same for all satellite receivers, this method does not go into tuning a satellite receiver while it is in motion. (seen below in Figure 3)

![Figure 3 - Antenna Positioning](image)

iii. Codes and Standards

The main focus of our codes and standards is the CAN bus system, which is how the actuator module communicates the master board. There are many different CAN bus standards, and a few been included on this list, in case we find that the standard we choose does not meet our requirements. Further details on the SAE CAN bus standards can be found in Appendix K: CAN Bus Codes and Standards.

ISO 11898-1: CAN Data Link Layer and Physical Signaling. We will be utilizing CAN bus communication in our project.

ISO 11898-2: CAN High-Speed Medium Access Unit. This uses a two-wire balanced signaling scheme. It is the most used physical layer in car powertrain applications and industrial control networks. It relates to our project in that we will be using our unit for vehicular application.


ISO 11898-4: CAN Time-Triggered Communication. This standard defines the time-triggered communication on CAN (TTCAN). It is based on the CAN data link layer protocol providing a system clock for the scheduling of messages. Additional CAN bus standards.

ISO 11898-5: CAN High-Speed Medium Access Unit with Low-Power Mode. Our project boasts a low-power mode.

ISO 11898-6: CAN High-speed medium access unit with selective wake-up functionality. Additional CAN bus standards.

ISO 11992-1: CAN fault-tolerant for truck/trailer communication. Additional CAN bus standards.
SAE J1752-1: This SAE Recommended Practice provides supporting information for the emission and immunity measurement procedures defined in the SAE J1752 series of documents. Our project will deal with significant heat and magnetic noise as created by the servo motors.

SAE J1939-15: 250 kbit/s, Unshielded Twisted Pair (UTP) (reduced layer). The SAE J1939 standard uses a two-wire twisted pair, −11 has a shield around the pair while −15 does not. SAE 1939 defines also application data and is widely used in heavy-duty (truck) and autobus industry as well as in agricultural & construction equipment. Additional CAN bus and noise standards.


SocketCAN: Formerly known as Low Level CAN Framework, it is a set of open source CAN drivers and a networking stack contributed by Volkswagen Research to the Linux kernel. We will be extensively utilizing the software based upon this protocol.

II. Problem Definition

a. Project Requirements

The functional and physical requirements of the project require the final product to be fully functional for uses in many different vehicular applications. The fundamental purpose of the product is to provide a generic CAN bus actuator module for mobility antennas. The final product is expected to provide a module that allows antennas to keep in constant contact with a satellite, and to provide flawless communication between the two. It must perform well in circumstances ranging from a daily commute, all the way to being in the middle of an attack at war. The system must be durable enough to withstand nearly any interference, whether physical or electrical. The reliability of the product is one of, if not, the most important feature because of the different situations in which it will be used. The device must be easily serviceable in the event of necessary software or hardware upgrades.

The actuator module is required to be created with modular circuitry, and use embedded logic to control the motors. A rugged connector and rugged housing are imperative for the variety of conditions the product will encounter. SocketCAN based software is required due to the ease of integration with vehicles and the system being created. The target dimension requirements of the final product are 1.5” X 3” X 0.25”. The requirements and constraints can be seen together on Table 1 below.

<table>
<thead>
<tr>
<th>Requirements and Constraints</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Master Hardware</td>
<td>TI OMAP Processor</td>
</tr>
<tr>
<td>Master Software</td>
<td>CAN Master Software</td>
</tr>
<tr>
<td>Bus Standard</td>
<td>CAN Bus</td>
</tr>
<tr>
<td>Housing</td>
<td>Rugged, Compact</td>
</tr>
<tr>
<td>Frequency Restriction</td>
<td>100Hz, low resource utilization</td>
</tr>
<tr>
<td>Size</td>
<td>1.5” X 3” X 0.25”</td>
</tr>
<tr>
<td>Cost</td>
<td>&lt;$400</td>
</tr>
</tbody>
</table>
b. Constraints

Various monetary, personnel, and design constraints limit the project. The final product needs to be manufactured for less than $400. Each member is interacting, trying to find the best, most cost efficient way to create the final product.

In terms of personnel, there are key items and theories that have not been seen by members of the team. Thermal noise is a factor that was not considered initially, but has been thoroughly looked into. Overheating of the PCB was not taken into account when putting together the original design. Through research and talking with experts in the subject, the group has decided to use heat sinks as a solution to the problem.

PCB fabrication is a vital component to the final product, and is being looked at by the members of the team. Members have been chosen to specialize in the design of the PCB and production of the board itself.

Given the experiences of each member, the team is able to work through the complexity of the project. Every aspect of the final product is something that each member has seen or heard of before and, with some research, can be solved. Team CBAM is able to complete the design meeting all necessary requirements. A time frame of nine months was given to complete the project, and the team believes that final deliverable will be easily available by the end of this period.

III. Design Specification

a. Design Overview and Deliverables

This project will deliver an actuator module with the ability to accurately and constantly reposition itself in order to face a satellite. The design will be constructed in a way that will allow it to be a modular system for future projects. The location of the antenna versus the satellite will be constantly polled, and the system will adjust itself accordingly. By using a CAN bus and transceiver, the module will work well for vehicular applications, seeing as CAN hardware is a widely used protocol for vehicular application. The concept diagram is seen below in Figure 4.
Each component of the system is shown in the figure above. The individual parts are described below.

- **Beaglebone Black** - This microcontroller will simulate inputs for the module. By using this piece of hardware, demos can be created in order to simulate real world situations.
- **CAN Bus** - The CAN bus is a common means of information transmission of data for vehicular applications. The CAN bus will act as the main communication between the sensor input and the module.
- **MCP2551** - The CAN transceiver will receive the input data and communicate that with the module.
- **PIC16** - The PIC16 is the microcontroller chosen for the embedded logic on the system. It will be programmed to perform the necessary functions needed for the module to perform as it is supposed to.
- **L293DNE** - This servo driver will allow the software from the embedded logic to properly communicate with the hardware used in the system.
- **RMCS-2257** - The two servo motors will be used as the driving force to the entire module. The position of the antenna will be constantly polled, and the servo motors will adjust themselves according to the given data.
• Module Power - The entire module will be powered by an external 5V DC power source.
• Servo Power - The servo motors will be powered by an auxiliary power source.

The block diagram shown in Figure 5 lays out the different components of the module.

**Figure 5 - Block Diagram**

b. Functional Specifications

The final deliverable, once finished, will consist of an actuator that will be able to point at any target (usually the satellite) in any pose conditions of the platform where it will be planted. For example, a car having the actuator on the roof would have to adjust the rotations of the servos that hold the antenna continuously because the road is not always flat and, even if it was, the car would also be performing yaw rotations. The final deliverable will run at a frequency of about 100Hz.

c. Physical Specifications

An important aspect of this project is to have a final small and rugged deliverable. The size of the actuator module, including housing, will be no bigger than 3” x 1.5” x 0.25”. This module will be connected to a BeagleBone Board through a Can Bus module and the rest of the sensors (in case we want to implement them instead of simulating the inputs) will be connected to the BeagleBone as well.

All the modules will be separated enough to avoid noise interferences between them, plus some capacitors will be connected to the inputs of each module to get a clean ‘in’ signal

IV. Design Results

a. System Design

The overall system will consist of three subsystems: the positioning subsystem, the CAN bus subsystem, and the servo subsystem. Together, these subsystems will allow the CAN bus actuator module to first read sensor data via outside sensors connected to a shared CAN bus. These sensors are outside of the scope of this design, but can include positioning sensors such as gyroscopes, GPS, and accelerometers. These sensors give data concerning the orientation of a vehicle to which an antenna is attached. Utilizing this sensor data and the current stored position of the mobile antenna, the CAN bus actuator module will calculate the new required
position of the antenna in order to keep it aligned in a fixed direction, such as towards a satellite, despite any movements to the underlying structure (in this case a vehicle) to which it is attached. The actuator module will then utilize this new calculated position in order to calculate the servomotor movements necessary to reposition the antenna from its current azimuth and elevation to the required azimuth and elevation. The servos will then be driven for the required times in the required directions, thus repositioning the antenna as required. See Figure 6 for a system figure.

The overall system requires interaction of the various subsystems. In order for the positioning subsystem to gather sensor data, it will require a working CAN bus link to any sensor modules. Once the positioning subsystem has calculated the required servo movements necessary to keep the antenna aligned according to the sensor data, this movement information will be transmitted to the servo subsystem via the CAN bus link. This CAN bus link will be implemented in the CAN bus subsystem. In other words, the positioning and servo subsystems require the CAN bus subsystem in order to communicate between each other. In addition, the positioning subsystem relies on its own CAN bus link to be able to gather any sensor data.

The components contained within the entire CAN bus actuator module consist of the following: A PIC16 microcontroller for any embedded logic, H-Bridges for providing adequate voltage to the servomotors, a CAN bus transceiver in order to interface to the CAN bus, and a CAN bus controller that provides a link between the transceiver and any embedded logic (the PIC). Additionally, a BeagleBone Black embedded system will be used as the master module on the CAN bus. Servomotors also comprise additional outside components. These components were selected in order to support the customer requirements and constraints. In its request for proposal (RFP), ViaSat required the sensor module (outside of this project) and actuator module (this project) to be interfaced via CAN bus to each other and to a BeagleBone Black embedded system. This BeagleBone Black would be the master device on the CAN bus, receiving and interpreting the data from the sensor module before using it to orient the antenna via the actuator module. The BeagleBone Black was specified for this purpose due to its low cost and easy to use nature. It is a powerful embedded system, easily suited to the task of incorporating positional sensor data into a coordinate system and calculating the necessary mechanical adjustments. Furthermore, there exists for it an open source CAN bus protocol, SocketCAN, that enables it to seamlessly act as a CAN bus node. While not strictly part of the CAN bus actuator module itself, it is nonetheless an integral part of its operation and will thus be included in the design.
The actuator module itself had additional constraints. Primarily, it needed to be able to interface to a CAN bus and to be able to drive a pair of servomotors (which would be attached to an antenna). Furthermore, it needed to fit within precise dimensions once fabricated. For these reasons were the other components chosen. The PIC16, when interfaced with the CAN bus controller and CAN bus transceiver, allows for CAN bus communication. The PIC16 is also able to utilize embedded logic to translate movement commands received via the CAN bus from the BeagleBone Black to drive the servomotors. The servomotors are driven by the H-Bridges, the H-Bridges being controlled, as noted earlier, by the PIC16. The use of the PIC16 and Microchip branded CAN bus controller and transceiver was decided upon due to ease of use and interoperability, as well as small physical size. The small size of these three components, especially when fabricated as surface mount components, allows the physical dimension constraints as outlined in the RFP to be handily met. The ease of use noted above comes from the extensive prior experiences of the team utilizing PIC microcontrollers in digital designs. The interoperability mentioned stems from the fact that all three chips are manufactured by Microchip, and are thus designed for easy integration with one another. Furthermore, Microchip provides a myriad of reference designs utilizing their CAN bus controller and transceiver in conjunction with PIC microcontrollers.

The servomotors, as mentioned, are not specifically part of the CAN bus actuator module. However, much like the BeagleBone, they are an integral part of the project as a whole. They were selected for their properties of high speed and high rotational resolution. They will have enough torque to quickly move an antenna, as well as a high enough resolution to enable satellite communications. They will be driven via logic from the PIC16, pulling power through H-Bridges, enabling them to be driven on much higher voltages than the logical voltages output by the PIC16. Given their necessity to the project, they will be included in the servo subsystem.

There was a single major tradeoff in design done in this project. This was required by the physical dimensions constraint. It would have been simplest to use either the BeagleBone Black or an Arduino processor and appropriate break-out boards to implement CAN bus communication, to retrieve and process sensor data, and to directly interface with the servomotors. Either would be certainly more than capable of all these tasks. However, these solutions are simply too large to fit into the required dimensions. Thus, the BeagleBone was implemented outside of the CAN bus actuator module design, with the internal logic and CAN bus communication being handled by multiple, although much, much smaller, chips. In this case, simplicity was sacrificed for physical size. However, this added complexity of using multiple chips internal to the module for CAN bus communication and servomotor logic is mitigated through the use of Microchip manufactured components. By utilizing a single manufacturer, the components have guaranteed interoperability. Not only that, but the team also has extensive experience utilizing PIC microcontrollers, as manufactured by Microchip. By utilizing this knowledge along with the extensive reference designs provided by Microchip for PIC based CAN bus nodes, this design tradeoff is easily overcome. See Figure 7 for component sizes.
Potential shortcomings and risks of the project design include insufficient servomotor accuracy response, the translation of positional data into antenna positional coordinates, insufficient heat management, and electromagnetic noise caused by the servomotors. The first risk is the use of untested servomotors. With a myriad of servomotors available on the market, the team chose ones that seemed to be of practical size, strength, and resolution, while not being too expensive. However, only during the testing and debugging phase of the project will it become known precisely how accurate the servomotor response is, and whether it will be useful in a satellite based system or not. It is important to keep in mind, though, that the servomotors are not strictly part of the CAN bus actuator module design. As its name implies, it will be a modular system, capable of interfacing with any servomotor with minimal modification to system logic. Thus, if the selected servomotors prove insufficient for the final project design, implementing different servomotors will be an insignificant hurdle.

The second potential risk is the software translation of sensor data into positional coordinates for an antenna, in this case azimuth and elevation. This will be a purely software based solution, taking place on the BeagleBone Black. The team currently has minimal experience programming this device.
Finally, there is the risk of insufficient heat management and electromagnetic noise as caused by the servomotors and their drivers. With the H-Bridges providing up to 30 Volts to the servomotors while still being inside the module casing, they have the potential of creating a large amount of heat. Although this is not something anyone on the team has ever had to design for, the design will include both on-chip fin heat sinks as well as copper plate heat sinks integrated into the PCB. Along with the heat, any design utilizing motors runs the risk of electromagnetic noise. This will be mitigated utilizing capacitive coupling of all power sources and grounds. Additionally, if that does not prove sufficient, filters can be implemented between the module outputs and the servomotors.

In conclusion, the team has designed a CAN bus actuator module that is able to gather data from sensors attached to a CAN bus. It is then able to process that data and position an antenna accordingly, by utilizing servomotors. The module used for the servomotor control will be within the dimensions as specified in the RFP. Furthermore, the data processing utilizing the sensor data will be implemented on a BeagleBone Black, again as specified in the RFP. The BeagleBone Black will be configured to communicate with both the servomotor driver module (the actuator module) and the various sensors via CAN bus and the SocketCAN protocol, necessitated in the RFP. Accordingly, all project requirements as outlined in the RFP will be met.

b. Positioning Subsystem Design

This subsystem consists solely of the BeagleBone Black and the software embedded thereupon. The positioning subsystem will incorporate the current antenna position (azimuth and elevation) and the sensor data read off of the CAN bus. This will be accomplished by utilizing the BeagleBone Black. The BeagleBone Black will use its own CAN bus link to communicate with any sensor systems attached to the CAN bus. This link will be implemented in software. It will then use this data to calculate how to position the antenna for satellite communications. This calculation will be done with an algorithm for rotating coordinates. The BeagleBone will then use this positioning information to determine how to move the servomotors to align the antenna with the desired position. It will then transmit this movement information to the servo subsystem via the CAN bus subsystem. The BeagleBone Black will interface with the CAN bus via the SocketCAN protocol, an open source CAN bus protocol developed for Linux based systems such as the BeagleBone Black. See Figure 8 for a block diagram and Figure 9 for a software flowchart.
As this subsystem is primarily software based, the main design results so far consist of the algorithm for updating a position in 3D space, as seen in Appendix I: Sample Code 1. The antenna is started pointing at a given altitude (vertical direction) and azimuth (rotational direction). Utilizing simulated sensor data, the algorithm is able to calculate the new altitude and azimuth the antenna needs to be pointed at relative to its current position. It is then able to incrementally change the current antenna position until it matches the required position. This entire process takes fractions of a second, enabling the antenna position to be changed in real-time according to changing sensor data (positional information). This subsystem is able to be demonstrated via software simulation. Additionally, the BeagleBone Black component is on hand and will have a functional (“Hello World”-type) demonstration.

c. CAN Bus Subsystem Design
This subsystem consists of three components: a CAN bus transceiver that translates digital logic into CAN bus logic, a CAN bus controller that is able to send and receive messages by utilizing the transceiver, and a PIC16 microprocessor containing all of the embedded logic required to enable communications. The CAN bus subsystem will allow the team’s CAN bus actuator module to actually function on a CAN bus. It is crucial for internal communication. Inter-module CAN bus communication is necessary for positioning and servo subsystem interaction. External communication is necessary for the positioning subsystem to access the available sensor data as well, although this will be implemented by utilizing the SocketCAN protocol on the BeagleBone Black. As for the actuator module itself, the CAN bus subsystem will be implemented there through the use of a CAN bus transceiver connected to a CAN bus controller. The controller is then connected to a microcontroller. In this case, the microcontroller is a PIC16, and the transceiver and controller are both Microchip produced components. This allows for easy integration of the subsystem by utilizing ready-made Microchip reference designs. See Figure 10 and Figure 11 below.
The design of this subsystem in nearly functionally complete, including having all components on hand. The wiring diagram can be seen below in Figure 12. The required embedded logic, as programmed into the PIC16, can be seen in Appendix D: PIC16 Datasheet. This program will allow the PIC microcontroller to interface with and communicate over the CAN bus by utilizing the CAN Controller (which in turn interfaces with the CAN Transceiver; the Transceiver, however, requires no embedded logic). This hardware design is based on Microchip and MikroC reference designs. The embedded logic was developed using MikroC due to the extensive C libraries available for PIC devices, including those for CAN bus operation.
For demonstration purposes, the team will have on hand a working example of a CAN Bus Subsystem. It will consist of two copies of the subsystem, linked together over the CAN bus. The 1\textsuperscript{st} node will read a button press (the button being connected to the PIC) and transmit the button press across the CAN bus. The 2\textsuperscript{nd} node will read this message and light an LED in response. If the button is pressed a second time, the LED will turn off. This demonstrates a working CAN Bus Subsystem. It will demonstrate working hardware and software, to include both transmitting and receiving a message on a CAN bus.

\subsection*{d. Servo Subsystem Design}

This subsystem includes the PIC16, any attached servomotors, and the auxiliary power source for the servomotors. The servo subsystem will enable the CAN bus actuator module to serve as an actuator. It will serve as the means to move an attached antenna. This subsystem consists of embedded logic used to drive the servomotors. In this case, the embedded logic is provided by the PIC16. The PIC16 will receive movement commands from the positioning subsystem via the CAN bus subsystem. It will execute these movement commands through control of two servomotors, one serving as azimuth and the other as elevation. The servomotors will be driven using an auxiliary power source, to include an H-Bridge for each, allowing for up to 30 Volts for each servomotor. The PIC16 will ensure accurate servomotor control through rotational encoding output from each servomotor. This will allow for an incredibly high degree of positional accuracy for each servomotor. See Figure 13 and Figure 14 for the software flowchart and subsystem block diagram.
The servomotors will be controlled through either SPI or I2C, depending on whichever ends up being easier to implement into the final design. They are capable of either. The team has the servomotors on hand and is able to demonstrate their use.
V. Design Plan

a. Stage 1 – Research

The Research Phase consisted of necessary research required to fully understand and comprehend the project and its scope. For the most part, the team has completed this phase of the project, but to review, the team has:

- Researched CAN bus architecture, to include SocketCAN software, a project requirement. This will be done in order to understand the protocol with which our device must communicate.
- Researched CAN bus interface with microcontroller. This will enable us to allow our actuator module to respond to messages sent to it over the CAN bus, a project requirement.
- Research similar projects in order to develop a greater understanding of the overall project scope, as well as narrow down design choices to those that have been shown to work in other projects.

b. Stage 2 – Design

The Design Phase consists of utilizing the knowledge gained in the research phase and applying it to the proposed design in order to come up with a working solution. So far the team has completed the following design goals:

- Produced graphical representation of module, both for the hardware and the software. The hardware will consist of a wiring diagram, the software of a software flowchart.
- Finalized hardware choices
- Purchased and obtained all required hardware.
- Created a breadboard prototype of the actuator module.
- Programed a simple program for the actuator module to demonstrate

Thermal noise is an issue that is present in almost all electrical systems. In order to reduce the noise in the actuator module, a low-pass Butterworth filter will be designed and implemented, similar to the one shown in Figure 15. Resistor and capacitor values will be chosen in order to minimize the thermal noise that may cause problems in the final design.

![Butterworth Filter](Figure 15 - Butterworth Filter)
c. Stage 3 – Prototype Construction

The construction of the module will go through different iterations before becoming the final deliverable. After completing a circuit design, the module will be constructed on a breadboard. The breadboard will consist of each individual piece needed in the final product. This breadboard prototype will help troubleshoot any problems that may arise before spending money to construct an actual model. By making this prototype, all unnecessary components or flawed designs will be identified and fixed.

After the breadboard prototype works as planned, the next iteration of the project will consist of manufacturing a board. By submitting a design schematic in early February to AP Circuits, the PCB for the module will be created. This board will then be sent to BEST (Business Electronic Soldering Technologies). This company will handle the professional soldering necessary to attach the necessary components to the PCB.

d. Stage 4 – Testing

Testing will initially consist of simulated sensor inputs generated by the BeagleBone Black. The sensors and signals will be tested in order to ensure that they will support the functions for the job. This will ensure the design is working as intended without worry about any real world systems integration issues. This will occur before the Critical Design Review, in order to identify any issues in the proposed final hardware and software design. This will also allow for the production of the Final Report and Presentation. This will be completed prior to the final PCB fabrication.

Upon final PCB fabrication, the design will be housed within a ruggedized container. Extensive stress testing will begin at this point, so as to ensure the excessive heat and noise risks were sufficiently mitigated. If this is not the case, the housing will need to be redesigned, or additional heat and noise reduction techniques will need to be implemented. The finalized ruggedized housing and connectors, as required by the request for proposal, will be complete before project delivery.

Project delivery will include a laser demonstration, whereupon the actuator module will be placed upon a moving platform and will continuously shine a laser on a specified point, simulating the continuous directing of an antenna towards a relatively fixed satellite. If necessary, simulated input data will be used. However, an actual sensor module will be used, if possible.

e. Stage 5 – Documentation

The Portfolio and Design Documentation Binder will be completed by 10 Dec 2013. It will document that the course outcomes have each been achieved. Additionally, it will contain all of the team’s design specifications, to include both hardware and software. It will include all the wiring diagrams and schematics, software flowcharts, and block diagrams. It will effectively allow for the project to be reproduced through documentation alone.
f. Schedule

The group received the original Concept Project Proposal on Thursday, September 12\textsuperscript{th}, 2013. The parts were ordered on Thursday, November 7\textsuperscript{th}, 2013. By the preliminary design review (PDR) on Tuesday, November 26\textsuperscript{th}, the group will have a functional demo of the product up and running. The group is confident that the actuator module will be prepared for a critical design review (CDR) by early March. The final project is anticipated for completion by mid-April. The technical specifications of the project can be seen below in the Schedule and Gantt Chart of Figure 16 and Figure 17.

<table>
<thead>
<tr>
<th>Task</th>
<th>Duration</th>
<th>Start Date</th>
<th>End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project proposal draft</td>
<td>9 days</td>
<td>Thu 9/12/13</td>
<td>Tue 9/24/13</td>
</tr>
<tr>
<td>Produce graphical representation of module</td>
<td>9 days</td>
<td>Thu 9/12/13</td>
<td>Tue 9/24/13</td>
</tr>
<tr>
<td>Research CAN bus architecture</td>
<td>3 days</td>
<td>Fri 9/20/13</td>
<td>Tue 9/24/13</td>
</tr>
<tr>
<td>Research CAN bus interface with microchip</td>
<td>5 days</td>
<td>Fri 9/20/13</td>
<td>Thu 9/26/13</td>
</tr>
<tr>
<td>Research similar projects</td>
<td>7 days</td>
<td>Fri 9/20/13</td>
<td>Mon 9/30/13</td>
</tr>
<tr>
<td>Finalize hardware choice</td>
<td>20 days</td>
<td>Tue 10/1/13</td>
<td>Mon 10/28/13</td>
</tr>
<tr>
<td>Complete wiring of the breadboard</td>
<td>28 days</td>
<td>Tue 10/1/13</td>
<td>Thu 11/7/13</td>
</tr>
<tr>
<td>Finalize initial software design for testing</td>
<td>28 days</td>
<td>Tue 10/1/13</td>
<td>Thu 11/7/13</td>
</tr>
<tr>
<td>Develop example test software</td>
<td>24 days</td>
<td>Thu 10/24/13</td>
<td>Tue 11/26/13</td>
</tr>
<tr>
<td>Preliminary design review</td>
<td>51 days</td>
<td>Tue 9/17/13</td>
<td>Tue 11/26/13</td>
</tr>
<tr>
<td>Breadboard fabrication</td>
<td>23 days</td>
<td>Tue 11/12/13</td>
<td>Thu 12/12/13</td>
</tr>
<tr>
<td>Critical design review</td>
<td>73 days</td>
<td>Tue 11/26/13</td>
<td>Thu 3/6/14</td>
</tr>
<tr>
<td>Finalize housing design for hardware</td>
<td>65 days</td>
<td>Mon 11/11/13</td>
<td>Fri 2/7/14</td>
</tr>
<tr>
<td>PCB Fabrication</td>
<td>31 days</td>
<td>Tue 1/28/14</td>
<td>Tue 3/11/14</td>
</tr>
<tr>
<td>Finalize wiring of all components</td>
<td>26 days</td>
<td>Thu 3/6/14</td>
<td>Thu 4/10/14</td>
</tr>
<tr>
<td>Finalize software integration</td>
<td>26 days</td>
<td>Thu 3/6/14</td>
<td>Thu 4/10/14</td>
</tr>
<tr>
<td>Enclose components in housing</td>
<td>26 days</td>
<td>Thu 3/6/14</td>
<td>Thu 4/10/14</td>
</tr>
<tr>
<td>Final project completed</td>
<td>31 days</td>
<td>Thu 3/6/14</td>
<td>Thu 4/17/14</td>
</tr>
</tbody>
</table>

Figure 16 - Gantt Chart 1

Figure 17 - Gantt Chart 2
The initial research that has been done has given team CBAM insight into what must be done in the coming months. Through extensive research, the team has come up with a solid solution to the problem presented to them. With most of the research having been completed, the team has moved onto obtaining parts, and creating a prototype.

Approximately 75% of the hardware components required for the project have been delivered. With these parts, team CBAM can begin construction of a functional prototype. The PDR requires working software, so the software REA, Gonzalo, has been working on a functional program to present.

With the arrival of the components, the team will be able to construct a working prototype of the project. A breadboard prototype will be completed by December 12th, 2013. This breadboard prototype will comprise of all necessary components, and will function as a piece of test equipment to troubleshoot possible future hardware problems. Once all the problems have been worked out, PCB fabrication will follow the breadboard prototype. PCB fabrication is scheduled for completion by March 11th, 2014.

The members of team CBAM will spend the remainder of March finalizing components and software issues. A housing structure will be set up in order to enclose the project, and by April 17th, 2014, a final deliverable will be available.

g. Budget

ViaSat requests that the product can eventually be produced for less than $400 per unit. The current budget also allows for faulty and broken components, as well as miscellaneous items. The budget is clearly laid out in Table 2.

<table>
<thead>
<tr>
<th>Component</th>
<th>Part/Material</th>
<th>Supplier</th>
<th>Cost</th>
<th>Quantity</th>
<th>Subtotal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microprocessor</td>
<td>PIC16F1825-I/P</td>
<td>Newark</td>
<td>$1.42</td>
<td>3</td>
<td>$4.26</td>
</tr>
<tr>
<td>CAN bus transceiver</td>
<td>MCP2551-E/P</td>
<td>Newark</td>
<td>$1.18</td>
<td>3</td>
<td>$3.54</td>
</tr>
<tr>
<td>CAN bus controller</td>
<td>MCP2515-I/P</td>
<td>Newark</td>
<td>$1.82</td>
<td>3</td>
<td>$5.46</td>
</tr>
<tr>
<td>CAN bus master device</td>
<td>BEAGLEBONE BLACK</td>
<td>Newark</td>
<td>$45.00</td>
<td>1</td>
<td>$45.00</td>
</tr>
<tr>
<td>H bridge</td>
<td>L293DNE</td>
<td>Newark</td>
<td>$2.09</td>
<td>6</td>
<td>$12.54</td>
</tr>
<tr>
<td>PIC programmer</td>
<td>PG164130</td>
<td>Newark</td>
<td>$44.95</td>
<td>1</td>
<td>$44.95</td>
</tr>
<tr>
<td>PIC CAN demo board</td>
<td>MCP2515DM-BM</td>
<td>Newark</td>
<td>$55.00</td>
<td>1</td>
<td>$55.00</td>
</tr>
<tr>
<td>Heat Sink</td>
<td>ICK 14/16 L</td>
<td>Newark</td>
<td>$0.93</td>
<td>12</td>
<td>$11.18</td>
</tr>
<tr>
<td>CAN interface port</td>
<td></td>
<td>Newark</td>
<td>$15.00</td>
<td>1</td>
<td>$15.00</td>
</tr>
<tr>
<td>Power interface port</td>
<td></td>
<td>Newark</td>
<td>$15.00</td>
<td>2</td>
<td>$30.00</td>
</tr>
<tr>
<td>Ruggedized</td>
<td></td>
<td>Newark</td>
<td>$75.00</td>
<td>1</td>
<td>$75.00</td>
</tr>
</tbody>
</table>
### h. Personnel

Team CBAM consists of Shane Fontaine, Christopher Anderson, Gonzalo Albaladejo, and Ryan Maliszewski. Each member is a senior Electrical Engineering student at the University of San Diego. All members bring their own experiences from previous assignments in industry, military, or other countries. The project is very important to each member of the team, and will be a passion for the next nine months. See Figure 18 for the organizational chart.
i. Shane Fontaine

Shane is the project manager of the CBAM. He is the liaison with the customer, and will communicate directly with them until the completion of the project. Through his internship at The Boeing Company, Shane brings a different perspective to the project that may not be seen by his peers. The industry experience allowed him to work with satellites, and other equipment associated with them. The CBAM is directly related to a satellite, thus making easier to see the bigger picture. In terms of the specifics of the project, Shane is in charge of the CAN bus interface hardware, transceiver, and the servo software. His résumé can be seen in Appendix L: Team Resumes.

ii. Christopher Anderson

Chris is the Hardware Responsible Engineering Authority (REA) of the actuator module project. He is also directly responsible for the driver hardware design and the servo hardware design. He will assist with the driver software, the module power design, and the final PCB layout design. Currently in his senior year at the University of San Diego as an Electrical Engineer, Chris has 3 years of student design and laboratory experience, including circuits and microcontrollers. This provides a strong grasp of microcontroller and circuit design, to include device interfacing on both a hardware and software level. He also brings to the table nearly 10 years of experience as an active duty Marine, providing the team with organizational and leadership skills developed there, as well as a focus on mission accomplishment. His résumé can be seen in Appendix L: Team Resumes.

iii. Gonzalo Albaladejo

Gonzalo is the Software REA and currently studies Industrial Engineering at the University of San Diego as an exchange student from the Universidad Pontificia de Comillas. He has studied C programming on his first year of Engineering. In his second year he took the first 2 classes (out of 4) of the Diploma in Mobile Robotic Systems that his university offers, in which he learned how to program embedded systems and he got in touch with the hardware world that is behind the software programming part. His two projects were to program a robot that would play a hide and seek game against other robots and a robot that would play basketball. In his third year, he took the Digital Electric Systems class, in which he continued programming embedded logic. His project this time was to design a completely automated house. In his fourth year, he studied at San Diego State University, where he took several robotic classes: Embedded System Programming, Robotics Math Programming & Control, Intelligent Systems and Control. The last year has been really helpful to his knowledge of algorithms (genetic algorithms, neural networks).

His areas of knowledge that will contribute to this project include: Electrical engineering, Programming in C and C++, Electronics and control systems, Mechanics and Civil Engineering, Artificial intelligence applied to mobile robotics. His résumé can be seen in Appendix L: Team Resumes.

iv. Ryan Mailszewski
Ryan is in charge of the Research and Design of the project. He also has the responsibility of managing the budget of the device. A former employee of Advantage Law Firm, a small patent firm, Ryan has received industry experience applying his Electrical Engineering knowledge. This industry experience has provided him the opportunity to work on researching prior art and writing persuasive arguments supporting new inventions. He is qualified to research similar project and make convincing arguments on how the project differs from them. Ryan has both hardware and programming experience developed throughout the electrical engineering course he has taken at the University of San Diego. His résumé can be seen in Appendix L: Team Resumes.
VI. References


PR, N. (2013, June 19). ViaSat Ka-band In-flight System Will Be New Option for Boeing Commercial Aircraft. PR Newswire US.


VII. Appendices

Appendix A: Stand-Alone CAN Controller with SPI Interface

Appendix B: High-Speed CAN Transceiver

Appendix C: Quadruple Half-H Drivers

Appendix D: PIC16 Datasheet

APPENDIX E: Microchip High-Speed CAN Transceiver Datasheet

APPENDIX F: NXP High-Speed CAN transceiver Datasheet


---

**TJA1049**

High-speed CAN transceiver with Standby mode

Rev. 3 — 16 September 2013

Product data sheet

1. **General description**

The TJA1049 high-speed CAN transceiver provides an interface between a Controller Area Network (CAN) protocol controller and the physical two-wire CAN bus. The transceiver is designed for high-speed (up to 1 Mbit/s) CAN applications in the automotive industry, supplying the differential transmit and receive capability to (a microcontroller with) a CAN protocol controller.

The TJA1049 belongs to the third generation of high-speed CAN transceivers from NXP Semiconductors, offering significant improvements over first- and second-generation devices such as the TJA1040. It offers improved ElectroMagnetic Compatibility (EMC) and ElectroStatic Discharge (ESD) performance, and also features:

- Ideal passive behavior to the CAN bus when the supply voltage is off
- A very low-current Standby mode with bus wake-up capability
- TJA1049T/3 and TJA1049TK/3 can be interfaced directly to microcontrollers with supply voltages from 3 V to 5 V

These features make the TJA1049 an excellent choice for all types of HS-CAN networks, in nodes that require a low-power mode with wake-up capability via the CAN bus.

2. **Features and benefits**

2.1 **General**

- Fully ISO 11898-2 and ISO 11898-5 compliant
- Suitable for 12 V and 24 V systems
- Low ElectroMagnetic Emission (EME) and high ElectroMagnetic Immunity (EMI)
- VCC input on TJA1049T/3 and TJA1049TK/3 allows for direct interfacing with 3 V to 5 V microcontrollers
- SPLIT voltage output on TJA1049T and TJA1049TK for stabilizing recessive bus level
- Both variants available in SO8 and HVSON8 packages
- Leadless HVSON8 package (3.0 mm × 3.0 mm) with improved Automated Optical Inspection (AOI) capability
- Dark green product (halogen free and Restriction of Hazardous Substances (RoHS) compliant)

2.2 **Low-power management**

- Very low-current Standby mode with host and bus wake-up capability
- Functional behavior predictable under all supply conditions
- Transceiver disengages from the bus when not powered up (zero load)
APPENDIX G: TI 3.3V CAN Transceiver Datasheet

Appendix H: Servomotor Datasheet

**RHINO MOTION CONTROLS**
RMCS-220X
High-Torque Encoder DC Servo Motor and Driver
UART, I2C, PPM and Analog input interface (Max. 15Vdc and 7A)

Installation Manual and Datasheet

http://www.rhinomc.com
http://www.rhinomotioncontrols.com
Appendix I: Sample Code 1

```c
#include <STM32F0xx.h>
#include <stdbool.h>
#include <math.h>
#define LED_NUM 2
#define PC
#define Pi 3.14159
#define L 3

volatile uint32_t cycleTime = 1000;
volatile uint32_t msTicks;

const unsigned long led_mask[] = {1UL << 8, 1UL << 9};

void delay(uint32_t ms);
void LED_Init (void);
void BTN_Init (void);
void LED_On (uint32_t num);
void LED_Off (uint32_t num);
uint32_t BTN_Get(void);
uint32_t getKey(void);
int state;

typedef struct{
    float elevation;
    float azimuth;
}satellite;

typedef struct{
    float elevation;
    float azimuth;
}antena;

typedef struct{
    float rotx, roty, rotz;
}car;

satellite s;
antena actual, desired;
car c;
double Mat[L][L];
satellite UpdateSatellite (void){
```
satellite aux;
    aux.azimuth=\frac{\pi}{2};
    aux.elevation=\frac{\pi}{4};
    return aux;
}

antena UpdateAntena (void){
    antena aux;
    aux.azimuth=0;
    aux.elevation=0;
    return aux;
}

car UpdateCar (void){
    car aux;
    aux.rotx=\frac{\pi}{100};
    aux.roty=\frac{\pi}{100};
    aux.rotx=0;
    return aux;
}

void Mult (double A[][][L], double B[][][L], double res[][][L], int n, int m, int m2){
    int i, j, k;
    for (i=0;i<n;i++){
        for (j=0;j<m2;j++){
            for (k=0;res[i][j]=0;k<m;k++){
                res[i][j]+=A[i][k]*B[k][j];
            }
        }
    }
}

void IniRotationM (double M[][][L], double rotx, double roty, double rotz){
    M[0][0]=\cos(\text{rotx})\cos(\text{roty});
    M[0][1]=\cos(\text{rotx})\sin(\text{rotz}) + \sin(\text{rotx})\sin(\text{roty})\cos(\text{rotz});
    M[0][2]=\sin(\text{rotx})\sin(\text{rotz}) - \cos(\text{rotx})\sin(\text{roty})\cos(\text{rotz});
    M[1][0]=\cos(\text{roty})\sin(\text{rotz});
    M[1][1]=\cos(\text{rotx})\cos(\text{rotz}) - \sin(\text{rotx})\sin(\text{roty})\sin(\text{rotz});
    M[1][2]=\sin(\text{rotx})\cos(\text{rotz}) - \cos(\text{rotx})\sin(\text{roty})\sin(\text{rotz});
    M[2][0]=\sin(\text{roty});
    M[2][1]=\sin(\text{rotx})\cos(\text{roty});
    M[2][2]=\cos(\text{rotx})\cos(\text{roty});
void UpdateRotationM (double M[][L], double rotx, double roty, double rotz){
    double aux[L][L], aux2[L][L];
    Copy(M,aux2,3,3);
    IniRotationM(aux,rotx,roty,rotz);
    Mult(aux2,aux,M,3,3,3);
}

bool Check (void){
    if (actual.azimuth==desired.azimuth && actual.elevation==desired.elevation){
        return true;
    }
    else{
        return false;
    }
}

antena FromGlobalToRelative (){}
    antena aux;
    double x[3][3], y[3][3];
    x[0][0]=cos(s.elevation)*cos(s.azimuth);
    x[1][0]=cos(s.elevation)*sin(s.azimuth);
    x[2][0]=sin(s.elevation);
    /*
    y[0]=cos(c.rotx)*(x[1]*cos(c.roty)*sin(c.rotz)-
    x[0]*sin(c.rotx)*sin(c.rotz)+x[2]*sin(c.rotx)*sin(c.rotz)+x[0]*cos(c.roty)*cos(c.rotz)+x[1]*sin(c.roty)*cos(c.rotz));
    y[1]=cos(c.rotx)*(x[1]*sin(c.roty)*cos(c.rotz)-
    x[0]*cos(c.rotx)*sin(c.rotz)+x[2]*sin(c.rotx)*sin(c.rotz)-x[0]*cos(c.roty)*sin(c.rotz)-
    x[1]*cos(c.roty)*sin(c.rotz)-
    x[2]*sin(c.roty)+sin(c.rotx)*x[0]*sin(c.roty)-x[1]*cos(c.roty));
    */
    y[0]=cos(c.rotx)*(-
    x[2]*sin(c.roty)*cos(c.rotz)+x[1]*sin(c.roty)+sin(c.rotx)*x[2]*sin(c.roty)+x[1]*sin(c.roty)*cos(c.rotz))+x[0]*cos(c.roty)*cos(c.rotz);  
    y[1]=cos(c.rotx)*(x[1]*cos(c.roty)*sin(c.rotz)+x[2]*sin(c.roty)*sin(c.rotz))+sin(c.rotx)*(-
    x[1]*sin(c.roty)*sin(c.rotz)+x[2]*cos(c.rotz))-x[0]*cos(c.roty)*sin(c.rotz); 
    y[2]=x[2]*cos(c.rotx)*cos(c.roty)-x[1]*sin(c.rotx)*cos(c.roty)+x[0]*sin(c.roty);  
    */
    Mult(Mat,x,y,3,3,1);
    if (sqrt(pow(y[0][0],2)+pow(y[1][0],2))==0){

if (y[2][0]>=0)
    aux.elevation=Pi/2;
else{
    aux.elevation=3*Pi/2;
}
else{
    aux.elevation=atan(y[2][0]/sqrt(pow(y[0][0],2)+pow(y[1][0],2)));
}
if (y[0][0]==0){
    if (y[1][0]>=0)
        aux.azimuth=Pi/2;
    else{
        aux.azimuth=3*Pi/2;
    }
}
else{
    aux.azimuth=atan(y[1][0]/y[0][0]);
}
return aux;

void MoveServos (void){
    if (fabs(desired.azimuth-actual.azimuth)<Pi/180)
        actual.azimuth=desired.azimuth;
    else{
        if (desired.azimuth>actual.azimuth)
            actual.azimuth+=Pi/180;
        else{
            actual.azimuth-=Pi/180;
        }
    }
    if (fabs(desired.elevation-actual.elevation)<Pi/180)
        actual.elevation=desired.elevation;
    else{
        if (desired.elevation>actual.elevation)
            actual.elevation+=Pi/180;
        else{
            actual.elevation-=Pi/180;
        }
    }
}
```c
int main()
{

    LED_Init(); // Initialize LED light
    // Setting up pull up
    GPIOC->PUPDR |= (1UL<<2*4|1UL<<2*5|1UL<<2*6|1UL<<2*7);
    // Setting up general purpose output mode
    GPIOC->MODER |= (1UL<<2*0|1UL<<2*1|1UL<<2*2|1UL<<2*3);

    state=0;
    actual=UpdateAntena();
    s=UpdateSatellite();
    c=UpdateCar();
    IniRotationM(Mat,c.rotx,c.roty,c.rotz);
    SystemCoreClockUpdate(); /* Get Core Clock Frequency */
    if (SysTick_Config(SystemCoreClock / 1000)) {/* SysTick 1 msec interrupts */
        while (1); /* Capture error */
    }

    while(1) { /* Loop forever */
        switch (state){
            case 0:
                break;
            case 1:
                MoveServos();
                if (Check()){
                    state=0;
                }
                break;
        }
    }

    void SysTick_Handler(void) {
        msTicks++;
    }
        //Count milliseconds

        if (msTicks>99){
            s=UpdateSatellite();
            c=UpdateCar();
            UpdateRotationM(Mat,c.rotx*0.1,c.roty*0.1,c.rotz*0.1);
            //IniRotationM(Mat,c.rotx,c.roty,c.rotz);
            desired=FromGlobalToRelative();
            //actual=UpdateAntena();
```
msTicks=0;
}
if (Check()){
  state=0;
}
else{
  state=1;
}
Appendix J: Sample Code 2

/*
MikroC CAN example
*/

unsigned char Can_Init_Flags, Can_Send_Flags, Can_Rcv_Flags;                  // can flags
unsigned char Rx_Data_Len;                                                   // received data length in bytes
char RxTx_Data[8];                                                           // can rx/tx data buffer
char Msg_Rcvd;                                                               // reception flag
const long ID_1st = 12111, ID_2nd = 3;                                       // node IDs
long Rx_ID;

// CANSPI module connections
sbit CanSpi_CS at RC0_bit;
sbit CanSpi_CS_Direction at TRISC0_bit;
sbit CanSpi_Rst at RC2_bit;
sbit CanSpi_Rst_Direction at TRISC2_bit;
// End CANSPI module connections

void main() {                                                                // Main Program
  ANSEL = 0;                                                                // Configure AN pins as digital I/O
  ANSELH = 0;

  PORTB = 0;                                                                 // clear PORTB
  TRISB = 0;                                                                 // set PORTB as output

  Can_Init_Flags = 0;
  Can_Send_Flags = 0;                                                        // clear flags
  Can_Rcv_Flags = 0;

  Can_Send_Flags = _CANSPI_TX_PRIORITY_0 &                                   // form value to be used
                   _CANSPI_TX_XTD_FRAME &                                    //     with CANSPIWrite
                   _CANSPI_TX_NO_RTR_FRAME;

  Can_Init_Flags = _CANSPI_CONFIG_SAMPLE_THRICE &                            // Form value to be used
                   _CANSPI_CONFIG_PHSEG2_PRG_ON &                           //     with CANSPIInit
                   _CANSPI_CONFIG_XTD_MSG &
                   _CANSPI_CONFIG_DBL_BUFFER_ON &
                   _CANSPI_CONFIG_VALID_XTD_MSG;
SPI1_Init(); // initialize SPI1 module

CANSPIInitialize(1,3,3,1,Can_Init_Flags); // Initialize external CANSPI module

CANSPISetOperationMode(_CANSPI_MODE_CONFIG,0xFF); // set CONFIGURATION mode

CANSPISetMask(_CANSPI_MASK_B1,-1,_CANSPI_CONFIG_XTD_MSG); // set all mask1 bits to ones

CANSPISetMask(_CANSPI_MASK_B2,-1,_CANSPI_CONFIG_XTD_MSG); // set all mask2 bits to ones

CANSPISetFilter(_CANSPI_FILTER_B2_F4,ID_2nd,_CANSPI_CONFIG_XTD_MSG); // set id of filter B2_F4 to 2nd node ID

CANSPISetOperationMode(_CANSPI_MODE_NORMAL,0xFF); // set NORMAL mode

RxTx_Data[0] = 9; // set initial data to be sent

CANSPIWrite(ID_1st, RxTx_Data, 1, Can_Send_Flags); // send initial message

while(1) { // endless loop
    Msg_Rcvd = CANSPIRead(&Rx_ID, RxTx_Data, &Rx_Data_Len, &Can_Rcv_Flags); // receive message
    if ((Rx_ID == ID_2nd) && Msg_Rcvd) { // if message received check id
        PORTB = RxTx_Data[0]; // id correct, output data at PORTB
        RxTx_Data[0]++; // increment received data
        Delay_ms(10);
        CANSPIWrite(ID_1st, RxTx_Data, 1, Can_Send_Flags); // send incremented data back
    }
}
Appendix K: CAN Bus Codes and Standards

The SAE J1939 Communications Network
An overview of the J1939 family of standards and how they are used
An SAE White Paper

Since its publication more than a decade ago, SAE J1939 has become widely accepted as the Controller Area Network (CAN) for on-highway trucks, off-highway equipment, agricultural equipment, construction equipment, and other vehicles.

What is J1939?

From the Foreword to J1939 (Serial Control and Communications Heavy Duty Vehicle Network)... 
*The SAE J1939 communications network is a high speed ISO 11898-1 CAN-based communications network that supports real-time closed loop control functions, simple information exchanges, and diagnostic data exchanges between Electronic Control Units (ECUs), physically distributed throughout the vehicle.*

The SAE J1939 common communication architecture strives to offer an open interconnect system that allows ECUs associated with different component manufacturers to communicate with each other.*

J1939 covers the design and use of devices that transmit electronic signals and control information among vehicle components. Used as an application layer, J1939 provides communication between the engine control, transmission control, vehicle body control, and other applicable sub-control systems.

J1939 also defines message timeouts, how large messages are fragmented and reassembled, the network speed, the physical layer, and how applications acquire network addresses.

The J1939 communications network is defined using a collection of individual SAE J1939 documents based upon the layers of the Open System Interconnect (OSI) model for computer communications architecture.
A “Family” of Documents

The J1939 standards “family” consists of the top level document (J1939 itself) and 16 companion documents.

J1939 is the master control for definitions common to many applications. This document provides the comprehensive list of all assigned data parameter and diagnostic identifiers (SPNs), all assigned messages (PGNs), and all assignments for NAME and Address identifiers.

The top level document serves as the central registry for these assignments even though the technical details for most SPNs and PGNs are specified throughout the other documents in the J1939 family.

The top level document describes the network in general, the OSI layering structure, and the subordinate document structure, as well as providing control for all preassigned values and names.

J1939 is:
- Developed for use in heavy-duty environments
- Suitable for horizontally-integrated vehicle industries

The physical layer aspects of SAE J1939 reflect its design goal for use in heavy-duty environments. But the J1939 communications network is applicable for light-duty, medium-duty, and heavy-duty vehicles used on-road or off-road, and appropriate stationary applications which use vehicle-derived components (such as generator sets).

The companion documents explain component rationalization and product standardization for a particular application or industry. Specific documents in the J1939 family describe the recommended practices for networks in:
- Heavy-Duty On-Highway Vehicles
- Agricultural and Forestry Off-Road Machinery
- Marine Stern Drive and Inboard Spark-Ignition Engines

Companion documents also describe layers used in the OSI network architecture, such as:
- Physical Layer
- Data Link Layer
- Network Layer
- Vehicle Application Layer

J1939 Compliance

There is no procedure presently in place to test, validate or provide formal approval for ECUs utilizing the SAE J1939 network. Developers are expected to design their products in the spirit of, as well as the specific content of, the recommended practice. In the future, procedures may be defined for the testing of products to ensure compliance. Until then, compliance is determined by the manufacturer of the component. J1939 gives OEMs the ability for customized communication.
SAE Ground Vehicle Technical Committees: Electrical Systems Electromagnetic Compatibility (EMC) Standards Committee

The Electromagnetic Compatibility (EMC) Standards Committee reports to the Electrical Systems Group of the Motor Vehicle Council. The Committee is responsible for developing and maintaining SAE Standards, Recommended Practices, and Information Reports related to all aspects of EMC as it applies to surface vehicles and their components regardless of propulsion method or electrical system voltage. Phenomena addressed include radio frequency emissions, radio frequency immunity, electrostatic discharge (ESD) immunity, and transient voltage emission and immunity.

The Electromagnetic Immunity (EMI) Task Force and Electromagnetic Radiation (EMR) Task Force have been developed under the committee’s scope.

Participants in the Electromagnetic Compatibility (EMC) Standards Committee include OEMs, suppliers, consulting firms, government, and other interested parties.

SAE Electromagnetic Compatibility (EMC) Standards Committee Standards Development & Revision Activities

J1113/1 Electromagnetic Compatibility Measurement Procedures and Limits for Components of Vehicles, Boats (up to 15 m), and Machines (Except Aircraft) (16.6 Hz to 18 GHz)

J1113/11 Immunity to Conducted Transients on Power Leads

J1113/12 Electrical Interference by Conduction and Coupling—Capacitive and Inductive Coupling via Lines Other than Supply Lines


J1113/26 Electromagnetic Compatibility Measurement Procedure for Vehicle Components—Immunity to AC Power Line Electric Fields


J1113/4 Immunity to Radiated Electromagnetic Fields—Bulk Current Injection (BCI) Method


J1752/2 Measurement of Radiated Emissions from Integrated Circuits—Surface Scan Method (Loop Probe Method) 10 MHz to 3 GHz
SAE Electromagnetic Compatibility (EMC) Standards committee
Standards Development & Revision Activities

J1752/3 Measurement of Radiated Emissions from Integrated Circuits—TEM/Wideband TEM (GTEM) Cell Method; TEM Cell (150 kHz to 1 GHz), Wideband TEM Cell (150 kHz to 8 GHz)

J1812 Function Performance Status Classification for EMC Immunity Testing

J2556 Radiated Emissions (RE) Narrowband Data Analysis—Power Spectral Density (PSD)

J2628 Characterization—Conducted Immunity

J551/1 Performance Levels and Methods of Measurement of Electromagnetic Compatibility of Vehicles, Boats (up to 15 m), & Machines (16.6 Hz to 18 GHz)

J551/15 Vehicle Electromagnetic Immunity—Electrostatic Discharge (ESD)

J551/16 Electromagnetic Immunity—Off-Vehicle Source (Reverberation Chamber Method)—Part 16: Immunity to Radiated Electromagnetic Fields

J551/17 Vehicle Electromagnetic Immunity—Power Line Magnetic Fields

J551/5 Performance Levels and Methods of Measurement of Magnetic and Electric Field Strength from Electric Vehicles, Broadband, 9 kHz To 30 MHz

One world.
One standard.
One Source.

Join an SAE Standards committee.

For more information or to participate on an Electromagnetic Compatibility (EMC) Standards Committee, contact:

Maryvonne L. Jacquemart
1-248-273-2467
mjacquemart@sae.org
http://www.sae.org/servlets/works

To purchase SAE Technical Standards

1-877-606-7323
(US and Canada) or
1-724-776-4970

www.store.sae.org
customerservice@sae.org
A complete CAN family from transceivers to gateways

Add value to your networks with CAN solutions that set the standard

NXP Semiconductors has played a leading role in establishing CAN as the automotive networking standard. Covering all CAN physical layer options, our product portfolio includes automotive-grade high-speed, single-wire and fault-tolerant transceivers along with stand-alone protocol controllers and Fail-Safe System Basis Chips. And all our next-generation devices use the flexible SOI technology for best-in-class EMC performance.

Key features
- CAN is a robust protocol – essential for automotive applications
- ISO 11898 and SAE J2641 are open standards
  - Well documented and fully supported worldwide
- Choice of three CAN physical layer options
  - High-speed (HS) for high data rates
  - Fault-tolerant (FT) for additional robustness
  - Single-wire (SW) for minimum wiring

NXP CAN portfolio
- Vast experience with CAN in automotive applications
  - Over one billion transceivers shipped in September 2005
- Range of transceivers and protocol controllers for all three CAN physical layer implementations
- A family of innovative products
  - Offer additional fail-safe and low-power features
  - Golden devices for new CAN standards
  - Best-in-class EMC performance using versatile SOI technology

Further integration simplifies individual node and system design
- Gateway controllers integrating multiple CAN controllers with a 32-bit ARM processor
- Fail-Safe System Basis Chips (SBCs) combining transceivers and voltage regulators with an autonomous node
- Fail-Safe system

The Controller Area Network (CAN) bus is the primary automotive networking protocol for powertrain, backbone bus and body electronics. NXP has been recognized as the leading CAN innovator ever since our first industry-standard PCA82C250 transceiver set the benchmark for high-speed CAN. We now offer designers a comprehensive portfolio comprising automotive-grade transceivers for all CAN physical layer options (high-speed, fault-tolerant and single-wire), stand-alone protocol controllers, System Basis Chips and the latest in IVN gateway ICs, delivering the performance and functionality needed for today’s wide variety of in-vehicle networking applications.
As the initial choice of CAN physical layer depends mostly on the network performance required, our CAN solutions ensure your selection meets the highest standards possible. Of the three current implementations of the CAN physical layer, high-speed CAN offers the highest transmission speed (up to 1 Mbit/s). Fault-tolerant CAN operates at a slower rate (125 kbit/s) but maintains functionality in the case of a broken or shorted bus wire – particularly important in body electronics where the wiring harness is more vulnerable. Both of these CAN physical layer implementations interconnect network nodes via a two-wire twisted pair bus with end-termination. Single-wire CAN is used primarily to reduce wiring in body electronics implementations and for diagnostic purposes, and operates up to 41.6 kbit/s.

**Delivering value-added performance to automotive CAN networks**

The CAN protocol assumes good connections and interference-free signals between nodes in a network. As the CAN protocol itself is not application specific, the automotive environment presents a significant challenge to achieving accurate operation. Obvious risks include physical damage or disturbance and electromagnetic interference. Another important factor is the battery power source, which puts additional demands on power consumption and short-circuit protection. The ISO and SAE CAN standards do not cover all these eventualities, leaving scope for extra functionality and reliability improvements when implementing CAN networks.

All NXP CAN transceiver and controller devices deliver more than the minimum CAN standard requirements, such as comprehensive fail-safe features, diagnostics and low-power modes. Many of these additional features are currently adopted by newer versions of the standards, another process in which our leading role continues. We were also first to introduce automotive-grade transceivers to the market.

Failure management in an automotive CAN network serves one goal: keeping the car going. When a failure is detected, the robust CAN protocol handles bus arbitration and the re-transmission of messages, helping minimize effects on the rest of the network and the car. Supplementing the error handling capabilities of the CAN protocol and the selected physical layer, NXP transceivers all provide additional protective functionality to help safeguard against hardware failures.

Power saving is important for key-off functionality, with networks remaining in standby when the car’s engine is switched off. These systems are directly connected to the battery, so power consumption must be kept low. Also, in partial networks (where some nodes remain active when the rest of the network is switched off) the key-on transceivers have to leave the active part of the network unaffected when their power supply is cut. Once again, this is an area where our CAN family delivers the additional features required by today’s automotive applications.

**The CAN protocol**

In networked applications the generic ISO/OSI reference model (illustrated) identifies seven distinct communication layers, excluding the actual bus wiring. What is commonly referred to as CAN involves only two of these layers – the data link layer and the physical layer – and is covered by the ISO 11898 standard. In CAN applications the higher-level layers of the ISO/OSI model deal with the application-specific processing of the CAN messages. As the actual CAN standard only addresses the basic network communications, this leaves the implementation of features such as fail-safe behaviour and low-power modes as proprietary options.

The ISO 11898 standard is divided into three sections:
ISO 11898-1 covers the CAN protocol, while parts -2 and -3 handle two of the three standardized implementations of the CAN physical layer (HS-CAN and FT-CAN respectively). Although the transmission medium is not included in the standard, ISO does assume CAN_L and CAN_H bus connections and defines the electrical signal levels for them.

Single-wire CAN is the third standardized CAN physical layer implementation in use today and is covered under the SAE/J2545 standard. SW-CAN uses only the CAN_H connection, taking the node’s ground as reference level. To maintain low electromagnetic emission, the maximum communication speed of SW-CAN systems is limited.
## Overview of CAN physical layer characteristics and application areas

<table>
<thead>
<tr>
<th>Features</th>
<th>HS-CAN</th>
<th>PT-CAN</th>
<th>SW-CAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data link layer standard</td>
<td>ISO 11898-1</td>
<td>ISO 11898-1</td>
<td>ISO 11898-1</td>
</tr>
<tr>
<td>Physical layer standard</td>
<td>ISO 11898-2</td>
<td>ISO 11898-3</td>
<td>SAE/ISO 2413</td>
</tr>
<tr>
<td>Number of bus wires</td>
<td>2 (twisted pair)</td>
<td>2 (twisted pair)</td>
<td>1</td>
</tr>
<tr>
<td>Maximum bus speed</td>
<td>1 Mbits/s</td>
<td>125 kbits/s</td>
<td>33/41.6 kbits/s</td>
</tr>
<tr>
<td>Bus communication signal</td>
<td>![Signal Diagram]</td>
<td>![Signal Diagram]</td>
<td>![Signal Diagram]</td>
</tr>
<tr>
<td>Bus termination principle</td>
<td>![Termination Diagram]</td>
<td>![Termination Diagram]</td>
<td>![Termination Diagram]</td>
</tr>
<tr>
<td>Bus wire short-circuit and interrupt</td>
<td>limited short-circuit</td>
<td>tolerant against any</td>
<td>no tolerance</td>
</tr>
<tr>
<td></td>
<td>tolerance</td>
<td>single bus wire</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>short or interrupt</td>
<td></td>
</tr>
<tr>
<td>NXP transceiver features 1</td>
<td>- bus dominant time-out</td>
<td>- bus dominant time-out</td>
<td>- loss of ground protection</td>
</tr>
<tr>
<td></td>
<td>- bus clamping protection</td>
<td>- bus clamping protection</td>
<td>- 100 kbits/s flash mode</td>
</tr>
<tr>
<td></td>
<td>- partial networking support</td>
<td>- partial networking support</td>
<td>- partial networking support</td>
</tr>
<tr>
<td></td>
<td>- stand-by and sleep modes</td>
<td>- stand-by and sleep modes</td>
<td>- selective sleep</td>
</tr>
<tr>
<td></td>
<td>- node power management</td>
<td>- node power management</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- local and remote wake-up</td>
<td>- local and remote wake-up</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- failure diagnosis</td>
<td>- failure diagnosis</td>
<td></td>
</tr>
<tr>
<td>Automotive applications</td>
<td>- engine management</td>
<td>- body &amp; comfort</td>
<td>- body &amp; comfort</td>
</tr>
<tr>
<td></td>
<td>- backbone bus</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- body &amp; comfort</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 depending on the transceiver used

---

![Vehicle Diagram]
Appendix L: Team Resumes

Education

University of San Diego        August 2010 - Present

- Engineering GPA: 3.73 – BS/BA in Electrical Engineering - Math Minor - 1st Honors Fall 2012 & Spring 2013 - Graduation Date: May 2014

Experience

Payload STE Engineering Intern | The Boeing Company        June 2013 – August 2013
Aid my coworkers in any tasks that they needed to complete, ranging from typing up a document to making hardware in the lab
Help plan, build, and test the Payload special test equipment (STE) for use on an upcoming satellite
Design and build a custom test fixture to troubleshoot RF switching issues
Review and create official documents to be presented to the customer

Residential Life Desk Worker | University of San Diego     August 2011 - Present
Operate the Onity program to allow students access to their rooms
Assist hundreds of people with housing, schooling, and general questions
Aid my superiors in any crucial tasks that needs to be completed by a certain deadline

Intern | Title365                                    June 2010-August 2010, May 2012-July 2012
Correct mistakes on any forms that were given to me
Transfer data for numerous checks received by the company in order to obtain an internal record on file

Courier | Titan Terminal and Transport          June 2008 – October 2008
Deliver pumps to a repair facility
Load and unload pumps from a truck
Communicate with the workers at each location

Skills

- C++/C
- MatLAB
- Assembly Language
- Experience in the Lab
- Experience with NXT Robots
- Word
- Excel
- Multisim
- Assembled/Programmed Solarbotics Sumovore Robot
- VHDL/FPGA Experience
- Leadership skills
- Arduino weekend projects
Achievements

• At The Boeing Company, a project I was working on was coming to its intense final stages, and everyone was extremely busy. A vendor who supplied us with over 200 switches for our hardware just notified us that 90% of the switches were made incorrectly, and that they would not work. This caused major problems up the entire employee chain, and the manager of the entire project was notified and brought into the situation. The problem was resolved when my superiors came to me and asked me to design, create, and use a switch testing machine, in order to see which switches worked and which didn't. They gave me two days, and I worked hard to come up with the drawing and work with a technician to complete this machine. It worked perfectly, so I was driven to an offsite company to test the switches before they were put into the final product. I also came back and tested the switches we had at the site, and reported my findings to my superiors. I performed well under the time restricted situation, and saved the group time and money that they could not afford to lose.

• While working at Title365, the administration realized that they had made mistakes on thousands of forms they had created over the course of a month. I was given the task of correcting these forms, and additional workers were hired to work on them as well. I realized that it was a repetitive task, so I used a program that I had previously used to automate the system. This program averaged 2.5 forms a minute, while a human could do no more than one per minute. By setting up this program on many different computers, the task of fixing the thousands of forms was completed within a week. If I had not automated this system, the company would have had to pay additional workers to fix these for at least two weeks.

Involvement

• Member of IEEE

• Player/Safety Officer of the University of San Diego Men’s club soccer team
Curriculum Vitae Gonzalo Albaladejo

860 Missouri Street, Ap. #D
San Diego, 92109, CA
DOB 03/11/1991 Madrid, Spain
+1 (619) 757-4331
gonzalo.albaladejo@hotmail.com

Summary
- Knowledge of mechanical, civil, electrical, computer science and electronic engineering obtained during his Industrial Engineering degree at ICAL
- Specialized in Artificial Intelligence and programming Intelligent systems

Languages
- Fluent in English and Spanish
- Studied 2 years abroad in the US during university, 2 months in Ireland and 2 months in Canada during high school

Education
Colegio Mirabal 1995 - 2009

Industrial Engineering 2009 - 2012
ICAL Universidad Pontificia de Comillas

Industrial Engineering (exchange program) 2012 - 2013
San Diego State University

Industrial Engineering (exchange program) 2013 - Present
University of San Diego

Awarded with the CAE (Cambridge Certificate in Advanced English) 2007
University of Cambridge. ESOL Examinations

Career History and Accomplishments
- Had been chosen annually by his school (Colegio Mirabal) to attend the Spring Mathematics Contest of the UCM (Universidad Complutense de Madrid)
- Had been Project Manager and Software Designer in three different projects:
  - Mobile robot that would play hide and seek in an unknown environment using only distance sensors and encoders for the motors of the wheels
  - Mobile robot that would play basketball
  - Program the artificial intelligence for an automated house that would control lighting, blinds, heating, etc. by itself and test it in a model equipped with the necessary sensors and actuators
- Was chosen to represent Spain at the IASS competition that took place at Cranfield University (England) in the summer of 2008. The International Aerospace Summer School is a program sponsored by Eurofighter Typhoon.

Software knowledge
- Office: Microsoft Word, Microsoft Excel, Microsoft PowerPoint
- Mathematics: MATLAB, Derived
- Graphic Design: AutoCAD, Cabri
- Electrical Engineering: Orcad, PSpice
- Statistics: SPSS
- Digital Modeling: ISE, Quartus
- Embedded Programming: Arduino, MPLAB, IAR
- Software Development languages: Assembler, Java, C, C++
Christopher Anderson

2240 Garnet Ave Apt 6
San Diego, CA 92109
(503) 200-7737
andersoncd14@gmail.com

OBJECTIVE
Long term goals are to develop the skills and knowledge required of a well-rounded engineer, both personally and professionally. Short term goals are to achieve a strong command of any delegated tasks, remain focused and assertive, network as much as possible, and remain active and approachable in the workplace.

EDUCATION
University of San Diego
Overall GPA: 3.70
2010-2014
Currently enrolled as a full time Electrical Engineering student.
Completed relevant coursework includes:

- Lab and course experience in circuit design and function, C++ and Assembly programming, VHDL and FPGA programming, digital logic design, microcontroller operation, and electromagnetic forces.

WORK EXPERIENCE
United State Marine Corps—San Diego, CA  January 2004 - Present
Currently an E-6 undergoing education as part of an officer selection program, I have developed a valuable set of skills, including: an exceptional work ethic, an acute attention to detail, teamwork, and, as a Staff NCO, well-rounded managerial and leadership capabilities.
RYAN MALISZEWSKI

18708 Lunada Pt  ●  San Diego, CA 92128  ●  (858) 663-3863  ●  rmalisze@hotmail.com

ENGINEERING INTERN

♦ High motivatcd engineering major seeking to apply technical knowledge to the business environment, and to contribute to, and gain experience with an industry leader.

♦ Detail-oriented and focused professional able to grasp abstract and complex concepts and proofread and identify proper documentation.

♦ Organized, reliable, and capable of establishing priorities to ensure completion of projects and achievement of goals.

KNOWLEDGE AND SKILLS

♦ Microsoft Office applications
♦ C++ programming
♦ Robotics design, build and programming
♦ Audio amplifier
♦ Assembly programming
♦ Windows OS
♦ PowerPoint presentation
♦ Circuity
♦ Patent research and proofreading
♦ FPGA design, simulation, implementation and debug

WORK EXPERIENCE

USD book store (August-September 2011, August-Now 2013)
University of San Diego- 5998 Alcala Park, San Diego, CA 92110

♦ Worked a cash register
♦ Helped organize books
♦ Helped customers find what they needed
♦ Put together online orders for customers to pick up later

Legal Proofreader (April 2013- August 2013)
ALC Intellectual Property— 922 W. Baxter Dr., Suite 100  South Jordan, UT 84095

♦ Reviewed and proofread issued U.S. patent applications and office action responses for printing errors and mistakes.
♦ Wrote office actions in response to examiner rejections

Teachers Assistance (Summer 2011, 2012)
University of San Diego- 5008 Alcala Park, San Diego, CA 92110

♦ Assisted with lab activities for JROTC summer camp
♦ Set up and judged robotic competition
Legal Proofreader (2006–2010)
Law Office of Gerald Maliszewski — San Diego, CA

- Reviewed and proofread issued U.S. patents for printing errors and mistakes.
- Organized and prioritized workload pertaining to documentation review.
- Contributed to decisions on requesting USPTO Certificates of Correction.
- Conducted prior art research related to patent application submissions.

EDUCATION

University of San Diego, San Diego, CA
BA/BS in Electrical Engineering (Anticipated - 2014)

Rancho Bernardo High School, San Diego, CA
Graduated 2010

REFERENCE

Dr. Thomas Schubert, Professor of Electrical Engineering, University of San Diego
Phone: (619) 260-4892
Email: schubert@sandiego.edu

Phone: (801) 285-5175
Email: bhanks@alg-ip.com