Disaster Zone Emergency Response Vehicle (DZERV)

Marcial Rosario, Michael Lopez, and Robert Smith

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — This document discusses the design of our Disaster Zone Emergency Response Vehicle (DZERV). DZERV is a mobile search and rescue vehicle designed to allow search and rescue crews to safely execute such missions without putting themselves in hazardous environments or imminent danger. These missions will be conducted through a simple and intuitive graphical user interface, allowing the operator to remotely maneuver the car through the use of vehicle vision, as well as collecting environment data. The project itself uses various hardware components to achieve these tasks. These components are an ultrasonic rangefinder, pressure sensor, temperature and humidity sensor, an IP camera, and a wireless RF module.**

***Index Terms* — DZERV, search and rescue, sensors, microcontroller, wireless communication**

I. Introduction

The Disaster Zone Emergency Response Vehicle (DZERV) is capable of serving many useful and practical purposes and applications. The technology utilized in this project is one that can be found in various consumer electronics. The major focus of DZERV is to serve as a search and rescue robot, which can be implemented in the field in danger, disaster-like situations. The robot is to serve as a means of making human involvement in dangerous search-and-rescue operations much safer, thus enabling higher success rates of such operations.

DZERV will be capable of detecting whether the conditions in a disaster area are conducive to human survival, while at the same time searching for survivors, all while relaying this information back to a remotely located operator. Such a vehicle can be quite useful for either military or police application in disaster zones such as urban disasters, mining accidents, hostage situations, or explosions. DZERV will be able to accomplish tasks on rescue sites where either human or canine resources are rendered powerless, or where the situation might be too dangerous for a human to enter. Other benefits include reduced personnel usage and manpower, reduced fatigue of workers, and gaining access to areas otherwise unreachable by human or canine.

The main components of DZERV are a rover four-wheeled unit, a microcontroller unit, a sensors unit, camera unit, and wireless module. Additionally, a dark-detecting circuit will be employed. The vehicle will be controlled on a laptop via a custom designed GUI. From the interfacing of the operator's computer, the user will have several options on gathering useful data from the surrounding environment and searching for possible victims or survivors. The operator will have the ability to wirelessly maneuver the vehicle via on-screen buttons in the GUI, or through the use of keyboard arrows. The operator interface will also feature a live feed of the camera mounted on DZERV, allowing the operator to see where they are maneuvering, while also allowing them to scan the area for victims. Lastly, the operator will also have access to up-to-date environment data consisting of temperature, humidity, barometric pressure, and distance of objects. All of the sensor data transmission and wireless control of the car will be accomplished through the use of XBee wireless RF modules.

II. Requirements

The main requirement of DZERV is that it must be able to survey and gather information from its surrounding environment including temperature, humidity, and barometric pressure and then wirelessly transmit and display the measured values to base computer in real time. DZERV must also be capable of transmitting real-time video feedback to the operator on the base computer. DZERV also has several other requirements which must be met for successful completion of the project. DZERV should have a maximum wireless communication range of 275 feet. DZERV must be lightweight and portable, thus it should have a final weight less than 10 pounds. All of the sensors mounted on DZERV must be accurate to within 10% to ensure maximum efficiency. The final requirement of DZERV is that it should be able to operator at least 30 minutes on a fully charged battery while having low power consumption less than 10 W.

III. Design Method

The major goal of our design method was to divide our project into two separate major categories, hardware and software. Within each of these two categories, we broke down all of the major components, which would be tied in to implement the final product. With these components divided, each group member was assigned a specific

component to ensure maximum results and efficiency. We believed such a method would not only help us to be efficient, but also help in keeping our project organized. From here, we would hold several weekly meetings to report findings and updates, as well as addressing issues and problems. Through these meetings, we discovered what would work for us and what wouldn't. From this, we conducted all of our initial design and testing utilizing a breadboard. Once all of the circuits had been tested to specific requirements and constraints, we fabricated the final printed circuit board.

IV. Project Management

Our first step in properly managing our project was to set an overall budget to which we could adhere and afford, as we were not receiving any external funding. When planning a budget, we decided to take the smarter route and plan on budgeting for spare parts during the testing and building phase. This proved to be a wise idea once we encountered issues such as adding on extra parts to our design or burnt our components.

Due to the fact that our group consisted of only three members, we decided that the best option was to make this a collaborative effort and although being assigned specific components, we would all take equal responsibility in our project by having no set leader. Major hardware and software decisions were discussed amongst the group and collectively decided.

Throughout the course of the project, each member was assigned a particular task, which they were responsible for completing. Despite the fact that this was a collaborative effort, that one individual was responsible for the completion of that task. Despite having no team leader, we all took initiative to make sure we were kept well informed and up-to-date on every member's work. Several weekly meetings were scheduled to discuss updates on progression, current design, and potential pitfalls. Such meetings insured we were all kept up to date and that no two members were completing the same task. These meetings helped to give us a solid foundation on the current status of the project and where it was heading.

V. Implementation

Before the implementation stage, each circuit and sensor was thoroughly tested under multiple conditions and test cases in order to ensure that our components properly worked. Testing all of the circuits individually was done in the sense that just because a component works on its own does not mean that it will work once implemented alongside all the other components. Upon proper testing of circuits individually and collectively, we designed our PCB and rigorously checked its continuity before soldering our parts onto it. Once the parts were soldered and mounted, the components were checked and tested to ensure that they met all of our specific requirements.

VI. System Components

Our system is best presented in terms of system components, which will be presented in the following section. We will provide an overview of each of the individual physical modules that are interfaced to create the final product. Fig. 1 is a block diagram presenting an overview of our system's hardware configuration.

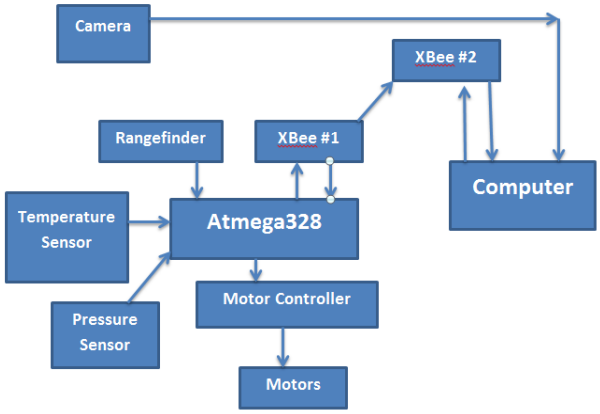

Fig. 1. Block diagram of all major subsystems for our project.

A. Microcontroller

The heart of the project is the Atmel ATmega328 which will be utilized as the MCU for the DZERV system. There were many deciding factors as to why we chose this microcontroller such as the flash memory, pin count, and CPU. The chip runs at 20 MHz, which is both necessary and sufficient to be able to handle the computations and component interfacing.

One of the most deciding factors for choosing an Atmel ATmega chip was the compatibility and availability of the Arduino development board and IDE. Arduino provides a wealth of sample code in order to aid developers in their projects. For testing and prototyping purposes, it was decided to use an Arduino Uno development board. This board came pre-equipped with 14 digital I/O pins, 6 analog pins, SPI, and serial RX and TX pins. For the final PCB utilizing a standalone ATmega 328 chip, it will be flashed with an Arduino bootloader in order to load

Arduino source code onto it developed in the Arduino language, a stand off of Java.

B. Barometric Pressure Sensor

The DZERV utilizes a barometric pressure sensor in order to detect an alarming rise in air pressure in the surrounding area. Such a rise in air pressure could be due to an explosion or a sudden change in altitude. While this would certainly be simple to see or hear with the use of the camera, it would also be helpful to have empirical data taken in the field to see the kind of effect this has on our vehicle.

The sensor chosen for this was the MPL115A1, manufactured by Freescale. The deciding factor in choosing this sensor was the fact that it transmits data via an SPI interface, which simply interfaces with the ATmega chip. The sensor has an absolute pressure range of 50 kPa – 115 kPa, and is accurate to within ±1 kPa.

The method of actually acquiring the pressure from the sensor can be a bit tricky, and consists of two formulas. The first formula (1) collects and calculates coefficients from the sensors, while the second equation (2) actually computes the pressure output in kilopascals. The formulas are as follows:

$$P_{comp} = a_0 + (b_1 + c_{12} \cdot T_{adc}) \cdot P_{adc} + b_2 \cdot T_{adc} \quad (1)$$

$$(kPa) = P_{comp} \cdot \left\lfloor \frac{115 - 50}{1023} \right\rfloor + 50 \quad (2)$$

C. Temperature and Humidity Sensor

The next sensor that DZERV will utilize is a temperature and humidity sensor. Such a sensor is essential to a search and rescue vehicle, as it will help the vehicle determine the ambient temperature of the surrounding environment. This will help the operator determine whether it will be safe for workers to enter the area to search and aid victims. Also, extreme changes or temperatures can not only harm victims, but can damage the vehicle as well.

The sensor chosen for this was the HTM1725LF produced by Measurement Specialties. The sensor has a typical 1 – 3.6 V output for 0 – 100% relative humidity and a temperature measurement through NTC 10kΩ ±3% direct output. The sensor has a range of 0 – 100% RH and -30 – 80 C with an accuracy of ±3% RH and ±3 C.

The method of acquiring the data output is rather simple one consisting of two linear equations for the relative humidity seen in (3) and (4), and another separate equation for the temperature seen in (5).

$$V_{out} = 25.68RH + 1078 \quad (3)$$

$$RH = 0.03892V_{out} - 41.98 \quad (4)$$

$$R_T = R_N \cdot e^{\beta\left[\frac{1}{T} - \frac{1}{T_N}\right]} \quad (5)$$

D. Range Finder

One of the major components of our project is our rangefinder. We use it to calculate distances as well to implement our obstacle avoidance algorithms. In selecting this component, there were a variety of factors we needed to take into consideration, as there are a few variations in technology for such rangefinders. There are infrared rangefinders, which rely on a line of sight principle to operate, which we felt would not adequately suit our needs. There are also ultrasonic rangefinders, which operate by sending out an ultrasonic pulse, and calculate the distance between the sensor and an object by calculating the amount of time it takes for the ultrasonic pulse to bounce off the object and travel back to the sensor. The advantage to this type of technology is the fact that these types of sensors do not operate on a line of sight principle, which means that objects can be detected by this sensor within a given angle of its position.

In order to accomplish this, we decided on an ultrasonic rangefinder manufactured by Maxbotix. Specifically, we chose to use the LV-MaxSonar-EZ1 rangefinder. We went with this model specifically because we had seen many projects that had used this sensor in the past with successful results, and decided that it would suffice to meet our requirements. The real advantages to using this sensor were that it offered us an incredible range, being that it is able to detect objects over 21 feet away. It also provides a great deal of accuracy, with an inch of resolution at that distance. We have this configured with our microcontroller to continuously sample distances so that we always know how far away our vehicle is from whatever it might encounter in the field. Additionally, we will have a collision detection algorithm implemented so that we can implement an auto-braking functionality to our project. This way, we can avoid running into objects on an autonomous level.

This rangefinder operates based on two very simple formulas. The first one (6) calculates a volts-per-inch

scaling factor, based on the input voltage. The second formula (7) then relates this voltage to a measured analog voltage to finally calculate the actual range. These equations can be seen below:

$$\frac{Vcc}{512} = Vi. \tag{6}$$

Here, Vcc is the supply voltage, and Vi is the volts-per-inch scaling factor. The actual range is calculated via (7):

$$\frac{Vm}{Vi} = Ri, \tag{7}$$

where Vm is the voltage measured by the rangefinder, and Ri is the range in inches.

E. Camera

Another crucial hardware component to our project is the camera. This will serve as our eyes and ears in the field, and it is important to have proper functionality. We considered a variety of options for our camera, including using an embedded camera, or even a normal everyday digital camera. However, in the end, we decided to stick with an IP camera. We decided that this would be the best option for us because it would prevent the need for additional processing power, as transmitting video wirelessly is incredibly resource-intensive. Additionally, we found that this also simplified the process of transmitting the video wirelessly. The major disadvantage of using an IP camera is the fact that they are significantly more expensive than any of the embedded cameras that we had researched. The higher cost associated with these cameras is fitting, however, given the fact that these cameras are significantly more advanced. They can handle much more data much more quickly, and require very little setup. Despite the much higher cost, we collectively decided that all the advantages highly outweighed this one major disadvantage, and proceeded with the IP camera.

To handle our video needs, we purchased the Wireless-G camera from Linksys. This camera operates on the IEEE 802.11g technology, and includes a built-in webserver for the transmission of data via the Internet. It supports MPEG 4 video format, and has a maximum video resolution of 324 by 240 pixels, which is not very high resolution, but for the goals of our project, the picture quality is not of great importance. Therefore, this resolution is acceptable. The only things this camera needs for proper functionality are a working Wi-Fi connection, a 5V, 1A power supply, and a computer with the camera firmware loaded. After deciding on a camera, we needed to determine a means of having the video transmitted from the vehicle to our remotely-located base computer. To do this, we found a library within Processing, the environment we are using for our Graphical User Interface. This library contains functions which actually pull the data from the camera over the internet and place the video feed within our custom-made user interface.

F. Dark-Detecting Circuit

Because of the need for external illumination for our vehicle when entering dark areas, we realized that we would need a source of light. We considered a few various options to make this happen, including wiring up an LED circuit to light our vehicle's path. However, keeping this light on at all times would add to our overall power consumption without a way of turning the light on and off remotely. Additionally, having an external switch for such a circuit would not be very feasible either, since it would require someone to physically turn the switch on and off. Because this would not be feasible, we needed to determine a way to automate this process for us.

This automation came in the form of a dark-detecting circuit, which can detect the absence or presence of light and operate an LED accordingly. This circuit is quite simple, containing only 4 components: an LED, a bipolar junction transistor, a resistor, and a phototransistor. The phototransistor is actually the element that detects the light and acts as a switch. Whenever light is not present, the switch is closed, and allows the normal BJT to drive the LED. When light is present, however, the switch opens, thus opening the circuit, and removing current from the circuit, thus turning off the LED. We placed this circuit on a small piece of perf board, soldering small pieces of wires to connect the circuit. This circuit runs on 4.5V and in order to maintain this circuit without the use of additional regulation, we opted to power this circuit separately using three AA batteries to give us the 4.5V we needed. We placed these batteries in a small battery holder and connected the positive and negative terminals to their respective places on the perf board.

We also needed to ensure that the light from the LED would not interfere with the camera in any way so as to maximize the usefulness of the light, as well as that of the camera. To accomplish this, we placed the LED circuit at the front of our vehicle, and placed the camera at the back of our vehicle, where it could be elevated above the light so that any such interference would be kept to a minimum. Fig. 2 below depicts the schematic which we will be implementing for our dark-detecting circuit. As previously

mentioned, this schematic will be implemented on a perf board rather than a printed circuit board.

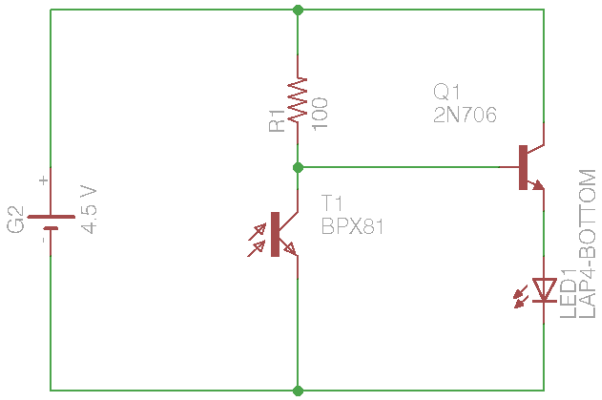

Fig. 2. Schematic for dark-detecting circuit.

G. Chassis

When first considering the chassis for our project, we had the option to use an established, pre-built one or design one ourselves. We ultimately decided to use the pre-existing body of a remote control car since the design was already optimized and tested and we wanted to spend more time on the actual design of our project rather than worry about designing a chassis that worked.

The requirements for our chassis were that it had to be compact enough to allow for easy maneuverability in and out of different disaster areas as well as to provide a basis for a modified platform to be mounted above the wheelbase where we will house our electronics. The model that allowed us to do this was a store bought Hummer H3. We were able to keep the stock motors attached to it since they allowed for a maximum speed of 7 mph, which allows us to limit the speed of our car easily in tight situations and to also achieve a decent maximum speed in case we must exit the zone quickly. Removal of the stock plastic that housed the battery proved easy and allowed us to attach a small horizontal platform that would hold our electronics and lower the overall height of the vehicle, allowing for us to access a wider range of areas due to the lowered vehicle height.

H. Power Supply, Batteries, and Voltage Regulation

For our main power supply, we decided to go with a 12 V 2000 mAh NiMH battery by Tenergy. The power requirement for all of our electronics is just below 5.4 W, which translates to us having about 30 minutes of operating time on a full charge. We opted for this NiMH battery because we did not have to worry about any memory effect issues over the course of this project and it was also less bulky and weighed less than the batteries that provided more mAh.

Most of our sensors require 5 V and in order to achieve this voltage level, we decided to use a switching regulator. We vouched for switching regulators due to their overall higher efficiency when compared to linear regulators as well as the wider voltage range it allowed at the input. Implementing the switching regulator added more components to the board, such as a Zener diode and an inductor, however, the Zener diode in the design would allow current to flow one way, saving our part in case current attempted to flow back into the regulator. We opted to use the LM2576 made by Texas Instruments for a few different reasons. For one, all of us had previous experience working with them in the lab. Free samples were also available from Texas Instruments and we took advantage of that opportunity. Also, this particular regulator allowed for a 7-40 V input voltage and since we were only using 12 V, we were able to achieve a 75% efficiency rating with this part, which would be much higher than if we used a linear regulator.

Our design also called for a 3.3 V regulator for our XBee module logic voltage. We input the same 12 V and used the 3.3 V model of the LM2576 and kept the rest of the schematic the same.

We opted for a separate power supply for the dark-detecting circuit that would illuminate the area and provide us with better vision for our camera. One main advantage for doing this is that it will not draw any additional current from our power source, allowing us to keep our desired 30 minutes of battery life. The circuit requires 4.5 V, which will be taken care of by 3 AA disposable batteries that will be held inside of a case.

I. Motor Controller

In order to control our motors, it was necessary to replace the existing circuitry that came with the vehicle with our own in order to convert the communication from RF to wireless. During our initial development stages, we decided to go with the Qik 2s9v1 Dual Serial Motor Controller made by Pololu to power and control our front and rear motors. The controller would have been connected to the two motors and then connected to an XBee wireless module that is responsible for sending and receiving instructions from the user via a laptop. The laptop would be connected to another Xbee module to serve as the master unit. This controller was ideal because it was cheap and allowed us to have full control over the direction and speed of our vehicle which we could control from the code that we created. Unfortunately, during testing, we found this unit to be rather unreliable, as testing with two of these units proved to be rather

unsuccessful. We therefore needed to look into another motor controller.

To compensate for all the issues we had experienced with the Pololu unit, as well as for lost time, as this is the unit we had been prototyping with the entire semester, we researched into using a controller with an H-bridge. The solution we decided to implement involved the Ardumoto motor driver shield, which is intended for direct interfacing with an Arduino development board. For the purposes of our project, we opted instead to build our own circuit revolving around another atmega328 microcontroller to mirror the functionalities of an Arduino board.

The motor controller, as shown in Fig. 3, features four output pins for the positive and negative terminals of two motors. It receives the same VCC power as the microcontroller it will be attached to, and also has a Vin input, which is used to drive the motors. This motor driver interfaces directly with four of the Atmega328's digital pins, 1 PWM pin and 1 digital output pin per motor. Additionally, this leaves real estate on the microcontroller open for directly interfacing with an Xbee wireless module. two pins are reserved for the XBee wireless module. The transmit pin on the XBee will be attached to the receive pin on the Atmega chip and the receive pin on the XBee will be connected to the transmit pin on the microcontroller.

Fig. 3. The Ardumoto motor driver shield. This controller serves as a middleman between our software and the actual control of our vehicle. Using software, we wirelessly send signals to this controller, which in turn operates the motors for our vehicle accordingly.

In addition, this motor controller is equipped with blue and yellow LEDs to indicate which motor is being driven. The driver lines are all diode protected to help protext against any back EMF that may be experienced as a result of operation.

J. RF Communication

Another large aspect of our project involved wireless communication of some sort. We needed a way to transmit all our sensor data and our video data to our base computer station. In the beginning, we had considered several options for accomplishing this task, including sending the data serially, via Bluetooth, Wi-Fi, or by implementing some other RF-based solution. We soon discovered that using RS-232 would be rather challenging, and a bit cumbersome to work with, and thus removed this from our list of possibilities. After investigating Bluetooth, we realized that this would not necessarily be a viable option for us as well, given the additional hardware and configuration required to make this work. We discovered that this was also the case for implementing a Wi-Fi based solution.

We thus came to the conclusion that implementing an RF-based solution would be the best way to proceed, and therefore spent time researching various methods for making this implementation work to fit our needs. After our research, we concluded that setting up a small network with XBee modules would be the most efficient way of doing this. Fig. 4 depicts one of these mdules.

Fig. 4. XBee RF wireless module. Serves as the backbone of our project in that these modules are responsible for all our telemetry, including the transmission of sensor data, as well as command data from the base computer.

Our solution hinges around the use of three separate modules. Two of them will be on our printed circuit board, and will handle the transfer of data from our vehicle back to the remaining module, which will be connected to the base computer and feed the data back to our control software. More specifically, one XBee module on our board will handle moving all the sensor data from the microcontroller to our computer and the other module will handle interfacing directly with our motor controller,

providing commands to the controller from our base computer. The final module will act as a hub of sorts to serve as the master, sending and receiving commands from the two slave modules located on our board.

VII. Software

In this section, we will be discussing the software which will encompass the entire project, including the sensors, microcontroller, wireless module, and the graphical user interface. The software for this project will be broken into two components, operator software and vehicle software. Fig. 5 below depicts the method in which we broke up the software, as well as giving an overview of each component.

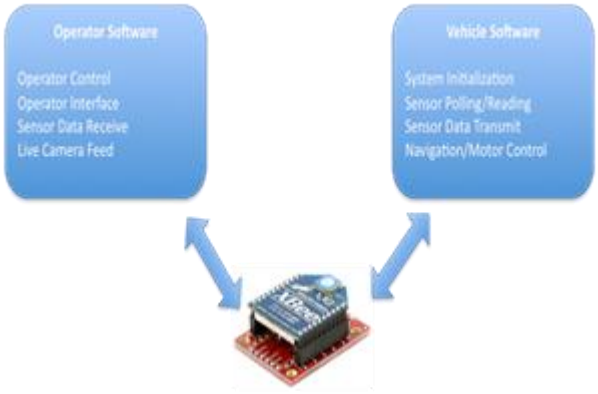

Fig. 5. Graphical overview of software functionality

A. Operator Software

The operator software will consist of the software written for the base computer which the operator will work off of. The major focus of the operator software will consist of the GUI developed for the user. Our goal was to develop a simple and intuitive interface in which any person without a technical background would be able to operate. The GUI will be written implementing Processing, an open source language and IDE. Processing was chosen to prior experiences and ease of use, as it contains all of the necessary libraries for capturing the live camera feed, interfacing with the XBee radios, and sending data over the serial port.

The user will have several options upon opening the interface. Initially, they will want to begin controlling the vehicle, where they have the option of either using the on screen controls, or controlling via the keyboard using the following combination of letters seen in Table 1 below.

TABLE I
SUMMARY OF VEHICLE OPERATION COMMANDS

Keyboard Command	Result
F	Forward
B	Brake
X	Emergency Brake
R	Reverse
→	Turn Right
←	Turn Left

Controlling the vehicle will also be completed implementing the Processing language. We chose this method due to the fact that the Processing language provides us with a method of transmitting data to the serial port in a simple fashion. The method in which the code will work, is that it will see if the user has pressed a key corresponding to the table above. If that key was pressed, we will send an appropriate hexadecimal value to the motor controller via the XBee module, which is calculated from the corresponding datasheet.

Once the user is navigating and controlling the car, they will automatically be viewing the live camera feed in the GUI transmitted over a wireless Internet signal. The user will also be receiving up-to-date sensor readings consisting of temperature, humidity, barometric pressure, and range of objects within the vehicle's path. Fig. 6 below depicts the interface created for the DZERV operator.

Fig. 6. Demonstration of the GUI's capabilities.

B. Vehicle Software

The heart of the software in the vehicle lies programmed within the ATmega microcontroller, which will be doing all of the sensor calculations and processing, as well as the manipulation of data to wirelessly send back to the base computer via the XBee modules.

Upon designing the software, we had some initial requirements which we planned on meeting for this

portion of the software. Our software must be able to handle all sensor polling and incoming sensors readings, handle the live camera feed, able to control both the rear drive and front steering motors of the vehicle, and communicate with the XBee modules to properly send and receive data.

The programming for the microcontroller was completed utilizing the Arduino language, as there is a plethora of examples and libraries available for guidance. The method in which the software will work is that the microcontroller will first poll the range finder, second the temperature and humidity sensor, and lastly, the pressure sensor. Upon polling the sensors, the microcontroller will store the readings and then send the data to the calculateTemp(), calculatePressure(), and calculateRange() functions, where the readings are calculated into respective values. Once calculated, the data will then be sent to the XBee module on the PCB, where it will transmit back to the base computer via the USB XBee module. Once it reaches this step, it will be read in and displayed to the user.

VIII. Conclusion

The Disaster Zone Emergency Response Vehicle project helped each group member to gain a considerable amount of experience within their respective fields of both electrical and computer engineering. Aside of this, the project helped to give us a real world insight on the way in which team projects function, which will help us in the work force.

This project also provided us with extensive hands-on experience with circuitry, hardware, software, as well as with the methodologies of integrating several hardware and software components into a fully functional system.

Overall, the project proved to be very comprehensive with the amount of hardware interfaces utilized and the complexity of the problem we were addressing. As a whole, the project has overcome various changes and revisions throughout these past few months. In the end, a search and rescue vehicle with wireless communication was successfully completed, meeting the goal of our initial proposal.

Acknowledgement

The authors wish to acknowledge and thank the review committee for taking time out of their schedules to judge and analyze our senior design project. We would also like to thank Dr. Richie for his assistance and guidance throughout these past two semesters. We would like to thank Joshua Childs, electrical engineer at Lockheed Martin, for his mentoring and guidance with issues in the project. We would also like to thank our families for providing us with moral support throughout these projects on the road to graduation and our engineering careers. Lastly, we would like to thank each other for all of the work that has gone into this project.

PROJECT ENGINEERS

Michael Lopez will be graduating from the University of Central Florida with a Bachelor's of Science in Computer Engineering in December 2012. He has participated in the Lockheed Martin College Work Experience for the past year, where he has gained valuable experience towards his engineering career. Upon graduation, he will be returning to pursue his Master's of Science in Engineering Management, while still participating the Work Experience Program.

Robert Smith will be graduating from the University of Central Florida in May of 2013 with a Bachelor's of Science in Electrical Engineering. His areas of interest include analog circuit design, linear controls and power systems. He plans to spend a couple of years in the work force and then return to school to pursue a Master's degree in his field of study.

Marcial Rosario will be graduating from the University of Central Florida in December 2012 with a Bachelor of Science degree in Electrical Engineering. He has participated in the UCF/Lockheed Martin College Work Experience program since February 2012, where he has worked for the Electrical Engineering department on various projects. Upon graduation, Marcial plans to work for Harris Corporation as a Systems Integration and Test Engineer.