A TELEMETRY SYSTEM FOR FIREFIGHTERS

Undergraduate Students: Sandip Uprety, Joseph Caglio, Michelle Ho, Chi Hou Chio, Stephanie Mckeefery, Jae Hyok Goh

Faculty Advisors: Michael Marcellin and Kathleen Melde
University of Arizona
Tucson, AZ 85721

Abstract

In this project, a telemetry system is implemented to save firefighters from potential danger in their working environment. Each Firefighter has a "node" or "unit" attached to them which contains temperature, oxygen, and carbon-monoxide sensors, and a transceiver. Each node constantly transmits data collected by the sensors to a central "base station." The base station consists of a laptop which is monitored by the Fire Chief at a safe distance from the scene, and it displays gas levels. The base station monitors the sensor readings, and sets off an alarm locally and also at the node if a reading has reached a predetermined critical value.

Keywords
Telemetry System, Saving firefighters, oxygen, carbon monoxide, and temperature monitoring.

INTRODUCTION

Currently an average of one hundred firefighters dies every year after being exposed to dangerous conditions. This is due in part to an incomplete monitoring system. Firefighter's equipment includes handheld radios, pagers, and other communication devices which do not alert them of dangerous conditions. Current firefighter equipment does not alert other firefighters and the base station that a firefighter is in danger.

The purpose of the project was to build a Telemetry System. Telemetry Systems can be used to remotely transmit and receive data. Components that are found in these systems include: signal source, data acquisition, data encoding, data modulation, transmit amplifier, transmit antenna, receive antenna, receive amplifier, demodulator (which includes carrier recovery, timing recovery, data decisions), and data decoding. Some of these components were designed by the team and some were purchased from different vendors.

This document discusses the system requirements, the design concepts and analysis, and design results followed by conclusions and recommendations, and acknowledgements for the project.

SYSTEM REQUIREMENTS

The system requirements have been listed in Table 1.

Table 1: Specification Chart for Firefighter Wireless Monitoring System

Specification	CTQ (metric)	Target Value & Tolerance	Part Name	Design Specification	Analysis	Degree of Certainty
Measure temperature (Critical temperature above 55°C)	Degree Celsius (°C)	Measure temperature between -40°C and 80°C within 1°C accuracy	TMP Digital Temperature Sensor	• Range: -40°C to 125°C • Accuracy: 0.00625°C	Design specification have more large range and less accuracy	High
Measure Carbon Monoxide level (Critical level above 500 ppm)	Parts per million (ppm)	Measure CO level between 10 ppm to 1000 ppm within 10 ppm	2CF CiTicel	• Range: 0-500ppm • Maximum Overload: 1000ppm • Resolution: 1ppm	Overload range meet the specification, and the resolution much less than target value	High
Measure Oxygen level (Critical level below 7%)	Percentage	Measure O_2 level in the air between 10% to 40% within 1%	MICROcel Oxygen Sensor	• Range: 0-25% O2 • Maximum Overload: 30% O2 • Accuracy: Not specification	The range of this sensor is good enough to use, because the critical level is below 7%	High
Transmit data wirelessly to base station	Range (meters)	Range of signal transmission at least 50 m	IRIS	Range: 500m	The range of IRIS mote is over 50m	High
Alert base station when critical levels are reached	Response Time (seconds)	Response time within 10 seconds	IRIS	Transmit data rate: 250kbps	The date rate of the mote is large enough to display the data in 10s	High
Locate the position of firefighters	Geographical coordinates	Locate the position within a radius of 5 meters	MTS 420 Sensor Board	Position accuracy: 10m	The accuracy of this board is twice than specification	Medium
Alert firefighters of danger when critical levels are reached	Response Time (seconds)	Response time within 2 seconds	TMP Digital Temperature Sensor	Response time: Temperature Sensor: 27.5ms	Temperature sensor has response time of 27.5ms which is much less than 2s	High
Alert firefighters of danger when critical levels are reached	Response Time (seconds)	Response time within 2 seconds	2CF CiTicel	Response Time (*T90): ≤17s	17 seconds is reasonable time	High
Alert firefighters of danger when critical levels are reached	Response Time (seconds)	Response time within 2 seconds	MICROcel Oxygen Sensor	Response Time (*T90): ≤ 15s	15 seconds is reasonable time	High

SYSTEM DESIGN

I) Design of Circuits

This telemetry system, as shown in Fig 1, collects data from the environment of the firefighter using two analog sensors and one digital sensor. Oxygen sensors and Carbon monoxide sensors are analog sensors. These typically have outputs as currents in the range of 1uA to 100 uA. Since these currents are very small in magnitude and contain a lot of noise, active filters are used to amplify and filter the signal obtained from the sensors. The following is a description of the

design of the filters used in the project. Separate designs are given for the Oxygen sensor and the Carbon monoxide sensor.

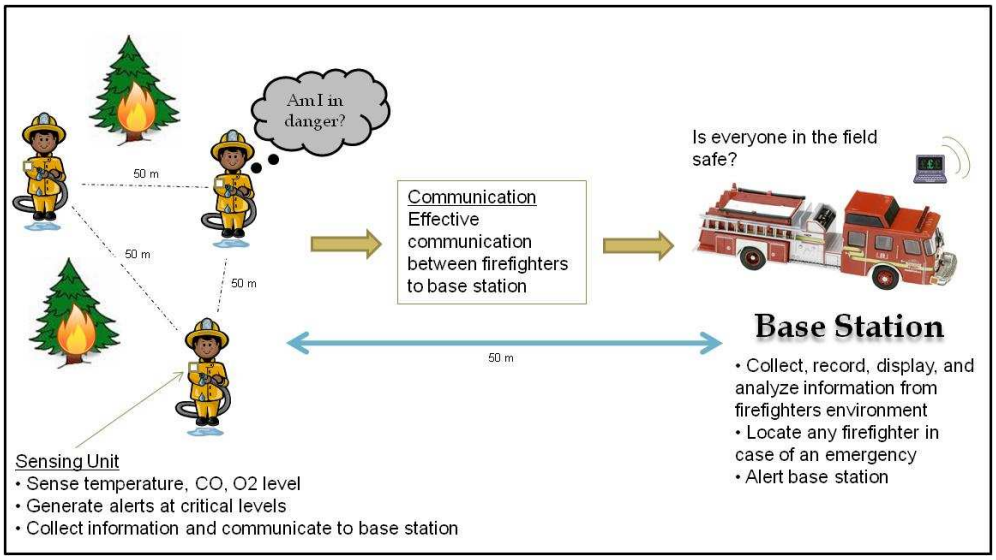

Figure1: Diagram of Firefighter Monitoring Telemetry System.

a) Oxygen Sensor Circuit

The O2-A2 sensor from Alphasense is used as an Oxygen sensor for this project. The output current from the sensor is collected over a 50 Ω resistor, which then produces voltages in the range of 1mV. This output from the sensors produced by electro-chemical reaction, is very noisy. Due to this, an active filter is essential to amplify the signal in the circuit and also to filter the noise. The schematic shown in Fig 2 is the circuit diagram for the low-pass filter implemented for the Oxygen Sensor and Fig 3 shows the Bode plot for the Oxygen sensor.

The active filter uses two op-amps OPA2376. This is a low-noise, low-quiescent current and high-precision operational amplifier with a very little offset voltage (5uV). It has a gain bandwidth product of 5.5GHz, and very low power consumption.

This active filter circuit has a gain of 52dB. The circuit consists of two parts: a pre-amplifier, and an active filter. The preamplifier as shown in the schematic has a DC gain of 26.89V/V. The input voltage to this circuit is the voltage across the resistor, over which the output current of the Oxygen sensor is collected. This part of the circuit produces an output which is in phase with the input. The output of this pre-amplifier is fed to the second part of the circuit which implements an active filter. The gain from the active filter is 14.52V/V. The resistors on the closed loop of both the stages have been kept low, which minimizes noise in the circuit. The cutoff frequency of the active filter is designed to be 50 Hz. This provides fast transient response of the circuit as it has a phase margin of 120˚, and also filters the undesirable noise from the circuit. The figure below shows the simulation of this circuit in LTSpice.

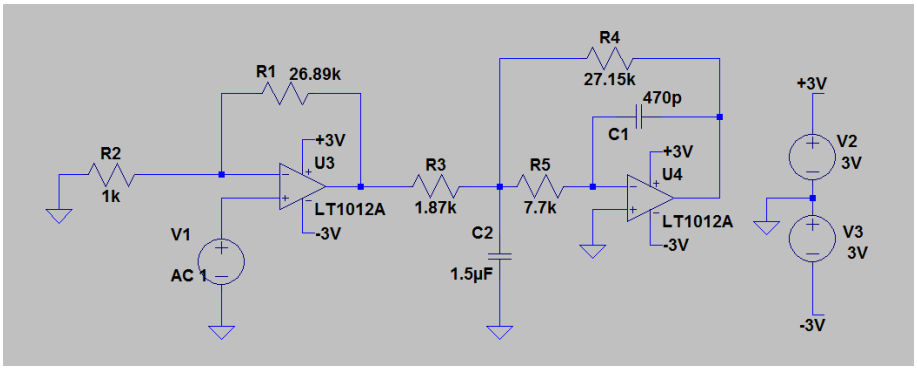

Figure 2: Diagram of low- pass filter for the Oxygen sensor.

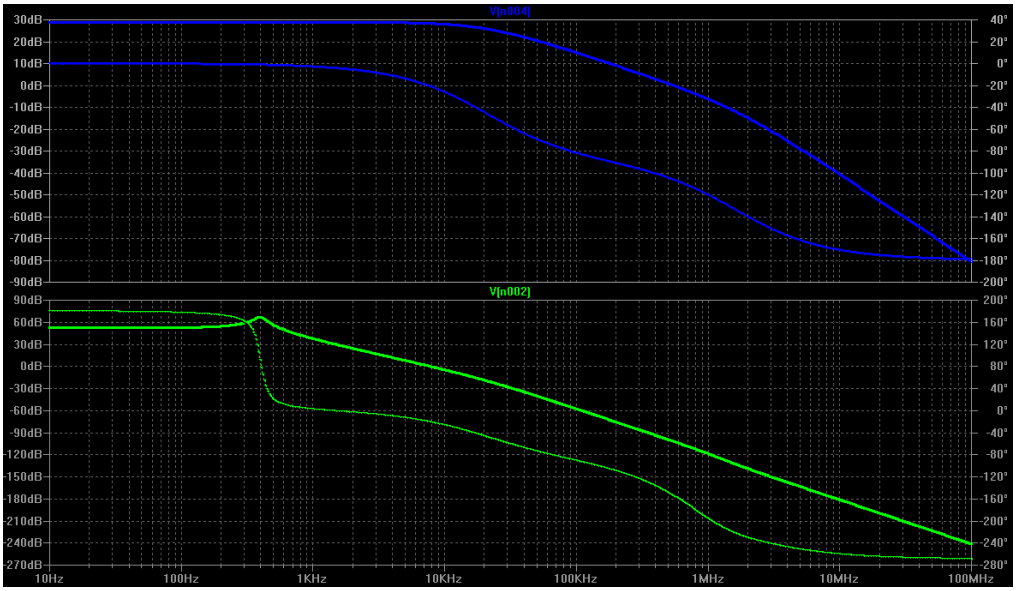

Figure 3: Simulation of the low-pass filter in LTSpice.

In the figure above, the upper panel shows the output of the pre-amplifier, which has 30 dB gain. The lower panel shows the final output of the filter. The desirable bandwidth and gain is achieved for the filter. The output of this filter is fed to the motes which then convert this analog data to digital for transmission to the base station.

b) Carbon monoxide sensor circuit

The Carbon-monoxide sensor used for the project is the CO-AE from Alphasense. This sensor also has current output in the range of 1uA to 100uA. It is a three pin sensor. The circuit shown in Fig 4 is implemented to make the sensor functional. The sensor has a working electrode, a counter electrode and a reference electrode. For a quick startup of the circuit, it is essential that the reference electrode and working electrode are at the same potential when the circuit has no

power. But, when the power is on, they have to have impedance across them. For this reason, JFET Q_1 is used in the circuit.

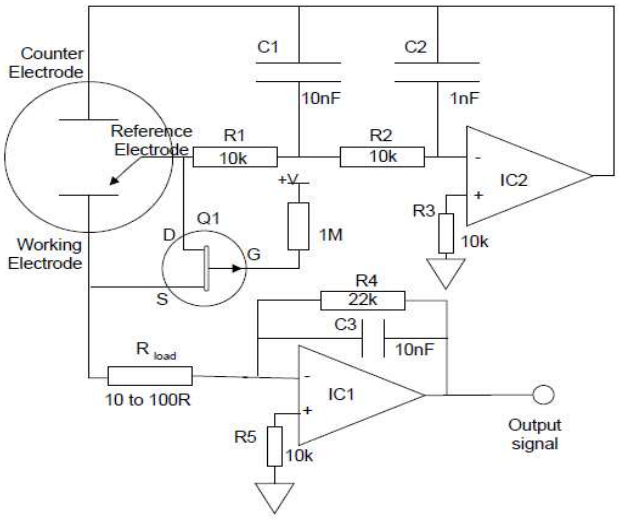

Figure 4: Diagram of Potentiostat circuit for zero bias toxic gas sensors.

Op-amp IC2 is used to supply current to the counter electrode, which completes the circuit loop when current flows out of the Working electrode dropping a voltage across R_{load}. Op-amp OPA2376 is used for this circuit as well. This sensor provides an output current in the range of 1uA at normal levels of CO in the atmosphere. This current is very low and sensitive to noise. So, a low pass filter of the first order is used to amplify the voltage across the resistor R_{load}. The gain of this filter can be set by varying resistor R_4. An R_{load} resistor of 57 Ω is used in this project. The cutoff frequency of the circuit can be adjusted by varying the capacitor C_3 in the circuit. The cutoff frequency is given by $\frac{1}{2 * pi * R4 * C3}$. The gain of the circuit is given by $\frac{R4}{Rload}$.

II) Design of Software

The software components of our project include two parts; the Crossbow Motes and the Base Station. The motes, which required modified firmware, run an open source embedded operating system called TinyOS. The Base Station consists of a custom Graphical User Interface (GUI) based application to display the sensor reading from the motes, to the user. Both of these components will be described in detail below, starting with the Crossbow Motes.

As mentioned earlier, the Crossbow Motes are wireless transceivers that can transmit data collected from attached sensors. The micro controller that is on the motes runs the TinyOS embedded operating system. Since the motes have TinyOS, programs can be written to run on the motes by using the programming language nesC.

Since nesC is object-oriented, it takes advantages of using interfaces that are used to control the various components on the motes. As shown in Fig 5, each nesC program that is written must be comprised of two files, a configuration file and a module file. The figure below shows a visual representation of how a nesC program is comprised of the two files.

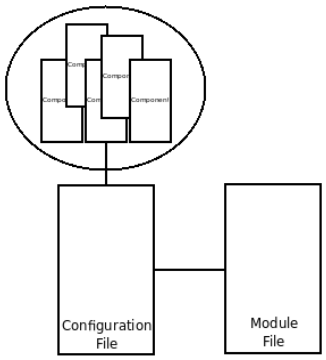

Figure 5: This is a diagram of a typical program written in nesC. Note how the module file has access to the components through the configuration file.

The configuration file is used to "wire" the different interfaces of the components that will be used in the program. This "wiring" is used so that the program being written can use different components on the motes without having to know exactly how the component is programmed. This enables new programs to be created faster and enables reuse of code so that the overall size of the program is reduced. The module file is used for the actual implementation for the program. It uses interfaces that were wired in the configuration file in which implementations of the events in the wired components has to be given. The module can also "provide" interfaces of its own. Using this programming language we interfaced the sensors that we built for the motes.

The other part of the software component is the Base Station. The GUI based interface is programmed in Java. The interface is responsible for displaying the sensor readings, sent from the motes, to the user. This is shown in Fig 6. We used Java, which is also an object oriented language, because it is portable and has a well documented API. Being portable indicates that the final application will be run on different computer platforms, without any modifications. This was important since some team members used the OSX operating system, while others used the Windows operating system. Having a well documented API was also important in our project because it enabled us to use a variety of libraries so that we could create a powerful interface without writing too many lines of code.

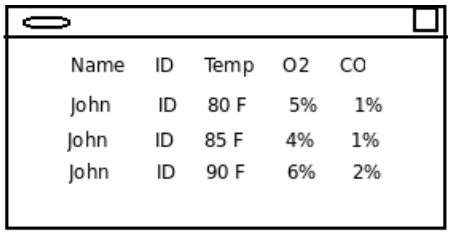

Figure 6: A rough sketch of the GUI for the Base Station.

The Base Station can be thought of being a way for user to monitor the wearer of the Crossbow Motes. While keeping the Base Station simple, we also want it to be reliable and expandable. The Base Station is kept simple by having a clean, easy to use GUI. A rough drawing of the GUI is shown in Figure 6. The data of each of the sensors on each mote can easily be seen therein. Also note that the user will quickly be able to see if any sensors read dangerous levels of temperature or gases according to a color coded system.

The Java application is well documented and takes advantages of the object oriented design so that the Base Station, if need be, can be easily modified. These two components make up the software part of the final design.

DESIGN INTEGRATION

This section discusses the interfacing between the sensors and the mote. The physical interface between the sensors and the motes is facilitated by the Crossbow MDA100 sensor board shown in Fig 7.

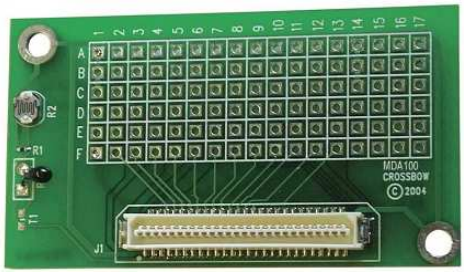

Figure 7: MDA100 Sensor Board from Crossbow.

It has a 51pin connector that matches the connector on the mote. The sensors and their peripheral circuitry were constructed on this board. The analog voltage output of the sensors after amplification and filtering is fed to the input of the ADC of the mote. We connected the temperature sensor, oxygen sensor, and carbon monoxide sensor to sockets F13, F14, and F15 on the MDA100 respectively. These three sockets correspond to ADC3, ADC4, and ADC5 ports of the mote. The sensor is powered directly from the battery of the node and this ensures the sensor output will never go above the limit of the analog to digital converter on the mote (0V – 3V).

RESULTS

I) Sensors

a) Carbon monoxide sensor

The carbon monoxide sensor was tested in two environments. When exposed to the regular atmosphere, the output voltage of the sensor read approximately 1.791V after stabilization. This voltage corresponds to about 0.5ppm CO in the atmosphere. When the carbon monoxide sensor

was placed at the opening of a car's exhaust pipe, the output voltage peaked at approximately 3.036V. The amount of carbon monoxide from a car exhaust pipe can be estimated to be 1000ppm. Fig 8 shows the model for the sensor.

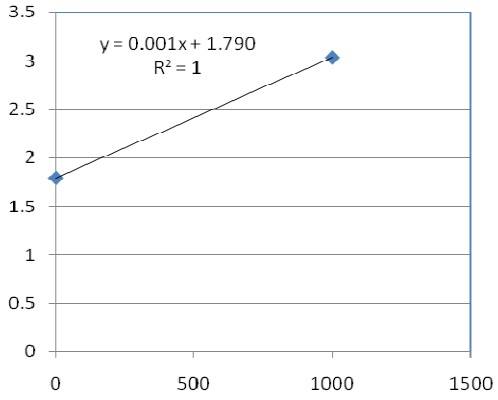

Figure 8: The sensors should be operating linearly from 0 to 2000 ppm. The equation that is obtained from this plot should allow for derivation of the voltage value that is at the critical level. Since the critical level is above 500ppm, according to the derived equation, this would equate to approximately 2.3904V.

b) Oxygen sensor testing and calibration

The Oxygen sensor and the signal processing circuit associated with it was tested for different levels of Oxygen in the atmosphere. An Oxygen tank with gas flow control was used to blow gas with varying Oxygen concentration into the sensor circuit. The following test data were obtained. The output of the Oxygen sensor went up with various oxygen levels according to the equation shown in Fig 9. The Oxygen sensor worked well, and the output current of the Oxygen sensor increased with the level of Oxygen concentration in the atmosphere.

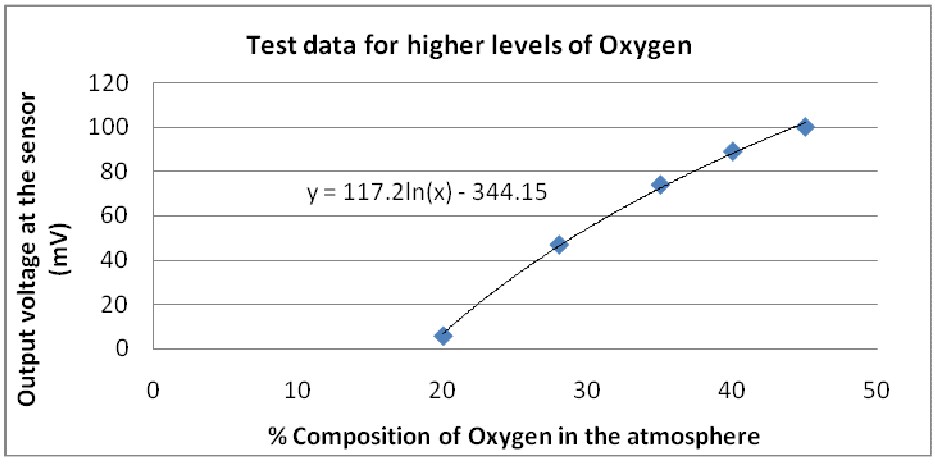

Figure 9: This graph represents the test data obtained for Oxygen levels.

Similar tests were done for Oxygen deficient atmosphere by blowing exhaled air which contains approximately 16% Oxygen. Using this data, calibration of the Oxygen sensor was done, and the equation for the model is listed in Fig 10, and used in the software to display Oxygen concentration for various levels of output voltage. The sensor and the circuit proved to be fully functional and successful. The circuit can be used to measure Oxygen levels from 0% to 40% in the atmosphere, which meets the functional requirements of the project.

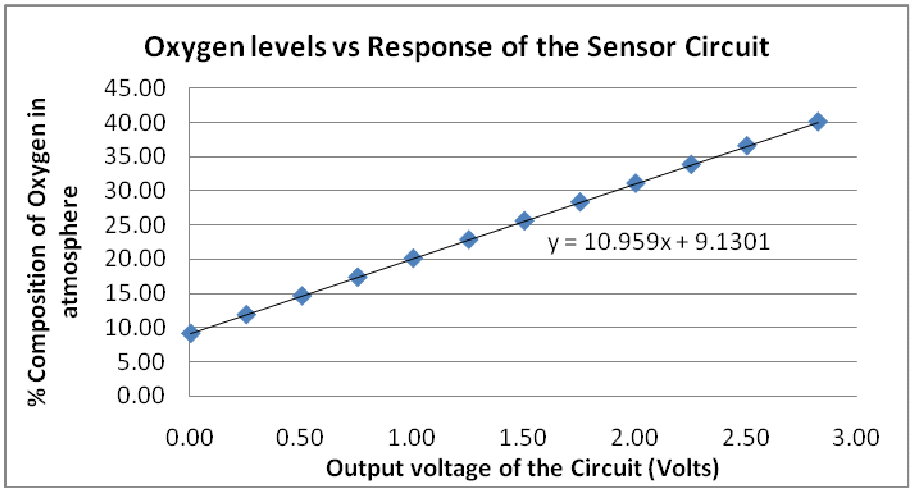

Figure 10: Linearized model of the Oxygen sensor.

II) Motes

The motes were tested at distances of about 200 feet, or approximately 60 meters. Data was easily transmitted with no delays. Furthermore, the motes were placed in different rooms to test transmission through indirect paths. Data was also obtained with no problems.

III) Overall System

In addition to testing individual components, the overall system was tested as well. While measuring oxygen and carbon monoxide data, voltmeter readings matched the data that was shown on the GUI display after wireless data transmission. Any noted differences were in the order of mVs. Consequently, data transmission for the overall system was accomplished.

Furthermore, when the data that transmitted reached certain threshold values, previously defined in the code, the "alarm" system was set off. The background color of the GUI successfully changed to demonstrate that critical values were reached.

CONCLUSION

With the use of motes, this telemetry system transmitted data successfully for at least fifty meters, even in the presence of walls, doors, and other obstructions. Data collected from sensors were successfully transmitted wirelessly and displayed on the base station GUI. The temperature, oxygen, and carbon monoxide gas sensors passed all tests, in which the output values changed according to the levels of temperature and gases in the environment. After each of the individual subsystems were interfaced together, data from the nodes were successfully transmitted to the base station. Thus, the project was implemented successfully and the overall system requirements and goals were achieved.

ACKNOWLEDGEMENTS

We would like to thank the following people who have supportive and have been involved in our senior capstone project:

- Dr. Michael Marcellin – project sponsor.
- Clayton Grantham – project mentor.
- Kevin Wood from Wireless Sensor Network Team for his help with fixing our drivers on our project hardware.

REFERENCES

[1] "*Designing a potentiostatic circuit,"* Alphasense, UK.

-http://www.alphasense.com/pdf/AAN_105.pdf

[2] Crossbow website: www.xbow.com

[3] TinyOS Community Forum: www.tinyos.net

[4] Carbon Monoxide research: http://www3.abe.iastate.edu/human_house/aen208.asp