Preliminary Screening System for Ambient Air Quality in Southeast Philadelphia

A Final Report submitted to Dr. Timothy Kurzweg, Dr. Kapil Dandekar, and the Senior Design Project Committee of the Electrical and Computer Engineering Department of Drexel University

Team Number: ECE-19

Team Members:

Mark Uva	Electrical Engineering
Robert Falcone	Electrical Engineering
Anthony McClellan	Electrical Engineering
Edward Ostapowicz	Electrical Engineering

Submitted in partial fulfillment of the requirements for the Senior Design Project

May 13, 2009

EXECUTIVE SUMMARY

Detection of dangerous particulate matter in ambient air has proven to be an expensive and time consuming process. As a result, a low-cost, yet effective, preliminary screening solution is necessary. The system developed utilizes a laser particle counter to capture samples of ambient air. Information collected by several laser particle counters is transmitted across an Xbee network to a central base station, where data from all nodes (each node consists of a laser particle counter and a Xbee component) is collected. Data is then passed through a mathematical calculation, so that it can be compared directly to the National Ambient Air Quality Standard (NAAQS).

The following milestones were reached:

- Fully functional system has been deployed in Southeast Philadelphia
- Software to handle all aspects of data analysis and presentation has been developed
    - Mathematical algorithm to convert laser particle counter data into data that can be compared against the NAAQS has been developed
- Dylos DC1100 Pro laser particle counter was customized to evaluate the number of particles deemed hazardous (those ranging from 0.5-2.5 μm and 2.5-10 μm) by the Environmental Protection Agency
- Experimentation performed using the Dylos DC1100 Pro yielded more precise power consumption data
    - Voltage regulator circuits (to limit the amount of power consumed by the system, and thus extend battery life) have been designed and fabricated
- Communication between nodes and a base station was achieved using Xbee wireless networking hardware
    - Data captured by the laser particle counter successfully transmitted to the base station
- Appropriate documentation (user and engineering manuals) have been written

The necessary system budget based on developments to date is $3,665, 8% lower than the initial estimate of $4,000. Developments have proceeded on schedule, and the system has been successfully deployed at the target location, with the help of the Clean Air Council of Philadelphia.

In addition to the technical milestones achieved, the design team had the opportunity to select and mentor a team of high school students from the Science Leadership Academy in Philadelphia, PA as part of the Drexel Engineering Projects in Community Service (EPICS) Program. The design team and high school students worked hand in hand through the engineering process, which concluded with the fabrication of a working system for their school.

ABSTRACT

Detection of dangerous particulate matter in ambient air by regulatory groups such as the Environmental Protection Agency (EPA) has proven to be an expensive and time consuming process. The design team has developed a low-cost yet effective preliminary screening solution, to determine if more comprehensive detection testing is required.

The system utilizes several laser particle counters (LPCs) to gather air quality data at different locations; each LPC is coupled with a wireless networking device, which transmits collected data to a central hub. The collected data is passed through a conversion algorithm (which converts particle count to concentration) so that it can be compared directly to the EPA standards.

Calibration was performed at the Air Management Service's (AMS) Lycoming Avenue Testing Facility in Northeast Philadelphia. The developed system was compared to the testing solutions utilized by the AMS to determine sensor accuracy and make improvements.

The solution in place will increase the number of neighborhoods that can be evaluated for particulate matter exposure. Currently deployed in South Philadelphia, the system conserves valuable time and capital by ensuring that only the areas most at risk are subjected to EPA testing.

This design process also incorporated high school students from the Science Leadership Academy in Philadelphia as part of The Drexel Engineering Projects in Community Service Program.

Table of Contents

List of Figures

List of Tables

Introduction and Problem Description

Over the past two decades, extensive research has been done to determine the deleterious health effects of fine particulate matter in ambient air on the human body. These studies have shown various short term, and long term effects resulting from exposure to $P.M._{10}$ (particles less than 10 µm in size) and $P.M._{2.5}$ (particles less than 2.5 µm in size) for certain lengths of time [1, 2, 3]. Studies evaluating the link between particulate matter and cardiopulmonary mortality have determined that approximately 50,000 to 100,000 Americans die each year as a result of exposure to these dangerous pollutants [4].

In the Philadelphia region, particulate matter monitoring is funded by the Environmental Protection Agency (EPA), and operated by the Philadelphia Air Management Service (AMS) [5]. The AMS's main system of monitoring is based on the EPA's Federal Reference Method; this is the method of passing a sample of air through a filter, manually removing the filter, weighing it, and determining the concentration of hazardous particulate matter in the given sample (in $\mu g/m^3$; see Table 1). This method gives extremely accurate results, but is both expensive and time consuming. Additionally, the filters yield only one reading per day, as opposed to many readings complete with time stamps, which would allow for the investigation of any hazardous event that would cause a spike in particulate matter. The cost for a technician to drive to, replace, and transport a filter can cost approximately $100-$200 each time [6]. In addition, the cost to set up a regional weighing lab is approximately $25,000 to $50,000. These costs are incurred by the individual states, which draw from a shared pool of funds totaling approximately $40 million each year [7].

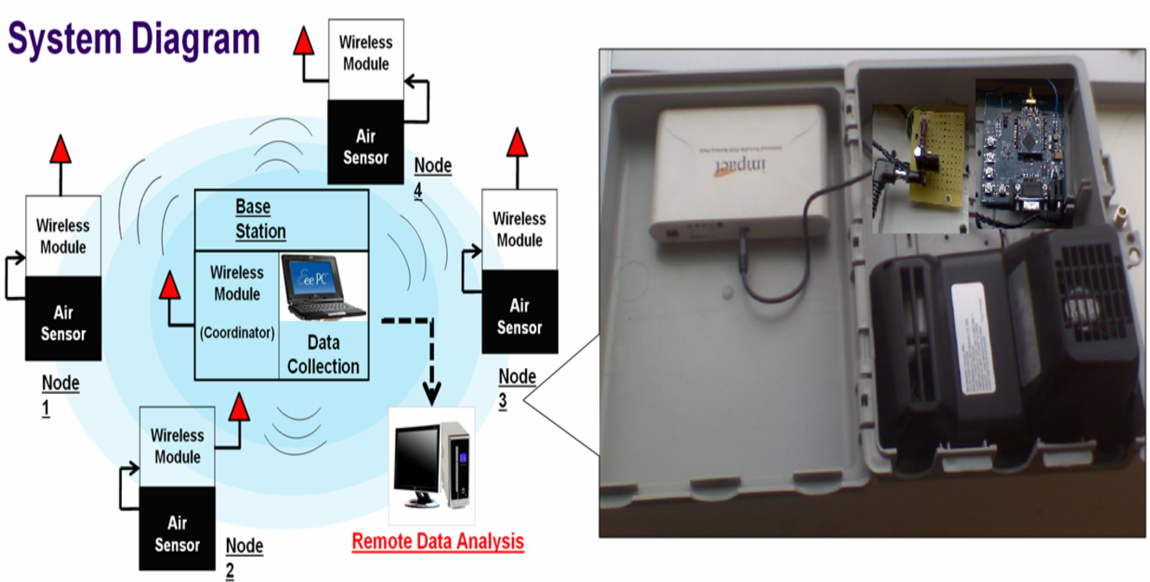


Figure 1: Overview of System Data Flow and Actual Node Configuration

National Ambient Air Quality Standards

	Primary Standards		Secondary Standards	
Pollutant	Level	Averaging Time	Level	Averaging Time
Particulate Matter (PM_{10})	150 µg/m^3	24-hour [3]	Same as Primary	
Particulate Matter ($PM_{2.5}$)	15.0 µg/m^3	Annual [4] (Arithmetic Mean)	Same as Primary	
	35 µg/m^3	24-hour [5]	Same as Primary	

Table 1: EPA National Ambient Air Quality Standard

Data retrieved from a filter takes approximately two weeks to process, due to the labor involved. For this reason, the AMS has installed "continuous" particulate matter monitors (Met-One Instruments, Model: BAM-1020) at some of its testing sites. While these devices can give an instantaneous PM concentration, they are labeled only "Federal Equivalent Method" certified by the EPA, meaning that they can not be used as the final judge of air quality against the EPA standard. Furthermore, the devices are expensive, costing $14,300 each [8, 9].

The preliminary screening system that has been developed is a cost effective alternative to the current method of evaluating the amount of particulate matter in ambient air. When deployed in a given neighborhood, it will determine whether or not the cost and time of the more precise EPA funded tests should be invested.

This project is being performed to the specifications of the Clean Air Council of Philadelphia (CAC). The CAC has requested assistance in developing a system to be deployed in Southeast Philadelphia, due to its proximity to Packer Marine Terminal; the CAC has developed a relationship with neighborhoods within this region that have allowed for the deployment of the system [10]. The work has also been performed with the help of a team of Science Leadership Academy high school students, as part of the Drexel Engineering Projects in Community Service (EPICS) Program.

Final Design

Due to the problems present in the current testing methods, this system has been developed for preliminary screening of ambient air to determine the necessity of more comprehensive tests. The system conserves costs, time, and manual labor, and provides effective feedback for the selected neighborhoods in Southeast Philadelphia to determine if the EPA should be contacted for evaluation. The system consists of both hardware and software components working together. An overview of the system, as well as a sample node, can be seen in Figure 1. A more detailed technical analysis can be found in Appendix H.

The system's air quality sensor solution is the Dylos DC1100 Pro (Figure 2), which collects a sample of air and determines how many particles within a specified size range are present in the air sample. (To compare to the NAAQS the two size ranges evaluated were 0.5-2.5 μm and 2.5 – 10 μm). The Dylos sensor was an ideal solution due to its small size, and low cost.

Figure 2: Dylos DC1100 Pro

Data gathered by the Dylos sensor is transmitted across a wireless network to a base station. The wireless network utilized was the Digi XBee Pro DigiMesh 900 series (Figure 3), selected for its minimal power consumption, effective sleep mode, and user friendly configuration settings.

Each node's sensor and wireless module is powered by an Impact BPD-8000 Rechargeable Battery (Figure 5). Using the battery in conjunction with a voltage regulator circuit allows the node to operate in the field for approximately six days.

The base station utilized to collect data from each node is an Asus Eee PC (Figure 4). The device features an on-board monitor and keyboard, minimizing the complexity of on-site troubleshooting. Loaded onto the base station is a developed software package used to appropriately analyze and display the collected data.

A data conversion algorithm was developed to convert from the Dylos sensor's output (particle count/.01 ft^3) to concentration ($\mu g/m^3$) as specified by the EPA standards. The algorithm is coded into the software package; more information can be found in the Particle Count to Concentration Conversion Algorithm section.

Throughout the design process, specific deliverables have been determined and followed toward system completion. There have been several challenges, which are also detailed in this report. These challenges, however, have been addressed and resolved in the fabrication of the final system.

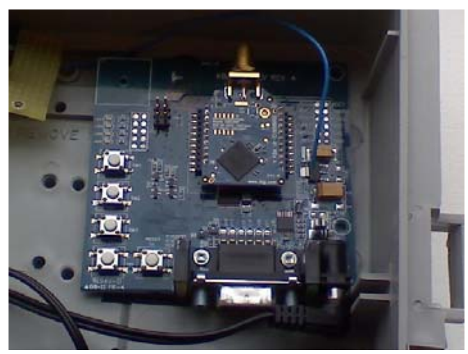


Figure 3: Xbee Pro DigiMesh 900

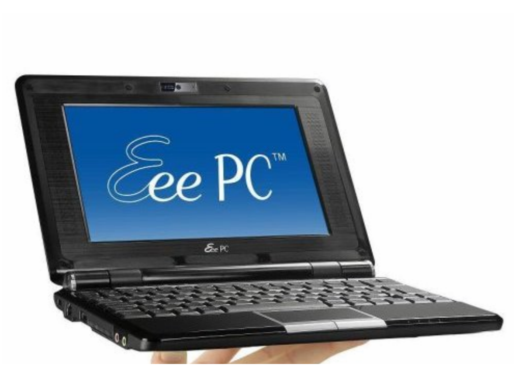


Figure 4: Base Station - Asus Eee PC

Figure 5: Impact BPD-8000 Battery

Deliverables

- Hardware
  - Multiple node network (minimum of 4)
    - Each node consists of an air sensor and wireless networking component
  - Base station to collect data
- Software
  - Networking
  - Data collection and storage
  - Data analysis algorithm
    - Convert data collected by air sensors into appropriate form for comparison to EPA Standard
- Documentation
  - User Manuals
    - Written to inform client on system use
  - Engineering Manuals
    - Written to inform future engineers who may build upon system
- Additional Deliverables that Have Been Added
  - Design and fabrication of node enclosures
  - Comparison of collected data to EPA/ AMS collected data

AMS Test Deployment – Calibration and Validation

The design team worked with the Air Management Service (AMS) to deploy two sensors at the AMS Lycoming Avenue testing site; here, comparison to the Environmental Protection Agency's accepted methods of measurement for $PM_{2.5}$ took place. This experience allowed the team to determine its sensors accuracy, as well as calibrate and improve the derived particle count to concentration conversion algorithm (specific details regarding the algorithm can be found in the next section).

There are several important notes regarding the AMS deployment. At the site, the design team was able to compare its system to the EPA Federal Reference Method (filters) and the EPA Federal Equivalent Method (continuous particulate matter monitor) previously discussed. Because the filter data takes several weeks to obtain, comparison to the EPA Federal Equivalent Method (FEM) was the primary factor used in improving the algorithm. The continuous monitor utilized in the FEM comparison was the Met-One Instruments BAM-1020, a large device which costs approximately $14,300 [9]. Studies have shown a very strong correlation between the FEM readings and the actual filter readings, and concluded that both the FEM and the filter readings "can be used for State regulatory purposes." [21]

The first sensor was deployed at the AMS site from April 3, 2009 to May 5, 2009. The second sensor was deployed at the site from April 21, 2009 to May 5, 2009, and was used to validate the improvements to the algorithm made using Sensor 1. Due to the two week lag time in the EPA's Federal Regulatory Method (filter) data, the team was only given filter data from April 5 to April 16, 2009.

Figure 13 (Appendix D) compares the AMS data (both the limited amount of filter data, and the continuous monitor data) to the design team's uncorrected data, collected using Sensor 1. (Uncorrected data refers to particle count data passed through the initially derived conversion algorithm, described in the next section, before the use of any correction factors).

This data would serve as the basis for the development of correction factors, needed to improve the particle count to conversion algorithm. Using this data, the team performed an extensive analysis of various weather factors that could impact data; this analysis was performed using a developed software package which gleaned historical weather data from WPHL17 television's website. In addition to determining that the presence of rain "cleaned the air" and led to good correlation between collected data, the team also determined that humidity (Figure 14, Appendix D) played a key role in mass underestimation. These observations were supported by Lee et. al., who noted that in the presence of high humidity, particles take on water and gain mass [20]. This phenomenon would not be accounted for by a laser particle counter (which simply counts the number of particles) and thus needed to be factored into the particle count to concentration conversion algorithm.

The design team developed correction factors (detailed in the next section) to compensate for humidity and rain in its particle count to concentration conversion algorithm. Figure 15 (Appendix D) shows the result of this modification; a very strong correlation can be seen between the AMS continuous data and the design team's corrected data. In an attempt to validate these modifications (show that the algorithm is useful for a different sensor), a second sensor was deployed at the AMS site. Figure 16 (Appendix D) shows that the modifications are useful and acceptable on each of the design team's sensors.

Although the amount of filter data collected was limited to 12 daily readings, it was still vital to compare the design team's continuous data, as well as the AMS continuous data, to these Federal Regulatory Method readings. Figure 17 and Table 5 (Appendix D) show the comparison of the daily average of the continuous monitors to the daily filter readings; based on this small sample, the design team's system outperformed the AMS continuous monitor.

Though the deployment at the AMS facility was crucial in the finalization of the design team's system, there were several key issues that must be mentioned:

- Only one month of data (during one season) was collected, due to the time restrictions of the design project. The AMS has collected data for several years, during all seasons, yet is still working to improve its own continuous monitoring methods.
- The AMS did not allow the design team access to its PM_{10} monitors. As a result, further work can be done to collect and evaluate PM_{10} data.
- The design team sought to minimize false alarms, or instances where its data exceeded the EPA limit of 35 μg/m^3 but the AMS monitors did not. During their respective deployment periods, Sensor 1 operated at a false alarm rate of 0.76% (113 false alarms out of 14,933 total readings), while Sensor 2 operated at a false alarm rate of .23% (17 false alarms out of 7,255 total readings).

- The design team's sensor package proved to be much more compact than the AMS continuous monitor (Figure 18, Appendix D) and far less complex than the AMS filter method (Figure 19, Appendix D).

Particle Count to Concentration – Conversion Algorithm

Deriving Algorithm and Code Development

Because the sensor used to measure air quality is a laser particle counter, the output data must be altered so that it can be directly compared to the EPA standard (Table 1). Specifically, an algorithm is needed to convert from particles/.01 ft^3 (the output of the Dylos DC1100 Pro) to $\mu g/m^3$. Similar procedures have been done previously, with a high level of success [12]. It is important to note that this conversion is only a strong approximation; because it is impossible to quantify the exact properties of each of the thousands of microscopic particles being counted, several assumptions are made in the calculation. The algorithm developed assumes:

- All particles are spherical, with a density of 1.65E12 $\mu g/m^3$ [12]
- The radius of a particle in the $PM_{2.5}$ channel is .44 µm [20]
- The radius of a particle in the PM_{10} channel is 2.60 µm [20]

From the above assumptions, the volume, and thus the mass of a particle in both size channels can be calculated (approximated). Multiplying the number of particles per volume by the mass per particle yields a concentration, which can be converted into appropriate units ($\mu g/m^3$) for comparison with the EPA standard.

As described in the previous section, correction factors were utilized to augment this conversion method, based on the presence of humidity and rain. Specifically, the improved algorithm can be described by the relation F= O x H x C, where O is the output of the initial conversion from particle count to concentration (described above), H is the relative humidity percentage, C is the correction factor, and F is the final output concentration. Table 2 shows the values of H and C in dry (no rain) conditions, while Table 3 shows the values of H and C in the presence of rain.

H	C
0-39%	13
40-49%	8
50-59%	6
60-69%	4
70-79%	1.75
80-89%	1.5
90%-100%	1

Table 2: H and C Factors in Dry Conditions

H	C
50-59%	2.5
60-69%	2.2
70-79%	2
80-89%	1.4
90-100%	0.8

Table 3: H and C Factors in Rain

These factors were developed based on the comparison to the AMS continuous monitor and humidity data as described in the previous section. The code needed to implement the conversion algorithm, and to display data in appropriate fashion, can be seen in Appendix C.

South Philadelphia Deployment

The final step in the development of the preliminary screening system was deployment in the neighborhood specified by the Clean Air Council. The goal of this deployment was to ensure that the system's main objective (a short term, low cost yet effective determination of the necessity of more comprehensive particulate matter testing) could be achieved in a real environment. Deployment was performed with the assistance of the Clean Air Council of Philadelphia, who worked to obtain the permissions necessary for success.

The base station was deployed at the Houston Community Center, located on 8th and Snyder Streets in Southeast Philadelphia (see Figure 20; Appendix E). Because no power source was available on the building's roof, the unit was secured inside an office on the third floor. As a result, the nodes could not maintain a communication link several blocks away, and had to be deployed closer to the base station. The system was deployed and collected data from April 28 to May 4, 2009.

The first node (Location 1) was deployed in a garden facility between 7th and 8th Street on Mercy Street. This location was excellent, as fencing was available; the fencing not only provided a structure from which the node could hang, but also kept pedestrians away from the unit.

Two additional nodes were deployed on the roof of the Houston Community Center. One node (Location 2) faced 8th Street; the other (Location 3) faced Snyder Street. These units were hung approximately 30 feet from one another, and overlooked high traffic urban streets (Figure XX, Appendix E).

Figure 6 shows the final system results. As previously described, the EPA standard specifies that the *average concentration for a particular day* should not exceed 35 μg/m^3. As seen in Table 4, this condition is met for each day of the deployment period.

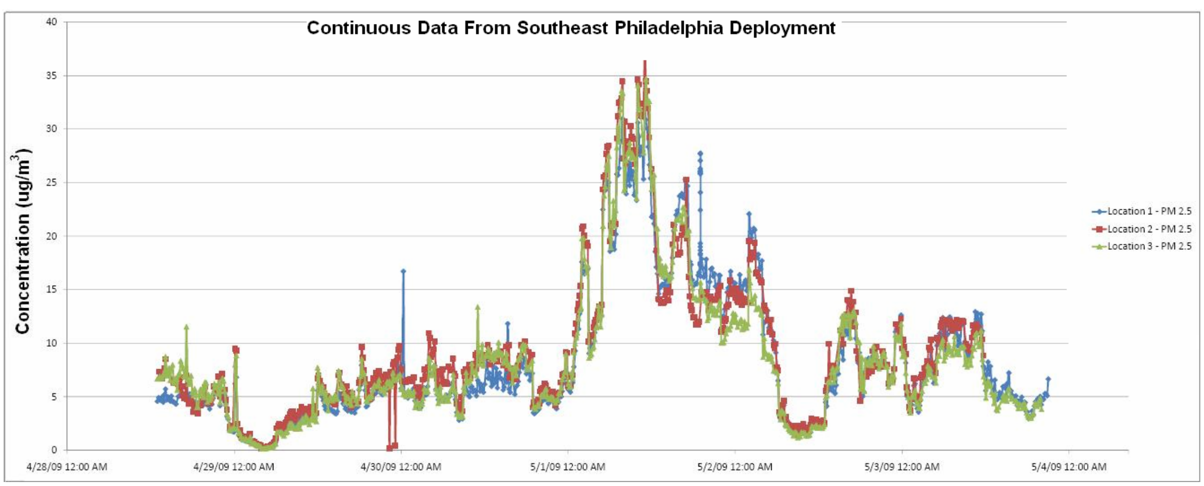


Figure 6: Continuous Data From Southeast Philadelphia System Deployment

Location	WDay	Year	Month	Day	PM 2.5 Average
Garden - 8th and Mercy St.	Wed	2009	4	29	4.61978
Garden - 8th and Mercy St.	Thu	2009	4	30	3.47878
Garden - 8th and Mercy St.	Fri	2009	5	1	5.81711
Garden - 8th and Mercy St.	Sat	2009	5	2	19.1732
Garden - 8th and Mercy St.	Sun	2009	5	3	8.43104
HCC Roof #1 - Facing 8th St.	Wed	2009	4	29	5.37765
HCC Roof #1 - Facing 8th St.	Thu	2009	4	30	4.15855
HCC Roof #1 - Facing 8th St.	Fri	2009	5	1	7.12313
HCC Roof #1 - Facing 8th St.	Sat	2009	5	2	19.2174
HCC Roof #1 - Facing 8th St.	Sun	2009	5	3	8.59338
HCC Roof #2 - Facing Snyder St.	Wed	2009	4	29	5.95159
HCC Roof #2 - Facing Snyder St.	Thu	2009	4	30	3.73218
HCC Roof #2 - Facing Snyder St.	Fri	2009	5	1	6.55181
HCC Roof #2 - Facing Snyder St.	Sat	2009	5	2	18.4497
HCC Roof #2 - Facing Snyder St.	Sun	2009	5	3	7.67917

Table 4: Southeast Philadelphia Daily Particulate Matter Averages

While the EPA daily average condition is met for each day of the deployment period, Figure 6 shows the usefulness of the preliminary screening system as a continuous monitor. For example, during a short time period on May 1, 2009, concentration levels become elevated to within 5 $\mu g/m^3$ of the EPA threshold. Because the EPA only specifies (and using filters, only tests for) a daily average, particulate matter concentration spikes may not be noticed. Using the preliminary screening system as a continuous monitor allows users to pinpoint exactly when and where short term concentration spikes may have occurred. For example, after receiving several weeks of data, a user may notice repeated concentration spikes; the public's repeated exposure to these elevated particulate matter levels could potentially be a cause for concern, and should be investigated by the appropriate authority. At the same time, if only a few spikes are noticed, the events triggering these elevated levels could be evaluated as one time events (fire, building demolition, etc.) that are not a cause for concern.

The high particulate matter concentration spikes seen at all three locations on May 1, 2009 are likely due to the high humidity seen during that particular day (minimum humidity 78%, maximum humidity 100%, and average humidity 90%). However, evaluation of more data would be required to determine whether these spikes are repeated at the same time each week.

Based on the data collected from the Southeast Philadelphia, the design team would not recommend that more expensive and comprehensive testing take place. It would also recommend an investigation into any possible events within the neighborhood that could have caused the elevated concentration levels seen on May 1, 2009. However, it is important to note that the collection of more data using the preliminary screening system can only improve on the accuracy of any recommendations.

Societal, Ethical, and Environmental Impacts

Each day, more hazardous materials are finding their way into ambient air, with fewer state dollars available to provide relief [6]. The result: by simply breathing, more and more people will face adverse health effects in the short and long term. These health effects are not trivial and range from brain and lung damage, to death. With less money available to help test for particulate matter, the developed system will ensure that the high cost of extensive testing is endured only when absolutely necessary; furthermore, more areas can be screened to determine if they are at risk, as opposed to testing them at all due to insufficient funds.

Collaboration with the Clean Air Council has allowed the design team the opportunity to deploy the system at the Houston Community Center (2029 South 8th Street) in Southeast Philadelphia. The design team worked hand in hand with the CAC to ensure that all appropriate deployment procedures were followed. Though the system has been deployed in this specific area, it has been developed for deployment at any location, making it a viable option for a number of locales looking for a cursory determination of the quality of their air.

Through the Drexel EPICS Program, the team has been able to truly influence the educational lives of young engineering students. The Science Leadership Academy students were not only extremely intelligent, but also hungry for knowledge, possessed a strong desire to help impact their community's environment. The design team was pleased to witness a noticeable increase in the students' comprehension of the engineering process, and admired their newfound interest in Drexel University. Photos from the design team's work with the high school students can be seen in Figures 22-23 (Appendix F).

Constraints

<u>Economic</u>

Cost effectiveness was a key goal in design and fabrication of the system, as it attempted to help solve the problem of high costs of current particulate matter testing. Thus, in an attempt to keep costs down, some system components (for example, the voltage regulator circuit) were fabricated by hand, rather than purchased from a vendor. Despite the emphasis on producing a low cost system, the design team does not feel it has sacrificed quality in any component of the system.

<u>Manufacturability</u>

Because the system was intended to be deployed outdoors, the final fabrication was constantly in mind during the design process. From the earliest stages, the team understood that a small to mid size weatherproof enclosure would provide the final housing for the system, thus this constraint was accounted for in the evaluation and selection of system components. Large or unwieldy components that were not conducive to the final fabrication goals were not considered. Furthermore, the design was kept simple and compact, to ensure the success of the final fabrication, and future reproductions.

Sustainability

The system was designed for a long lifespan, and contains no disposable parts; this design goal played a large role in the decision to use a rechargeable battery as the system's main power source. Furthermore, the voltage regulator circuit reduces wear on the system components (for instance, the Dylos sensor's fan unit), ensuring that they are powered on for only a fraction of each hour.

Environmental

As previously noted, the environmental impacts on the system were accounted for in the design process. At the same time, environmental constraints also played a role in the physical deployment of the system; to avoid exposure to rain, it was determined that orienting the enclosure with its vents facing down was the only feasible option. This constraint also limited where the Dylos sensor could be placed within the enclosure, as to maximize the amount of air it can capture, while still shielding the wireless boards from moisture.

Ethical Health and Safety

Because protection of the public was a driving factor in the system's development, ethical health and safety provided motivation as opposed to constraint. At the same time, the team does recognize the need to abide by the ethical reporting code, in the collection of its data; findings should be presented to the appropriate governing bodies, rather than taking results directly to the public forum.

Social

The role of the public in the system's success can not be understated. While some government owned buildings are available to potentially host system components, public participation would allow a far greater area to potentially be screened. In addition to public participation, the design team needed to keep in mind the possibility of individuals tampering with system components. Thus, the decision to hang components only in inconspicuous locations, at heights out of reach of pedestrians, was made.

Political

The design team learned that it must operate under the watchful eye of governing bodies such as the Clean Air Council and Air Management Service. These organizations are the current authorities on public environmental issues; thus, the team needed to be sure that it did not overstep any bounds in its design, development and deployment. For example, the design team needed to get approval by the Air Management Service to observe its data and current testing equipment, and could only operate onsite under their supervision and specification. Most importantly, in collecting data, the design team needed to drive home the point that the preliminary screening system was developed to protect the public and supplement current testing procedures, rather than compete with or "show up" the current government testing.

Timeline

The progression of the design and testing process is shown in Appendix A. The majority of spring term work was spent deploying the system at the AMS Site and in Southeast Philadelphia, analyzing the data collected, finalizing the conversion algorithm, and writing the software code needed to capture and present the data.

Additionally, considerable time was spent working with high school students from the Science Leadership Academy. This time took the form of weekly meeting, in which the design team taught the students about the engineering process, and helped the students develop their own system of testing for deployment in their school.

Budget

The system design budget and industrial budget are captured in Appendix B. The system was designed and fabricated for 8% less than the initial estimate of $4,000, a savings of $335. The final system did not require the purchase of mounting devices, as zip ties were used to hang system components. The team did purchase a second Asus Eee PC, which was deployed at the AMS Site.

Teamwork

The design team worked very well together over the course of the Senior Design Project, in both group and individual sessions. The team members interacted and learned from not only one another, but also two outstanding faculty advisors, a knowledgeable and helpful graduate advisor, representatives from the Clean Air Council, Environmental Protection Agency and product vendors, and a team of motivated high school students from the Science Leadership Academy.

Anthony McClellan served as the team's leader, overseeing all aspects of the design process and ensuring that all responsibilities were delegated and internal deliverables met. Additionally, he worked tirelessly to develop and perfect the extensive software package needed to collect and analyze the massive amount of data collected.

Mark Uva served as the team's wireless component expert; he evaluated a variety of networking options before determining the perfect solution for the task at hand. He also oversaw the budget, and ensured that savings were locked in where possible, enabling the final design to be produced for less than the initial estimate.

Edward Ostapowicz oversaw the system's power specifications including the power budget, power source evaluation, and current draw analysis. Furthermore, he designed and fabricated the crucial voltage regulator circuits, and was also the team's main point of contact to the Clean Air Council.

Robert Falcone derived the initial particle count to conversion algorithm, used within the team's data analysis software package. He also coordinated the written reports and oral presentations, and led meetings with the Science Leadership Academy students.

Conclusion

Considerable progress has been made on the preliminary screening system for particulate matter in ambient air since work began in the fall of 2008. While all stated deliverables have been met in the first fabricated version of the system, there is still room for improvement, based on existing constraints. Suggestions for future work include:

- Further reduce power consumption, to extend battery life
- Perfect mesh networking structure of Xbee hardware, to increase system range
- Improve data conversion algorithm based on further testing
- Utilize creative marketing, in conjunction with test results, to obtain deployment permission from a greater number of residents within a given test area

The design team has learned a great deal about the issue of particulate matter in ambient air, the problems inherent in the current testing solutions, and about the methods of comparison to the accepted standards. Furthermore, knowledge of the engineering process has been gathered firsthand, and will undoubtedly provide useful to the members of the design team in the future.

Technical documentation has been included in this packet for reference. The engineering manual is meant to direct personnel interested in reproducing the final system. A user manual was written to instruct a potential client in the nuances of deploying the completed system in a live field.

References

[1] Dockery, D.W. et al., "An Association Between Air Pollution and Mortality in Six US Cities," *New England Journal of Medicine*, vol. 329, pp. 1753-1759, 1993.

[2] Samet, J.M., Dominici, F., Curriero, F.C., Coursac, I., Zeger, S., "Fine Particulate Air and Mortality in 20 US Cities 1987–1994," *New England Journal of Medicine,* vol. 343, pp. 1742–1749, 2000.

[3] Gauderman, W.J. et al, 2004. "The Effect of Air Pollution on Lung Development from 10 to 18 Years of Age," *New England Journal of Medicine,* vol. 351, pp. 1057–1067, 2004.

[4] Pope III, C.A., Burnett, R.T., Thun, M.J., Calle, E.E., Krewski, D., Ito, K., Thurston, G.D., "Lung Cancer, Cardiopulmonary Morality, and Long-term Exposure to Fine Particulate Air Pollution," *Journal of American Medical Association*, vol. 287, pp. 1132–1141, 2002.

[5] Andrew Hass- Environmental Protection Agency (Private Communication), 2009

[6] Mark Winter- Environmental Protection Agency (Private Communication), 2008

[7] Andrew Hass- Environmental Protection Agency (Private Communication), 2008

[8] Paresh Mehta- Air Management Service (Private Communication), 2009

[9] Sales Representative- Met-One Instruments (Private Communication), 2009

[10] Joseph Otis Minott- Clean Air Council, 2008

[11] Dorsey, R. et al, "Simulation and Quantification of Particulate Matter Emissions in Philadelphia – ECE-17," Drexel University Electrical and Computer Engineering Senior Design, 2008

[12] Tittarelli, T. et al., "Estimation of particle mass concentration in ambient air using a particle counter," *Atmospheric Environment*, vol. 42, pp. 8543-8548, 2008.

[13] American Allergy Supply, "Dylos DC1100 Air Quality Monitor," June 2008, http://www.americanallergysupply.com/dylos.htm

[14] Dylos Corporation, "DC1100 Pro Air Quality Monitor," http://dylosproducts.com/ornodcproair.html

[15] Digi International, "XBee-Pro DigiMesh 900 Product Data Sheet," 2009, http://www.digi.com/products/wireless/zigbee-mesh/xbee-digimesh-900.jsp

[16] Digi International, "XBee-Pro DigiMesh 900 Product User Manual," 2009, http://ftp1.digi.com/support/documentation/90000903_B.pdf

[17] XBee Technician - Digi.com (Private Communication), 2009

[18] Primex's Indoor/Outdoor Enclosures, 2009, http://www.primex.ca/enclosures.htm

[19] L-Com: Global Connectivity, "Weatherproof enclosures," 2009, http://www.l-com.com/content/NEMAEnclosures.html

[20] Lee, J. et al., "Seasonal variations of particle size distributions of PAHs at Seoul, South Korea," *Air Quality Atmospheric Health,* vol. 1, pp. 57-68, 2008.

[21] Quok et al., "Comparison of the ARB Continuous $PM_{2.5}$ Monitoring Network to the $PM_{2.5}$ Federal Reference Method Network," California Environmental Protection Agency, 2006

Appendix A – Work Schedule

Item	Task Name	Start Date	End Date	Time Duration (Days)
1	**Meetings**			
1.1	Weekly Team Meetings	1/7	3/11	10
1.2	Weekly Meetings with advisors	1/9	3/13	10
2	**SLA/Freshman Design Team**			
2.1	Make Presentations	1/5	1/16	12
2.2	Kickoff Meeting	3/9	3/9	1
2.3	SLA Team Presentation/Meetings	1/30	1/30	1
3	**Networking**			
3.1	Order Remaining Parts	1/5	2/14	40
3.2	Develop Network	1/19	2/20	33
3.3	Interface With LPCs	2/16	2/27	12
3.4	Test Network	3/2	3/13	12
4	**Power Considerations**			
4.1	Research Power Options	1/5	1/23	19
4.2	Order Parts	1/26	2/6	12
4.3	Inteface with solution	2/9	2/27	19
4.4	Develop Voltage Regulator Circuit	2/9	3/6	26
4.5	Test Power Consumption With System	3/2	3/13	12
5	**Finish Algorithm**			
5.1	Research Algorithm Further	1/5	1/30	26
5.2	Finalize Algorithm	2/2	3/13	40
6	**Progress Report**			
6.1	Work On Progress Report	2/16	3/4	17
6.2	Work On Presentation	3/4	3/13	10

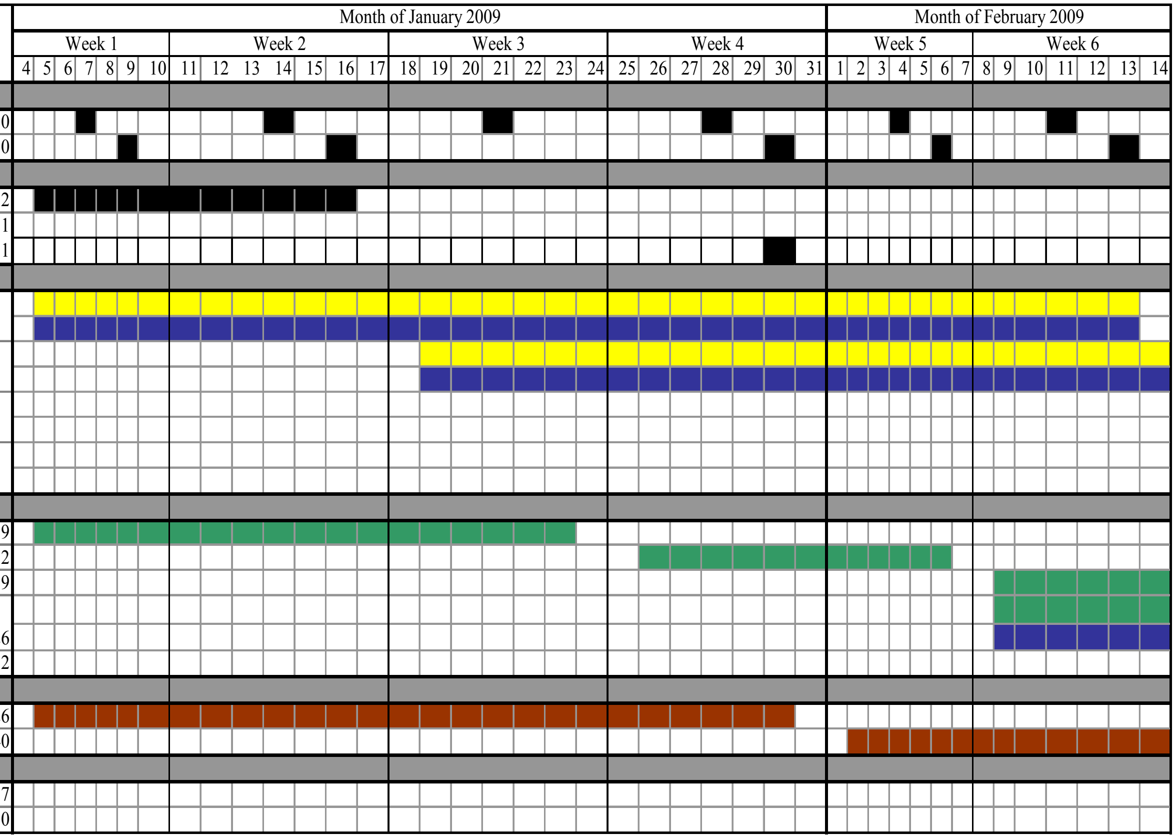
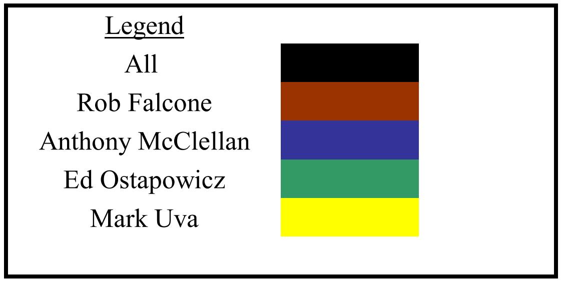


Figure 7: Winter Term Work Schedule (Weeks 1-6)



Item	Task Name	Start Date	End Date	Time Duration (Days)
1	**Meetings**			
1.1	Weekly Team Meetings	1/7	3/11	10
1.2	Weekly Meetings with advisors	1/9	3/13	10
2	**SLA/Freshman Design Team**			
2.1	Make Presentations	1/5	1/16	12
2.2	Kickoff Meeting	3/9	3/9	1
2.3	SLA Team Presentation/Meetings	1/30	1/30	1
3	**Networking**			
3.1	Order Remaining Parts	1/5	2/14	40
3.2	Develop Network	1/19	2/20	33
3.3	Interface With LPCs	2/16	2/27	12
3.4	Test Network	3/2	3/13	12
4	**Power Considerations**			
4.1	Research Power Options	1/5	1/23	19
4.2	Order Parts	1/26	2/6	12
4.3	Inteface with solution	2/9	2/27	19
4.4	Develop Voltage Regulator Circuit	2/9	3/6	26
4.5	Test Power Consumption With System	3/2	3/13	12
5	**Finish Algorithm**			
5.1	Research Algorithm Further	1/5	1/30	26
5.2	Finalize Algorithm	2/2	3/13	40
6	**Progress Report**			
6.1	Work On Progress Report	2/16	3/4	17
6.2	Work On Presentation	3/4	3/13	10

Month of March 2009

Week 7: 15 16 17 18 19 20 21 | Week 8: 22 23 25 25 26 27 28 | Week 9: 1 2 3 4 5 6 7 | Week 10: 8 9 10 11 12 13 14 | Week 11: 15 16 17 18 19 20 21

Figure 8: Winter Term Work Schedule (Weeks 7-11)

Legend
All
Rob Falcone
Anthony McClellan
Ed Ostapowicz
Mark Uva

								Month of April 2009																								
				Time	Week 1							Week 2							Week 3							Week 4						
Item	Task Name	Start Date	End Date	Duration (Days)	29	30	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
2.1	Validation with AMS	3/30	5/12	45																												
3	**Algorithm Finalization**																															
3.1	Finalization of Algorithm	3/30	5/12	45																												
4	**South Philly Deployment/Testing**																															
4.1	South Philly Deployment/Testing	4/28	5/12	15																												
5	**Code Package**																															
5.1	Development of C Package	3/30	5/12	45																												
5.2	Development of MATLAB Code	3/30	5/12	45																												
6	**Final Report**																															
6.1	Final Report Preparation	4/20	5/13	24																												
6.2	Final Presentation Preparation	4/27	5/22	26																												
7	**SLA Students**																															
7.3	Weekly Meetings	4/4	5/22	7																												

Figure 9: Spring Term Work Schedule (Weeks 1-6)

Legend
All
Rob Falcone
Anthony McClellan
Ed Ostapowicz
Mark Uva

										Month of May																						
					Week 5							Week 6							Week 7							Week 8						
Item	Task Name	Start Date	End Date	Time Duration (Days)	26	27	28	29	30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	**Meetings**																															
1.1	Weekly Team Meetings	4/1	5/13	7																												
1.2	Weekly Meetings with advisors	4/3	5/15	7																												
2	**AMS Site Testing**																															
2.1	Validation with AMS	3/30	5/12	45																												
3	**Algorithm Finalization**																															
3.1	Finalization of Algorithm	3/30	5/12	45																												
4	**South Philly Deployment/Testing**																															
4.1	South Philly Deployment/Testing	4/28	5/12	15																												
5	**Code Package**																															
5.1	Development of C Package	3/30	5/12	45																												
5.2	Development of MATLAB Code	3/30	5/12	45																												
6	**Final Report**																															
6.1	Final Report Preparation	4/20	5/13	24																												
6.2	Final Presentation Preparation	4/27	5/22	26																												
7	**SLA Students**																															
7.3	Weekly Meetings	4/4	5/22	7																												

Figure 10: Spring Term Work Schedule (Weeks 7-11)

Legend

- All
- Rob Falcone
- Anthony McClellan
- Ed Ostapowicz
- Mark Uva

Appendix B – Budget

Proposed Budget					Actual Budget				Comparison
Part	QTY	Benefits	Price (Ea)	Total	Part	QTY	Price (Ea)	Total	Cost details
Dylos DC1100 Pro	5	-20%	$305.00	$1,220.00	Dylos DC1100 Pro (Custom Calibration)	5	$304.94	$1,524.70	Higher cost than proposed
Wireless Transceivers (Nodes and Base Station)	6		$200.00	$1,200.00	XBee Pro DigiMesh 900 Development Kit	3	$216.14	$648.42	Lower cost than proposed
Interface Cables (Serial, USB, Etc.)	6		$10.00	$60.00	USB to Serial Cable	1	$10.00	$10.00	Lower cost than proposed *Other cables included in XBee development kit
Single Board Computers	1		$150.00	$150.00	Asus Eee PC	2	$299.00	$598.00	Higher cost than proposed, needed 2 one for AMS and one for BS
Batteries	6		$60.00	$360.00	Batteries	10	$51.84	$518.40	Higher cost than proposed
Antenna	6		$30.00	$180.00					*Included in XBee devlopment kit
Cellular Connection - Phone/ USB Modem/ or PCMCIA Card	1		$150.00	$150.00					*Not used in design
Cellular Service (Per Month)	6	-17%	$39.99	$199.15					*Not used in design
Weatherproof Enclosures	6		$50.00	$300.00	Node Enclosures	5	$38.00	$190.00	Lower cost than proposed
					Base Station Enclosure	1	$45.00	$45.00	Lower cost than proposed
Mounting	6		$30.00	$180.00					*Not used in design
					Velcro	1	$80.00	$80.00	Needed for enclosure to secure parts
					Voltage Regulator Circuit Components	10	$5.00	$50.00	*Not in proposed budget
TOTAL				**$3,999.15**	**TOTAL**			**$3,664.52**	**$334.63**
									Currently under budget

Figure 11: System Design Budget Comparison

Proposed Senior Design 2008/2009 ECE Team 19 Expense Report						
Category	QTY	Benefits	Quarter 1	Quarter 2	Quarter 3	Total
Personel						
Faculty Advisors ($200/hr for 20 hours)	2	30%	$10,400.00	$10,400.00	$10,400.00	$31,200.00
Consultant ($50/hr for 20 hours)	1	30%	$1,300.00	$1,300.00	$1,300.00	$3,900.00
Design Team ($35/hr for 150 hours)	4	30%	$27,300.00	$27,300.00	$27,300.00	$81,900.00
Freshman Engineers ($12/hr for 75 hours)	4	0	$0.00	$3,600.00	$3,600.00	$7,200.00
Total Personel			$39,000.00	$42,600.00	$42,600.00	**$124,200.00**
Hardware						
Laser Particle Counters	5	-20%	$305.00			$1,220.00
Batteries	6			$60.00		$360.00
Weatherproof Enclosures	6			$50.00		$300.00
Mounting Assemblies	6			$30.00		$180.00
Wireless Transceivers (Nodes and Base Station)	6		$200.00			$1,200.00
Antenna	6			$30.00		$180.00
Cellular Connection - Phone/ USB Modem/ or PCMCIA Card	1			$150.00		$150.00
Single Board Computers	1			$150.00		$150.00
Serial Cables	6			$10.00		$60.00
Total Hardware			$2,420.00	$1,380.00	$0.00	**$3,800.00**
Miscellaneous/ Software/Service						
Laptops (Windows XP)	4		$1,500.00			$6,000.00
Internet ($40/mo for 9 months)	1		$120.00	$120.00	$120.00	$360.00
Development Software	1			$2,000.00		$2,000.00
MATLAB (license)	1			$500.00		$500.00
Cellular Service (Data) ($39.99/ mo 6 months)	1	-17%		$99.60	$99.60	$199.20
Total Software			$6,120.00	$2,719.60	$219.60	**$9,059.20**
Subtotal			$47,540.00	$46,699.60	$42,819.60	**$137,059.20**
Overhead		100%				**$137,059.20**
Total			**$95,080.00**	**$93,399.20**	**$85,639.20**	**$274,118.40**

Actual Senior Design 2008/2009 ECE Team 19 Expense Report						
Category	QTY	Benefits	Quarter 1	Quarter 2	Quarter 3	Total
Personel						
Faculty Advisors ($200/hr for 20 hours)	2	30%	$10,400.00	$10,400.00	$10,400.00	$31,200.00
Consultant ($50/hr for 20 hours)	1	30%	$1,300.00	$1,300.00	$1,300.00	$3,900.00
Design Team ($35/hr for 150 hours)	4	30%	$27,300.00	$27,300.00	$27,300.00	$81,900.00
Freshman Engineers ($12/hr for 75 hours)	4	0	$0.00	$3,600.00	$3,600.00	$7,200.00
Total Personel			$39,000.00	$42,600.00	$42,600.00	**$124,200.00**
Hardware						
Dylos DC1100 Pro (Custom Calibration)	5		$304.94	$304.94		$1,524.70
Batteries	10			$51.84		$518.40
Node Enclosures	5			$38.00		$190.00
Base Station Enclosure	1			$45.00		$45.00
XBee Pro DigiMesh 900 Development Kit	3			$216.14		$648.42
Asus Eee PC	2			$299.00		$598.00
Velcro	1			$80.00		$80.00
Voltage Regulator Circuit Components	10			$5.00		$50.00
Serial to USB	1			$10.00		$10.00
Total Hardware			$304.94	$3,279.58	$0.00	**$3,664.52**
Miscellaneous/ Software/Service						
Laptops (Windows XP)	4		$1,500.00			$6,000.00
Internet ($40/mo for 9 months)	1		$120.00	$120.00	$120.00	$360.00
MATLAB (license)	1			$500.00		$500.00
Total Software			$6,120.00	$620.00	$120.00	**$6,860.00**
Subtotal			$45,424.94	$46,499.58	$42,720.00	**$134,644.52**
Overhead		100%				**$134,724.52**
Total			**$90,849.88**	**$92,999.16**	**$85,440.00**	**$269,289.04**

Figure 12: Industrial Budget Comparison

Appendix C – Particle Count to Mass Conversion Algorithm

Import Weather Data

Code Type: VBA Macro

```
Sub Import_Weather()
'Description: This program captures all of the weather information needed for the algorithm (Humidity and rain conditions)
' The location of the network and start/end dates are input on the 'user input' tab
'Updated: 5/12/09
'Creator: Anthony McClellan
'Update User name for each computer
Dim UserName As String
UserName = "Anthony"

'Capture all Date/Location information entered by the user
Sheets("User Inputs").Select
'Start Date
Dim BeginYear As Integer
   BeginYear = Cells(13, "D").Value
Dim BeginMonth As Integer
   BeginMonth = Cells(13, "B").Value
Dim BeginDay As Integer
   BeginDay = Cells(13, "C").Value
Dim BeginDate As String
   BeginDate = BeginYear & "/" & BeginMonth & "/" & BeginDay
'End Date
Dim EndYear As Integer
   EndYear = Cells(18, "D").Value
Dim EndMonth As Integer
   EndMonth = Cells(18, "B").Value
Dim EndDay As Integer
   EndDay = Cells(19, "C").Value
Dim EndDate As String
   EndDate = EndYear & "/" & EndMonth & "/" & EndDay

'Location Information - used to find the best weather based on proximity to source (NE Phila or Phila. Int. Airports)
Dim Location As String
   Location = Cells(13, "G").Value
Dim LocationID As String
   If Location = "Philadelphia International Airport" Then
   LocationID = "/KPHL/"
   End If
   If Location = "NE Philadelphia Airport" Then
   LocationID = "/KPNE/"
   End If

 'Save Entered Dates
 'This could be used to enhance the algorithm file
 'Currently not being used
  Sheets("DO NOT CHANGE2").Select
```

```
  Application.DisplayAlerts = False
  ActiveSheet.SaveAs Filename:="C:\Documents and Settings\" & UserName & "\Desktop\Weather
Output\Entered Dates and Location\Entered Dates and Location.csv", FileFormat:=xlCSV, _
                        CreateBackup:=False
  Application.DisplayAlerts = True

Sheets("User Inputs").Select

'Loop to continue capturing weather data until the program reaches the end date
Do

' Creation of Title used for saving the file
    Dim DateTitle As String
    DateTitle = BeginYear & "-" & BeginMonth & "-" & BeginDay

    Sheets.Add.Name = DateTitle
    Sheets(DateTitle).Select
    'Property used to capture information from myphl17 weather
    With Sheets(DateTitle).QueryTables.Add(Connection:= _
        "URL;http://weather.myphl17.com/auto/wb17/history/airport" & LocationID & BeginDate &
"/DailyHistory.html" _
        , Destination:=Range("$A$1"))
        .Name = "DailyHistory"
        .FieldNames = True
        .RowNumbers = False
        .FillAdjacentFormulas = False
        .PreserveFormatting = True
        .RefreshOnFileOpen = False
        .BackgroundQuery = True
        .RefreshStyle = xlInsertDeleteCells
        .SavePassword = False
        .SaveData = True
        .AdjustColumnWidth = True
        .RefreshPeriod = 0
        .WebSelectionType = xlSpecifiedTables
        .WebFormatting = xlWebFormattingNone
        .WebTables = "6"
        .WebPreFormattedTextToColumns = True
        .WebConsecutiveDelimitersAsOne = True
        .WebSingleBlockTextImport = False
        .WebDisableDateRecognition = False
        .WebDisableRedirections = False
        .Refresh BackgroundQuery:=False
    End With

    'Format time to match with algorithm program
    Columns("A:A").Select
    Selection.NumberFormat = "h:mm;@"

    'Save copy of weather data to bring into algorithm program
    Application.DisplayAlerts = False
    ActiveSheet.SaveAs Filename:="C:\Documents and Settings\" & UserName & "\Desktop\Weather
Output\Output from 'Import Weather Data'\Weather " & DateTitle & ".csv", FileFormat:=xlCSV, _
                        CreateBackup:=False
    Application.DisplayAlerts = True
```

```
    BeginDay = BeginDay + 1

    'Properties used to properly increase date
    'Automatically includes leap year
    If BeginDay = 29 And BeginMonth = 2 Then
    BeginDay = 1
    BeginMonth = BeginMonth + 1
    End If
    'All months with 30 days
    If BeginDay = 31 And BeginMonth = 4 Or BeginDay = 31 And BeginMonth = 6 Or BeginDay = 31 And
BeginMonth = 9 Or BeginDay = 31 And BeginMonth = 11 Then
    BeginDay = 1
    BeginMonth = BeginMonth + 1
    End If
    'All months with 31 days
    If BeginDay = 32 And BeginMonth = 1 Or BeginDay = 32 And BeginMonth = 3 Or BeginDay = 32 And
BeginMonth = 5 Or BeginDay = 32 And BeginMonth = 7 Or BeginDay = 32 And BeginMonth = 8 Or
BeginDay = 32 And BeginMonth = 10 Then
    BeginDay = 1
    BeginMonth = BeginMonth + 1
    End If
    'Increase to next year
    If BeginMonth = 12 And BeginDay = 32 Then
    BeginMonth = 1
    BeginDay = 1
    BeginYear = BeginYear + 1
    End If

    BeginDate = BeginYear & "/" & BeginMonth & "/" & BeginDay


Loop Until BeginDay = EndDay And BeginMonth = EndMonth And BeginYear = EndYear


'Automatically open algorithm program (C++)
 RetVal = Shell("C:\Documents and Settings\" & UserName & "\Desktop\Data and Analysis\Data and
Code\Data Extraction and Algorithm\Debug\Data Extraction and Algorithm.exe", 1)
 Application.DisplayAlerts = False
 Application.Quit
End Sub
```

Data Extraction and Algorithm Application

Code Type: C++

```
//** IMPORTANT - Change all "Anthony" to match user name of Computer (ECETeam19)


//*********************************************************************************
//  Name: Data Extraction Tool and Algorithm Application
//  Description: Overall, this tool will be used to extract the useful information
//          from the RealTerm output text file and apply this data to a particle
//          count to mass algorithm. The output of this program will be saved
//          in a new text file. This file will contain the timestamp, unique
```

```
//             node identifier,node location, small/large particle counts
//             (count/.01ft^3), and concentration (ug/m^3).
//
//  Creator: Anthony McClellan
//  Date Created: 3/31/09
//  Data Modified: 5/12/09
//
//  Additional Notes for User:
//  1. Need to update 'Node Location Table' in Part 4 with actual node locations
//*****************************************************************************



#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctime>
#include <time.h>
#include <direct.h>
#include <windows.h>
#include <cstdlib>


using namespace std;


// main program
int main()
{

        // Information for User
        cout << "** Data Extraction Tool and Algorithm Application **" <<endl <<endl;

        cout << "* Description of Program *" << endl;
        cout << "This tool will read in the RealTerm text file and extact the"<<endl;
        cout <<  "useful data. This data will then be passed through a mass" << endl;
        cout <<  "conversion algorithm. All data will then be saved to a new file. "
                << endl << endl;

        cout << "* Location of RealTerm text file is in folder 'RealTerm Output' *" <<endl;

        cout << "* Output format of new files in folder 'C++ Results' *"<<endl<< endl;



        //Enter Set-up Location
        // 1 - AMS or Demo Setup
        // 2 - Houston Community Center
        cout << "** User Directions **" <<endl;
        cout << "Enter 1 - AMS" << endl;
        cout << "Enter 2 - Houston Community Center" << endl << endl;
        cout << "User, enter number from choices above: ";
```

```
        int choice;
        cin >> choice;
        cout << endl << endl;

//**************************************************************************
// Part 1: Read RealTerm output file - "results.txt"
//              Create New Folder and Files
//      Data index variables defined for row numbering
//      Count Variables used to justify daily averages
//**************************************************************************

// Read in RealTerm Output File
        ifstream input;
        input.open ("C:/Documents and Settings/Anthony/Desktop/RealTerm Output/results.txt",
ios::binary);

//Create New Folder
        mkdir("C:/Documents and Settings/Anthony/Desktop/C++ Results");

// New Output Files - All Headers
// All Data
        ofstream outputAllData;
        //File name
        outputAllData.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/All Data.txt");

        //Header
        outputAllData
          //Title
                << setw(9)  << left << "All Data"
                                << endl << endl
                                //Header
                    << setw(6)  << left << "Index"
                    << setw(6)  << left << ",WDay"
                                << setw(17) << left << ",Date/Time"
                    << setw(6)  << left << ",Node"
                    << setw(36) << left << ",Node Location"
                                << setw(13) << left << ",Humidity(%)"
                                << setw(12) << left << ",Conditions"
                                << setw(13) << left << ",Small Count"
                                << setw(13) << left << ",Large Count"
                                << setw(14) << left << ",Conc(PM 2.5)"
                                << setw(13) << left << ",Conc(PM 10)"
                                << setw(8)  << left << ",Message"
                                << endl;

        // Separate Output Files
        // Location 1
        ofstream outputLoc1;
        outputLoc1.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/Location 1
Data.txt");
        outputLoc1
                                //Title
                << setw(16)  << left << "Location 1 Data"
                                << endl << endl
                                //Header
                  //Header
```

```
                        << setw(6)  << left << "Index"
                        << setw(6)  << left << ",WDay"
                                    << setw(17) << left << ",Date/Time"
                        << setw(6)  << left << ",Node"
                        << setw(36) << left << ",Node Location"
                                    << setw(13) << left << ",Humidity(%)"
                                    << setw(12) << left << ",Conditions"
                                    << setw(13) << left << ",Small Count"
                                    << setw(13) << left << ",Large Count"
                                    << setw(14) << left << ",Conc(PM 2.5)"
                                    << setw(13) << left << ",Conc(PM 10)"
                                    << setw(8)  << left << ",Message"
                                    << endl;

        // Location 2
        ofstream outputLoc2;
        outputLoc2.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/Location 2
Data.txt");
        outputLoc2
                                    //Title
                  << setw(16)  << left << "Location 2 Data"
                                    << endl << endl
                                    //Header
                        //Header
                        << setw(6)  << left << "Index"
                        << setw(6)  << left << ",WDay"
                                    << setw(17) << left << ",Date/Time"
                        << setw(6)  << left << ",Node"
                        << setw(36) << left << ",Node Location"
                                    << setw(13) << left << ",Humidity(%)"
                                    << setw(12) << left << ",Conditions"
                                    << setw(13) << left << ",Small Count"
                                    << setw(13) << left << ",Large Count"
                                    << setw(14) << left << ",Conc(PM 2.5)"
                                    << setw(13) << left << ",Conc(PM 10)"
                                    << setw(8)  << left << ",Message"
                                    << endl;

                // Location 3
        ofstream outputLoc3;
        outputLoc3.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/Location 3
Data.txt");
        outputLoc3
                                    //Title
                  << setw(16)  << left << "Location 3 Data"
                                    << endl << endl
                                    //Header
                        //Header
                        << setw(6)  << left << "Index"
                        << setw(6)  << left << ",WDay"
                                    << setw(17) << left << ",Date/Time"
                        << setw(6)  << left << ",Node"
                        << setw(36) << left << ",Node Location"
                                    << setw(13) << left << ",Humidity(%)"
                                    << setw(12) << left << ",Conditions"
                                    << setw(13) << left << ",Small Count"
```

```
                    << setw(13) << left << ",Large Count"
                    << setw(14) << left << ",Conc(PM 2.5)"
                    << setw(13) << left << ",Conc(PM 10)"
                    << setw(8)  << left << ",Message"
                    << endl;

    // Location 4
    ofstream outputLoc4;
    outputLoc4.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/Location 4
Data.txt");
    outputLoc4
                    //Title
        << setw(16)  << left << "Location 4 Data"
                    << endl << endl
                    //Header
            //Header
            << setw(6)  << left << "Index"
            << setw(6)  << left << ",WDay"
                    << setw(17) << left << ",Date/Time"
            << setw(6)  << left << ",Node"
            << setw(36) << left << ",Node Location"
                    << setw(13) << left << ",Humidity(%)"
                    << setw(12) << left << ",Conditions"
                    << setw(13) << left << ",Small Count"
                    << setw(13) << left << ",Large Count"
                    << setw(14) << left << ",Conc(PM 2.5)"
                    << setw(13) << left << ",Conc(PM 10)"
                    << setw(8)  << left << ",Message"
                    << endl;


    // Location 5
    ofstream outputLoc5;
    outputLoc5.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/Location 5
Data.txt");
    outputLoc5
                    //Title
        << setw(16)  << left << "Location 5 Data"
                    << endl << endl
                    //Header
            //Header
            << setw(6)  << left << "Index"
            << setw(6)  << left << ",WDay"
                    << setw(17) << left << ",Date/Time"
            << setw(6)  << left << ",Node"
            << setw(36) << left << ",Node Location"
                    << setw(13) << left << ",Humidity(%)"
                    << setw(12) << left << ",Conditions"
                    << setw(13) << left << ",Small Count"
                    << setw(13) << left << ",Large Count"
                    << setw(14) << left << ",Conc(PM 2.5)"
                    << setw(13) << left << ",Conc(PM 10)"
                    << setw(8)  << left << ",Message"
                    << endl;
```

```
    // Daily Averages
    ofstream averages;
    averages.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/Averages.txt");
    averages
                        //Title
            << setw(20)  << left << "Daily Averages (All Locations)"
                        << endl << endl
                        //Header
                        << setw(36) << left << "Location"
                << setw(6)  << left << ",WDay"
                        << setw(6)  << left << ",Year"
                        << setw(7)  << left << ",Month"
                        << setw(5)  << left << ",Day"
                << setw(20) << left << ",PM 2.5 Average"
                        << setw(20) << left << ",PM 10 Average"
                        << setw(20) << left << ",Message"
                        << endl;

// Warning Report
    ofstream warning;
    warning.open("C:/Documents and Settings/Anthony/Desktop/C++ Results/Warning Report.txt");
    warning
                        //Title
            << setw(20)  << left << "Warning Report"      << endl
                        << setw(20)  << left << "If PM 2.5 > 35ug/m^3"<< endl
                        << setw(20)  << left << "If PM 10 > 150g/m^3" << endl << endl
                        //Header
                << setw(6)  << left << "WDay"
                        << setw(17) << left << ",Date/Time"
                << setw(36) << left << ",Node Location"
                        << setw(13) << left << ",Humidity(%)"
                        << setw(12) << left << ",Conditions"
                        << setw(20) << left << ",Conc(PM 2.5)"
                        << setw(19) << left << ",Conc(PM 10)"
                        << setw(9)  << left << ",Message"
                        << endl;


// Error if input file does not exist

    if (input.fail())
    {
        cout << "File does not exist. Please place file in correct folder."
            << endl;
        cout << "Program will now end" << endl;
        return 0;
    }

    //Data Point Index (variables) - used to supply row number to each set of data
    int AllDataIndex = 1;
    int Loc1Index = 1;
    int Loc2Index = 1;
    int Loc3Index = 1;
    int Loc4Index = 1;
    int Loc5Index = 1;
```

```
	//Variables to save previous values

	// Location 1
   int countLoc1Data = 1;
	int countLoc1Cycle = 1;
	double sumSmallLoc1 = 0;
	double sumLargeLoc1 = 0;
	string prevYear1;
	string prevMonth1;
	string prevDay1;

	// Location 2
	int countLoc2Data = 1;
   int countLoc2Cycle = 1;
	double sumSmallLoc2 = 0;
	double sumLargeLoc2 = 0;
	string prevYear2;
	string prevMonth2;
	string prevDay2;

	// Location 3
	int countLoc3Data = 1;
   int countLoc3Cycle = 1;
	double sumSmallLoc3 = 0;
	double sumLargeLoc3 = 0;
	string prevYear3;
	string prevMonth3;
	string prevDay3;

	// Location 4
	int countLoc4Data = 1;
   int countLoc4Cycle = 1;
	double sumSmallLoc4 = 0;
	double sumLargeLoc4 = 0;
	string prevYear4;
	string prevMonth4;
	string prevDay4;

	// Location 5
	int countLoc5Data = 1;
   int countLoc5Cycle = 1;
	double sumSmallLoc5 = 0;
	double sumLargeLoc5 = 0;
	string prevYear5;
	string prevMonth5;
	string prevDay5;

	string condition;

//***************************************************************************
// Part 2: Exclude useless lines from RealTerm Output
//				(These lines show only the sensor label "Dylos DC1100" and NO data)
//***************************************************************************

	// Read text file until 'end of file'
	while(!input.eof())
```

```
{
        string line;
        getline(input,line);

        // Exclude lines ending in Dylos DC1100
        int dylosPos = line.find ("Dylos");
        if (dylosPos == -1)
        {
                // Exclude lines with errors in size
                // These contain two identifiers in the same line but not two
                // sets of data (identifier contain "@", such as @H)
                int identifierFrontLoc = line.find ("@");
                int identifierBackLoc = line.rfind ("@");

                if (identifierFrontLoc == identifierBackLoc)
                {
                        // Exclude lines with no identifier included in line
                        // This occurs when the data is not fully received.
                        // (This usually occurs at the last line of the file)

                        if(identifierFrontLoc >= 0)
                        {

// **All important lines are now included from the RealTerm Output file. The useful
//   information from each line now will be extracted.


//*****************************************************************************
// Part 3: Extraction of useful data from RealTerm Output
//                      (Including Time stamp, node identifierm small count, large count)
//*****************************************************************************

        //*****************************************************************************
        // Part 3a: Extraction of Time Stamp. Reformat to include day, month, time
        //*****************************************************************************
                        // Extract Unix Time Stamp (find first ",")
                                int firstMarkerLoc = line.find (",");
                                string timeStamp = line.substr(0,firstMarkerLoc);

                                //Convert time stamp to Int
                                int timeStampInt = atoi(timeStamp.c_str());

                                //Add 4 hours since problem below = 14400 seconds
                                timeStampInt+=14400;

                        //Convert stamp to readable format
                         time_t timeT = (time_t)timeStampInt;
                         char *time = ctime(&timeT);

                         //Change format from char to string
                        string timeString (time);
                        int timeStringSize = timeString.size();
                        string timeStringSub = timeString.substr(0, timeStringSize - 1);


                        //Week Day Identifier (Sun - Mon)
```

```
string weekDayID = timeStringSub.substr(0,3);

//Date Identifiers (Monday/Day/Year)
        //Month Identifier
        string monthID = timeStringSub.substr(4,3);
        if (monthID == "Jan")
                monthID = "1";
        else if (monthID == "Feb")
                monthID = "2";
        else if (monthID == "Mar")
                monthID = "3";
        else if (monthID == "Apr")
                monthID = "4";
        else if (monthID == "May")
                monthID = "5";
        else if (monthID == "Jun")
                monthID = "6";
        else if (monthID == "Jul")
                monthID = "7";
        else if (monthID == "Aug")
                monthID = "8";
        else if (monthID == "Sep")
                monthID = "9";
        else if (monthID == "Oct")
                monthID = "10";
        else if (monthID == "Nov")
                monthID = "11";
        else if (monthID == "Dec")
                monthID = "12";
        else
                monthID = "ERROR";

        //Day Identifier
        string dayID = timeStringSub.substr(8,2);
        string dayIDFirst = timeStringSub.substr(8,1);

        if(dayIDFirst == "0")
                dayID = timeStringSub.substr(9,1);

        //Year Identifier
        string yearID = timeStringSub.substr(20,4);

        //Hour Identifier
        string hourID = timeStringSub.substr(11,2);
        string hourIDFirst = timeStringSub.substr(11,1);

        if(hourIDFirst == "0")
                hourID = timeStringSub.substr(12,1);

        string hourIdent = hourID + ":";

        //Convert hour ID to integer
        int hourIDInt;
        hourIDInt = atoi(hourID.c_str());

        //Minute Identifier
```

```
                    string minuteID = timeStringSub.substr(14,2);

        //***************************************************************************
        // Part 3b: Extraction of Node Identifier
        //***************************************************************************

                // Extract Node Identifier (find "@")
                // Capture 2 character identifier after @
                    int secondMarkerLoc = line.find ("@");
                    string nodeIdentifier = line.substr(secondMarkerLoc, 3);

                    //Location of þ
                    int thirdMarkerLoc = line.find("þ", firstMarkerLoc + 1);

                    //Location of second comma
                    int fourthMarkerLoc = line.find(",", firstMarkerLoc + 1);

        //***************************************************************************
        // Part 3c: Extraction of Particle Counts
        //***************************************************************************

                // Extract Small Particle Count (PM 2.5)
                    //istringstream smallCount;
                    string smallCount = line.substr(thirdMarkerLoc + 2,
                                    (fourthMarkerLoc - thirdMarkerLoc));

                // Extract Large Particle Count (PM 10)
                    int lineSize = line.size(); // Size of entire line
                    string largeCount = line.substr(fourthMarkerLoc + 1,
                                        lineSize - fourthMarkerLoc-1);


            // Particle Count - Convert from string to integer
                    int smallCountInt, largeCountInt;

                    smallCountInt = atoi(smallCount.c_str());
                    largeCountInt = atoi(largeCount.c_str());


    //***************************************************************************
    // Part 4: Node Location Lookup Table
    //      (Connects node identifier output file to physical placements of the device)
    //***************************************************************************

            // This table will be updated once all node identifiers
        // and all node locations have been identifier

        string location;
        int locationNumber = 0;

    //Type of Deployment
        //AMS Location or Demo
        if (choice == 1)
        {
```

```
			if(nodeIdentifier == "@H|")
			{
				location = "Under MetOne Sensor";
				locationNumber = 1;
			}
			else if (nodeIdentifier == "@-H")
			{
				location = "Attached to MetOne Sensor";
				locationNumber = 2;
			}
		}

		//Houston Community Center Set Up
		if (choice == 2)
		{
			if(nodeIdentifier == "@K#")
			{
				location = "Garden - 8th and Mercy St.";
				locationNumber = 1;
			}
			else if (nodeIdentifier == "@K\"")
			{
				location = "HCC Roof #1 - Facing 8th St.";
				locationNumber = 2;
			}
			else if (nodeIdentifier == "@Uº")
			{
				location = "HCC Roof #2 - Facing Snyder St.";
				locationNumber = 3;
			}
//			else if (nodeIdentifier == "@C#")
//			{
//				location = "Location 4";
//				locationNumber = 4;
//	}
//			else if (nodeIdentifier == "@D#")
//			{
//				location = "Location 5";
//				locationNumber = 5;
//			}
			else
			{
				location = "ERROR";
			}
		}
//***************************************************************************
// Part 5a: Conversion Algorithm - Particle Count (per 0.1ft^3) to Mass (per m^3)
//	Original: Rob Falcone, Updates:Anthony McClellan
//***************************************************************************

double r25 = 0.44 * pow(10.0,-6.0); //um, reference Lee paper
double r10 = 2.6 * pow(10.0,-6.0); //um, reference Lee paper

const double PI = 3.14159;

double vol25= (4.0/3.0) * PI * pow(r25, 3.0);
```

```
double vol10= (4.0/3.0) * PI * pow(r10, 3.0);

double density = 1.65 * pow(10.0,12.0); //ug/m3, reference titarelli paper

double mass25=density*vol25; //ug
double mass10=density*vol10; //ug

//dylos output = particles/.01 ft^3
//35.315 ft^3 = 1 m^3
//35.315/.01=3531.5/m^3

double K = 3531.5; // per m^3


//matrix = [1620 58; 1700 52; 9000 107; 1840 75; 1968 85; 1654 75; 1730 52];
//PC25 = matrix(:,1);
//PC10 = matrix(:,2);

//PM2.5 concentration (particle count from channel 1)
double conc25First = smallCountInt*K*mass25; // ug/m^3
double conc25;
//PM10 concentration (particle count - sum of small and large concentrations)
double concLarge = (largeCountInt)*K*mass10; // ug/m^3
double conc10;



//*********************************************************************************
// Part 5b: Weather lookup for PM2.5 Correction Factor
//          NOTE: This section is only to improve the quantity for PM2.5.
//                                  This process can not be used for PM10 since we were unable
//                                  to obtain PM10 measurements from AMS.
//
// IMPORTANT - This section will only work properly if all weather data has been imported
//                                 using the Excel Macro "Import Weather Data."
//*********************************************************************************

// Read in Weather File
        string ext = ".csv";
        string dash = "-";
        string weather = "Weather ";
  string directory = "C:/Documents and Settings/Anthony/Desktop/Weather Output/Output from 'Import
Weather Data'/";
        string fileName = directory + weather + yearID + dash + monthID + dash + dayID + ext;
        double humidity;

        ifstream weatherFile;
        weatherFile.open(fileName.c_str(), ios::binary);

        if (weatherFile.fail())
        {
                cout << "Weather error - " << endl;
                cout << "Weather file does not exist. Please FIRST run 'Import Weather Data' for all
dates needed."
                     << endl;
                cout << "Program will now end" << endl;
```

```
		return 0;
	}

	// Read text file until 'end of file'
	while(!weatherFile.eof())
	{
		string weatherLine;
		getline(weatherFile,weatherLine);

		// Exclude header line
	   int headerLoc = weatherLine.find ("Time");
		if (headerLoc == -1)
		{
			//Find weather hour
			int hourFind = weatherLine.find(":");
			string wHour = weatherLine.substr(0,hourFind);
			//convert hour to int
			int hourFinalInt = atoi(wHour.c_str());

			//Ideal Case:  IF weather hour matches hour for sensor
			//Alternative: Take humidity reading from next hour
			//if(hourFinalInt == hourIDInt || hourFinalInt+1 == hourIDInt)
			if(hourFinalInt == hourIDInt)
			{
				//Comma Locations 1-4
				//Comma 1 Location
				int comma1Loc = weatherLine.find(",");
				//Comma 2 Location
				int comma2Loc = weatherLine.find(",", comma1Loc+1);
				//Comma 3 Location
				int comma3Loc = weatherLine.find(",", comma2Loc+1);

				//Find Humidity Reading
				string humidityString = weatherLine.substr(comma3Loc+1,4);
				int percentLoc = humidityString.find("%");
				string humidityValue = humidityString.substr(0,percentLoc);
				//Convert humidity to an integer
				// Particle Count - Convert from string to integer to double
				int humidityInt = atoi(humidityValue.c_str());
				humidity = atof(humidityValue.c_str());

				//Comma 4 Location
				int comma4Loc = weatherLine.find(",", comma3Loc+1);
				//Comma 5 Location
				int comma5Loc = weatherLine.find(",", comma4Loc+1);
				//Comma 6 Location
				int comma6Loc = weatherLine.find(",", comma5Loc+1);
				//Comma 7 Location
				int comma7Loc = weatherLine.find(",", comma6Loc+1);
				//Comma 8 Location
				int comma8Loc = weatherLine.find(",", comma7Loc+1);
				//Comma 9 Location
				int comma9Loc = weatherLine.find(",", comma8Loc+1);
				//Comma 10 Location
				int comma10Loc = weatherLine.find(",", comma9Loc+1);
```

```
//Find Condition
string condString = weatherLine.substr(comma10Loc+1,6);
int condLoc = condString.find(",");
string condition = condString.substr(0,condLoc);

//Apply separate factor for rain
if(condition == "Rain ")
{
        //Rain Table - Improvement Factor
        double rain50 = 2.5; //50-59%
        double rain60 = 2.2; //60-69%
        double rain70 = 2.0; //70-79%
        double rain80 = 1.4; //80-89%
        double rain90 = 0.8; //90-100%

        //Humidity between 50-59%
        if(humidityInt >= 50 && humidityInt <= 59)
        {
                conc25 = conc25First*(humidity/100)*rain50;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 60-69%
        else if(humidityInt >= 60 && humidityInt <= 69)
        {
                conc25 = conc25First*(humidity/100)*rain60;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 70-79%
        else if(humidityInt >= 70 && humidityInt <= 79)
        {
                conc25 = conc25First*(humidity/100)*rain70;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 80-89%
        else if(humidityInt >= 80 && humidityInt <= 89)
        {
                conc25 = conc25First*(humidity/100)*rain80;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 90-100%
        else if(humidityInt >= 90 && humidityInt <= 100)
        {
                conc25 = conc25First*(humidity/100)*rain90;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
```

```
else
        cout << "Error - see humidity correction section of
code - (S1 Rain)"<<endl;

}

else if (condition == "  ")
{
        //NO Rain Table - Improvement Factor
        double factor0 = 13; //0-39%
        double factor40 = 8; //40-49%
        double factor50 = 6; //50-59%
        double factor60 = 4; //60-69%
        double factor70 = 1.75; //70-79%
        double factor80 = 1.5; //80-89%
        double factor90 = 1; //90-100%

        //Humidity between 0-39%
        if(humidityInt > 0 && humidityInt <= 39)
        {
                conc25 = conc25First*(humidity/100)*factor0;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 40-49%
        else if(humidityInt >= 40 && humidityInt <= 49)
        {
                conc25 = conc25First*(humidity/100)*factor40;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 50-59%
        else if(humidityInt >= 50 && humidityInt <= 59)
        {
                conc25 = conc25First*(humidity/100)*factor50;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 60-69%
        else if(humidityInt >= 60 && humidityInt <= 69)
        {
                conc25 = conc25First*(humidity/100)*factor60;
                conc10 = (conc25 + concLarge);
                goto stop;

        }
        //Humidity between 70-79%
        else if(humidityInt >= 70 && humidityInt <= 79)
        {
                conc25 = conc25First*(humidity/100)*factor70;
                conc10 = (conc25 + concLarge);
                goto stop;
```

```
                                        }
                                        //Humidity between 80-89%
                                        else if(humidityInt >= 80 && humidityInt <= 89)
                                        {
                                                conc25 = conc25First*(humidity/100)*factor80;
                                                conc10 = (conc25 + concLarge);
                                                goto stop;

                                        }
                                        //Humidity between 90-100%
                                        else if(humidityInt >= 90 && humidityInt <= 100)
                                        {
                                                conc25 = conc25First*(humidity/100)*factor90;
                                                conc10 = (conc25 + concLarge);
                                                goto stop;

                                        }
                                        else
                                        {
                                                cout << "Error - see humidity correction section of
code - (S1 No Rain)"<<endl;
                                        }
                                }
                        }
        }       }

stop:
        weatherFile.close();

//*********************************************************************************
// Part 5c: Error Messages based on Algorithm Output
//*********************************************************************************

        //Add Message to each line
          //If PM2.5 > 35 or PM10 > 150 a warning message is displayed
   //If Lower - then nothing is added

  string message;
        //Warn 3 - both PM2.5 and PM10 over threshold
        if (conc25 > 35 && conc10 > 150)
           message = "Warning PM 2.5 and PM 10 too high";
        //Warn 1 - PM2.5 over threshold
        else if    (conc25 > 35)
                message = "Warning PM 2.5 too high";
        //Warn 2 - PM10 over threshold
        else if (conc10 > 150)
    message = "Warning PM 10 too high";
        // No Warning
        else
           message = " ";

//*********************************************************************************
// Part 6: Write all data to a new files - Warning Report, All data, Locations 1 - 5

//*********************************************************************************
                //Exclude invalid data from sensor (See in Testing)
```

```
if (smallCountInt > 0 && largeCountInt < 1200000000)
{


        //Creation of concatDate
        string concatDate = monthID + "/" + dayID + "/" + yearID;

        //Creation of concatTime
        string concatTime = hourID + ":" + minuteID;
        //Creation of concatDateTime
        string concatDateTime = concatDate + " " + concatTime;


                // Write all information to new file "All Data"
                        outputAllData
                                << setw(6)  << left << AllDataIndex   << ","
                          << setw(5)  << left << weekDayID      << ","
                                << setw(16) << left << concatDateTime << ","
                                << setw(5)  << left << nodeIdentifier << ","
                          << setw(35) << left << location       << ","
                                << setw(12) << left << humidity       << ","
                                << setw(11) << left << condition      << ","
                                << setw(12) << left << smallCountInt  << ","
                                << setw(12) << left << largeCountInt  << ","
                                << setw(13) << left << conc25         << ","
                                << setw(12) << left << conc10         << ","
                                << setw(7)  << left << message
                                << endl;

                        AllDataIndex++; // Increase to next row value (count data
entries)

                // Warning Report
                        string warningMessage;
                if (conc25 > 35 && conc10 > 150)
                        warningMessage = "Warning PM 2.5 and PM 10 too high";
                //Warn 1 - PM2.5 over threshold
                else if    (conc25 > 35)
                        warningMessage = "Warning PM 2.5 too high";
                //Warn 2 - PM10 over threshold
                else if (conc10 > 150)
                        warningMessage = "Warning PM 10 too high";

                if (conc25 > 35 || conc10 > 150)
                {
                        warning
                          << setw(5)  << left << weekDayID      << ","
                                << setw(16) << left << concatDateTime << ","
                          << setw(35) << left << location       << ","
                                << setw(12) << left << humidity       << ","
                                << setw(11) << left << condition      << ","
                                << setw(13) << left << conc25         << ","
                                << setw(12) << left << conc10         << ","
                                << setw(12) << left << warningMessage
                                << endl;
                }
```

```
// Write to separate files - "Location 1 through 5"

        //Location 1
        if (locationNumber == 1)
        {

        outputLoc1
                << setw(6)  << left << Loc1Index      << ","
          << setw(5)  << left << weekDayID      << ","
                << setw(16) << left << concatDateTime << ","
                << setw(5)  << left << nodeIdentifier << ","
          << setw(35) << left << location       << ","
                << setw(12) << left << humidity       << ","
                << setw(11) << left << condition      << ","
                << setw(12) << left << smallCountInt  << ","
                << setw(12) << left << largeCountInt  << ","
                << setw(13) << left << conc25         << ","
                << setw(12) << left << conc10         << ","
                << setw(7)  << left << message
                << endl;


        Loc1Index++; // Increase to next row value (count data
entries)

        if (Loc1Index > 2)
        {
        if (prevDay1 != dayID)
        {
                double smallAverageLoc1 =
sumSmallLoc1/(countLoc1Data-1);
                double largeAverageLoc1 =
sumLargeLoc1/(countLoc1Data-1);


                //Warning messages
                string messageLoc1;
                //Warn 3 - both PM2.5 and PM10 over thresholds
                if (smallAverageLoc1 > 35 && largeAverageLoc1 >
150)
                messageLoc1 = "Warning PM 2.5 and PM 10 too
high";
                //Warn 1 - PM2.5 over 35ug/m^3
                else if    (smallAverageLoc1 > 35)
                messageLoc1 = "Warning PM 2.5 too high";
                //Warn 2 - PM10 over 150ug/m^3
                else if (largeAverageLoc1 > 150)
                messageLoc1 = "Warning PM 10 too high";
                // No Warning
                else
                messageLoc1 = " ";

                averages
                   << setw(35) << left << location        << ","
                   << setw(5)  << left << weekDayID       << ","
                   << setw(5)  << left << yearID                  << ","
```

```
<< setw(6)  << left << monthID         << ","
<< setw(4)  << left << dayID           << ","
        << setw(20) << left << smallAverageLoc1
<< ","

<< setw(20) << left << largeAverageLoc1 << ","
        << setw(20) << left << messageLoc1
<< endl;


    countLoc1Cycle++;
    countLoc1Data = 0;
    sumSmallLoc1 = 0;
    sumLargeLoc1 = 0;
}
}

//Average Calculations
countLoc1Data++;// Count total entries for average
sumSmallLoc1 += conc25; // Sum Concentration of 2.5
sumLargeLoc1 += conc10; // Sum Concentration of 10
//Apply previous variables for average calculation
prevYear1  = yearID;
prevMonth1 = monthID;
prevDay1   = dayID;

}

//Location 2
else if (locationNumber == 2)
    {
    outputLoc2
        << setw(6)  << left << Loc2Index      << ","
    << setw(5)  << left << weekDayID      << ","
        << setw(16) << left << concatDateTime << ","
        << setw(5)  << left << nodeIdentifier << ","
    << setw(35) << left << location       << ","
        << setw(12) << left << humidity       << ","
        << setw(11) << left << condition      << ","
        << setw(12) << left << smallCountInt  << ","
        << setw(12) << left << largeCountInt  << ","
        << setw(13) << left << conc25         << ","
        << setw(12) << left << conc10         << ","
        << setw(7)  << left << message
        << endl;

    Loc2Index++; // Increase to next row value (count data
entries)

    if (Loc2Index > 2)
    {
    if (prevDay2 != dayID)
    {
        double smallAverageLoc2 =
sumSmallLoc2/(countLoc2Data-1);
        double largeAverageLoc2 =
sumLargeLoc2/(countLoc2Data-1);
```

```
//Warning messages
string messageLoc2;
//Warn 3 - both PM2.5 and PM10 over thresholds
if (smallAverageLoc2 > 35 && largeAverageLoc2 >
150)

messageLoc2 = "Warning PM 2.5 and PM 10 too
high";

//Warn 1 - PM2.5 over 35ug/m^3
else if     (smallAverageLoc2 > 35)
messageLoc2 = "Warning PM 2.5 too high";
//Warn 2 - PM10 over 150ug/m^3
else if (largeAverageLoc2 > 150)
messageLoc2 = "Warning PM 10 too high";
// No Warning
else
messageLoc2 = " ";

averages
        << setw(35) << left << location        << ","
  << setw(5)  << left << weekDayID       << ","
  << setw(5)  << left << yearID                 << ","
  << setw(6)  << left << monthID         << ","
  << setw(4)  << left << dayID           << ","
        << setw(20) << left << smallAverageLoc2
<< ","

  << setw(20) << left << largeAverageLoc2 << ","
        << setw(20) << left << messageLoc2
  << endl;

countLoc2Cycle++;
countLoc2Data = 0;
sumSmallLoc2 = 0;
sumLargeLoc2 = 0;
}}
//Average Calculations
countLoc2Data++;// Count total entries for average
sumSmallLoc2 += conc25; // Sum Concentration of 2.5
sumLargeLoc2 += conc10; // Sum Concentration of 10
//Apply previous variables for average calculation
prevYear2  = yearID;
prevMonth2 = monthID;
prevDay2   = dayID;

}


//Location 3
else if (locationNumber == 3)
    {
    outputLoc3
            << setw(6)  << left << Loc3Index      << ","
      << setw(5)  << left << weekDayID      << ","
            << setw(16) << left << concatDateTime << ","
            << setw(5)  << left << nodeIdentifier << ","
      << setw(35) << left << location       << ","
```

```
                    << setw(12) << left << humidity       << ","
                    << setw(11) << left << condition      << ","
                    << setw(12) << left << smallCountInt  << ","
                    << setw(12) << left << largeCountInt  << ","
                    << setw(13) << left << conc25         << ","
                    << setw(12) << left << conc10         << ","
                    << setw(7)  << left << message
                    << endl;

                Loc3Index++; // Increase to next row value (count data
entries)

                if (Loc3Index > 2)
                {
                if (prevDay3 != dayID)
                {
                    double smallAverageLoc3 =
sumSmallLoc3/(countLoc3Data-1);
                    double largeAverageLoc3 =
sumLargeLoc3/(countLoc3Data-1);
                    //Warning messages
                    string messageLoc3;
                    //Warn 3 - both PM2.5 and PM10 over thresholds
                    if (smallAverageLoc3 > 35 && largeAverageLoc3 >
150)
                    messageLoc3 = "Warning PM 2.5 and PM 10 too
high";
                    //Warn 1 - PM2.5 over 35ug/m^3
                    else if    (smallAverageLoc3 > 35)
                    messageLoc3 = "Warning PM 2.5 too high";
                    //Warn 2 - PM10 over 150ug/m^3
                    else if (largeAverageLoc3 > 150)
                    messageLoc3 = "Warning PM 10 too high";
                    // No Warning
                    else
                    messageLoc3 = " ";

                    averages
                        << setw(35) << left << location        << ","
                     << setw(5)  << left << weekDayID       << ","
                     << setw(5)  << left << yearID                 << ","
                     << setw(6)  << left << monthID         << ","
                     << setw(4)  << left << dayID           << ","
                        << setw(20) << left << smallAverageLoc3
<< ","
                     << setw(20) << left << largeAverageLoc3 << ","
                        << setw(20) << left << messageLoc3
                     << endl;

                    countLoc3Cycle++;
                    countLoc3Data = 0;
                    sumSmallLoc3 = 0;
                    sumLargeLoc3 = 0;
                }}
                //Average Calculations
                countLoc3Data++;// Count total entries for average
```

```
sumSmallLoc3 += conc25; // Sum Concentration of 2.5
sumLargeLoc3 += conc10; // Sum Concentration of 10
//Apply previous variables for average calculation
prevYear3  = yearID;
prevMonth3 = monthID;
prevDay3   = dayID;

}

//Location 4
else if (locationNumber == 4)
{
outputLoc4
<< setw(6)  << left << Loc4Index      << ","
<< setw(5)  << left << weekDayID      << ","
<< setw(16) << left << concatDateTime << ","
<< setw(5)  << left << nodeIdentifier << ","
<< setw(35) << left << location       << ","
<< setw(12) << left << humidity       << ","
<< setw(11) << left << condition      << ","
<< setw(12) << left << smallCountInt  << ","
<< setw(12) << left << largeCountInt  << ","
<< setw(13) << left << conc25         << ","
<< setw(12) << left << conc10         << ","
<< setw(7)  << left << message
<< endl;

Loc4Index++; // Increase to next row value (count
data entries)

if (Loc4Index > 2)
{
if (prevDay4 != dayID)
{
double smallAverageLoc4 =
sumSmallLoc4/(countLoc4Data-1);
double largeAverageLoc4 =
sumLargeLoc4/(countLoc4Data-1);

//Warning messages
string messageLoc4;
//Warn 3 - both PM2.5 and PM10 over thresholds
if (smallAverageLoc4 > 35 && largeAverageLoc4 >
150)
messageLoc4 = "Warning PM 2.5 and PM 10 too
high";
//Warn 1 - PM2.5 over 35ug/m^3
else if     (smallAverageLoc4 > 35)
messageLoc4 = "Warning PM 2.5 too high";
//Warn 2 - PM10 over 150ug/m^3
else if (largeAverageLoc4 > 150)
messageLoc4 = "Warning PM 10 too high";
// No Warning
else
messageLoc4 = " ";
```

```
averages
        << setw(35) << left << location        << ","
  << setw(5)  << left << weekDayID       << ","
  << setw(5)  << left << yearID              << ","
  << setw(6)  << left << monthID         << ","
  << setw(4)  << left << dayID           << ","
        << setw(20) << left << smallAverageLoc4
<< ","

  << setw(20) << left << largeAverageLoc4 << ","
        << setw(20) << left << messageLoc4
  << endl;

countLoc4Cycle++;
countLoc4Data = 0;
sumSmallLoc4 = 0;
sumLargeLoc4 = 0;
}}
//Average Calculations
countLoc4Data++;// Count total entries for average
sumSmallLoc4 += conc25; // Sum Concentration of 2.5
sumLargeLoc4 += conc10; // Sum Concentration of 10
//Apply previous variables for average calculation
prevYear4  = yearID;
prevMonth4 = monthID;
prevDay4   = dayID;
}

//Location 5
else if (locationNumber == 5)
    {
    outputLoc5
            << setw(6)  << left << Loc5Index      << ","
      << setw(5)  << left << weekDayID      << ","
            << setw(16) << left << concatDateTime << ","
            << setw(5)  << left << nodeIdentifier << ","
      << setw(35) << left << location       << ","
            << setw(12) << left << humidity       << ","
            << setw(11) << left << condition      << ","
            << setw(12) << left << smallCountInt  << ","
            << setw(12) << left << largeCountInt  << ","
            << setw(13) << left << conc25         << ","
            << setw(12) << left << conc10         << ","
            << setw(7)  << left << message
            << endl;

    Loc5Index++; // Increase to next row value (count data
entries)


    if (Loc5Index > 2)
    {
    if (prevDay5 != dayID)
    {
        double smallAverageLoc5 =
sumSmallLoc5/(countLoc5Data-1);
```

```
                                        double largeAverageLoc5 =
sumLargeLoc5/(countLoc5Data-1);
                                        //Warning messages
                                        string messageLoc5;
                                        //Warn 3 - both PM2.5 and PM10 over thresholds
                                        if (smallAverageLoc5 > 35 && largeAverageLoc5 >
150)
                                        messageLoc5 = "Warning PM 2.5 and PM 10 too
high";
                                        //Warn 1 - PM2.5 over 35ug/m^3
                                        else if    (smallAverageLoc5 > 35)
                                        messageLoc5 = "Warning PM 2.5 too high";
                                        //Warn 2 - PM10 over 150ug/m^3
                                        else if (largeAverageLoc5 > 150)
                                        messageLoc5 = "Warning PM 10 too high";
                                        // No Warning
                                        else
                                        messageLoc5 = " ";

                                        averages
                                             << setw(35) << left << location        << ","
                                          << setw(5)  << left << weekDayID       << ","
                                          << setw(5)  << left << yearID                     << ","
                                          << setw(6)  << left << monthID         << ","
                                          << setw(4)  << left << dayID           << ","
                                             << setw(20) << left << smallAverageLoc5
<< ","
                                          << setw(20) << left << largeAverageLoc5 << ","
                                             << setw(20) << left << messageLoc5
                                          << endl;

                                        countLoc5Cycle++;
                                        countLoc5Data = 0;
                                        sumSmallLoc5 = 0;
                                        sumLargeLoc5 = 0;
                                   }}
                                   //Average Calculations
                                   countLoc5Data++;// Count total entries for average
                                   sumSmallLoc5 += conc25; // Sum Concentration of 2.5
                                   sumLargeLoc5 += conc10; // Sum Concentration of 10
                                   //Apply previous variables for average calculation
                                   prevYear5  = yearID;
                                   prevMonth5 = monthID;
                                   prevDay5   = dayID;
                                   }

                              } //End If

                         } //End If

                        } //End If

                  } //End If

             } //End While
             //Close files
```

```
            input.close();
            outputAllData.close();
    outputLoc1.close();
    outputLoc2.close();
            outputLoc3.close();
            outputLoc4.close();
            outputLoc5.close();
            averages.close();
            // Program Complete
            cout << "* Program Complete *" <<endl;
            cout << "Now execute 'Graphical Analysis Program'" <<endl <<endl;
return 0;
            }
```

Graphical Analysis

Code Type: VBA Macro

```
'Program Name: PM Data Updater
'Description: This macro will need to be run once all PM text files are updated.
'           These files are kept in the folder labeled "C++ Results"
'Creator: Anthony McClellan
'Date of last update: 5/12/09
'ImportTextFile referenced from: http://www.cpearson.com/excel/ImpText.aspx

Sub Graphical_Analysis()

'********************************************************************************
'PART 1 - Import all text files
'        This data is located in the c++ results folder on the desktop
'********************************************************************************
'**************************************************
'Part 1a - Import "All Data.txt" to sheet "All Data"
'**************************************************
'Update User name for each computer
Dim UserName As String
UserName = "Anthony"


'Select sheet
Sheets("All Data").Select
'Clear current data in sheet
Sheets("All Data").Cells.Clear
'Begin copying file on first cell
Range("A1").Select

'All variables
Dim RowNdx As Long
Dim ColNdx As Integer
Dim TempVal As Variant
Dim WholeLine As String
Dim Pos As Integer
Dim NextPos As Integer
Dim SaveColNdx As Integer
```

```
Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\All Data.txt"
Open FName For Input Access Read As #1
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(1)
   Line Input #1, WholeLine
   If Right(WholeLine, 1) <> Sep Then
      WholeLine = WholeLine & Sep
   End If
   ColNdx = SaveColNdx
   Pos = 1
   NextPos = InStr(Pos, WholeLine, Sep)
   While NextPos >= 1
      TempVal = Mid(WholeLine, Pos, NextPos - Pos)
      Cells(RowNdx, ColNdx).Value = TempVal
      Pos = NextPos + 1
      ColNdx = ColNdx + 1
      NextPos = InStr(Pos, WholeLine, Sep)
   Wend
   RowNdx = RowNdx + 1
Wend
Close #1

'Format Time
   Columns("C:C").Select
   Selection.NumberFormat = "[$-409]m/d/yy h:mm AM/PM;@"
'Go to first cell
   Range("A1").Select

'******************************************************************
'Part 1b - Import "Location 1 Data.txt" to sheet "Location 1 - Data"
'******************************************************************

'Select sheet
Sheets("Location 1 - Data").Select
'Clear current data in sheet
Sheets("Location 1 - Data").Cells.Clear
'Begin copying file on first cell
Range("A1").Select

Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\Location 1 Data.txt"
```

```
Open FName For Input Access Read As #2
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(2)
   Line Input #2, WholeLine
   If Right(WholeLine, 1) <> Sep Then
      WholeLine = WholeLine & Sep
   End If
   ColNdx = SaveColNdx
   Pos = 1
   NextPos = InStr(Pos, WholeLine, Sep)
   While NextPos >= 1
      TempVal = Mid(WholeLine, Pos, NextPos - Pos)
      Cells(RowNdx, ColNdx).Value = TempVal
      Pos = NextPos + 1
      ColNdx = ColNdx + 1
      NextPos = InStr(Pos, WholeLine, Sep)
   Wend
   RowNdx = RowNdx + 1
Wend
Close #2

'Format Time
   Columns("C:C").Select
   Selection.NumberFormat = "[$-409]m/d/yy h:mm AM/PM;@"
'Go to first cell
   Range("A1").Select


'*******************************************************************
'Part 1c - Import "Location 2 Data.txt" to sheet "Location 2 - Data"
'*******************************************************************

'Select sheet
Sheets("Location 2 - Data").Select
'Clear current data in sheet
Sheets("Location 2 - Data").Cells.Clear
'Begin copying file on first cell
Range("A1").Select

Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\Location 2 Data.txt"
Open FName For Input Access Read As #3
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(3)
   Line Input #3, WholeLine
   If Right(WholeLine, 1) <> Sep Then
```

```
      WholeLine = WholeLine & Sep
   End If
   ColNdx = SaveColNdx
   Pos = 1
   NextPos = InStr(Pos, WholeLine, Sep)
   While NextPos >= 1
      TempVal = Mid(WholeLine, Pos, NextPos - Pos)
      Cells(RowNdx, ColNdx).Value = TempVal
      Pos = NextPos + 1
      ColNdx = ColNdx + 1
      NextPos = InStr(Pos, WholeLine, Sep)
   Wend
   RowNdx = RowNdx + 1
Wend
Close #3

'Go to first cell
   Range("A1").Select

'*******************************************************************
'Part 1d - Import "Location 3 Data.txt" to sheet "Location 3 - Data"
'*******************************************************************

'Select sheet
Sheets("Location 3 - Data").Select
'Clear current data in sheet
Sheets("Location 3 - Data").Cells.Clear
'Begin copying file on first cell
Range("A1").Select

Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\Location 3 Data.txt"
Open FName For Input Access Read As #4
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(4)
   Line Input #4, WholeLine
   If Right(WholeLine, 1) <> Sep Then
      WholeLine = WholeLine & Sep
   End If
   ColNdx = SaveColNdx
   Pos = 1
   NextPos = InStr(Pos, WholeLine, Sep)
   While NextPos >= 1
      TempVal = Mid(WholeLine, Pos, NextPos - Pos)
      Cells(RowNdx, ColNdx).Value = TempVal
      Pos = NextPos + 1
      ColNdx = ColNdx + 1
      NextPos = InStr(Pos, WholeLine, Sep)
```

```
    Wend
    RowNdx = RowNdx + 1
Wend
Close #4

'Format Time
    Columns("C:C").Select
    Selection.NumberFormat = "[$-409]m/d/yy h:mm AM/PM;@"
'Go to first cell
    Range("A1").Select

'*******************************************************************
'Part 1e - Import "Location 4 Data.txt" to sheet "Location 4 - Data"
'*******************************************************************

'Select sheet
Sheets("Location 4 - Data").Select
'Clear current data in sheet
Sheets("Location 4 - Data").Cells.Clear
'Begin copying file on first cell
Range("A1").Select

Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\Location 4 Data.txt"
Open FName For Input Access Read As #5
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(5)
    Line Input #5, WholeLine
    If Right(WholeLine, 1) <> Sep Then
        WholeLine = WholeLine & Sep
    End If
    ColNdx = SaveColNdx
    Pos = 1
    NextPos = InStr(Pos, WholeLine, Sep)
    While NextPos >= 1
        TempVal = Mid(WholeLine, Pos, NextPos - Pos)
        Cells(RowNdx, ColNdx).Value = TempVal
        Pos = NextPos + 1
        ColNdx = ColNdx + 1
        NextPos = InStr(Pos, WholeLine, Sep)
    Wend
    RowNdx = RowNdx + 1
Wend
Close #5

'Format Time
    Columns("C:C").Select
    Selection.NumberFormat = "[$-409]m/d/yy h:mm AM/PM;@"
```

```
'Go to first cell
   Range("A1").Select

'********************************************************************
'Part 1f - Import "Location 5 Data.txt" to sheet "Location 5 - Data"
'********************************************************************

'Select sheet
Sheets("Location 5 - Data").Select
'Clear current data in sheet
Sheets("Location 5 - Data").Cells.Clear
'Begin copying file on first cell
Range("A1").Select

Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\Location 5 Data.txt"
Open FName For Input Access Read As #6
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(6)
   Line Input #6, WholeLine
   If Right(WholeLine, 1) <> Sep Then
      WholeLine = WholeLine & Sep
   End If
   ColNdx = SaveColNdx
   Pos = 1
   NextPos = InStr(Pos, WholeLine, Sep)
   While NextPos >= 1
      TempVal = Mid(WholeLine, Pos, NextPos - Pos)
      Cells(RowNdx, ColNdx).Value = TempVal
      Pos = NextPos + 1
      ColNdx = ColNdx + 1
      NextPos = InStr(Pos, WholeLine, Sep)
   Wend
   RowNdx = RowNdx + 1
Wend

Close #6

'Format Time
   Columns("C:C").Select
   Selection.NumberFormat = "[$-409]m/d/yy h:mm AM/PM;@"
'Go to first cell
   Range("A1").Select

'********************************************************************
'Part 1g - Import "Averages.txt" to sheet "Daily Averages"
'********************************************************************
```

```
'Select sheet
Sheets("Daily Averages").Select
'Clear current data in sheet
Sheets("Daily Averages").Cells.Clear
'Begin copying file on first cell
Range("A1").Select

Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\Averages.txt"
Open FName For Input Access Read As #7
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(7)
   Line Input #7, WholeLine
   If Right(WholeLine, 1) <> Sep Then
      WholeLine = WholeLine & Sep
   End If
   ColNdx = SaveColNdx
   Pos = 1
   NextPos = InStr(Pos, WholeLine, Sep)
   While NextPos >= 1
      TempVal = Mid(WholeLine, Pos, NextPos - Pos)
      Cells(RowNdx, ColNdx).Value = TempVal
      Pos = NextPos + 1
      ColNdx = ColNdx + 1
      NextPos = InStr(Pos, WholeLine, Sep)
   Wend
   RowNdx = RowNdx + 1
Wend

Close #7

'Format Time
   Columns("B:B").Select
   Selection.NumberFormat = "[$-409]m/d/yy h:mm AM/PM;@"
'Go to first cell
   Range("A1").Select

'End of PART 1
'*************************************************************************************
'*****************************************************************************
'PART 2 - Print Warning Sheet
'        This sheet will contain any reading when PM2.5 > 35 and PM10 > 150
'*****************************************************************************

'Select sheet
Sheets("Warning Report").Select
'Clear current data in sheet
Sheets("Warning Report").Cells.Clear
```

```
'Begin copying file on first cell
Range("A1").Select

'Copy "All Data,txt" and filter message for <>
Application.ScreenUpdating = False
'On Error GoTo EndMacro:

SaveColNdx = ActiveCell.Column
RowNdx = ActiveCell.Row

'File location in C++ Results
FName = "C:\Documents and Settings\" & UserName & "\Desktop\C++ Results\Warning Report.txt"
Open FName For Input Access Read As #8
Sep = ","

'Use while loop to copy text file into worksheet
While Not EOF(8)
   Line Input #8, WholeLine
   If Right(WholeLine, 1) <> Sep Then
      WholeLine = WholeLine & Sep
   End If
   ColNdx = SaveColNdx
   Pos = 1
   NextPos = InStr(Pos, WholeLine, Sep)
   While NextPos >= 1
      TempVal = Mid(WholeLine, Pos, NextPos - Pos)
      Cells(RowNdx, ColNdx).Value = TempVal
      Pos = NextPos + 1
      ColNdx = ColNdx + 1
      NextPos = InStr(Pos, WholeLine, Sep)
   Wend
   RowNdx = RowNdx + 1
Wend

'Go to first cell
Range("A1").Select

'End of PART 2
'*************************************************************************************
'************************************************************************
'PART 3 - Copy all sheets to new workbook
'************************************************************************

Dim sAppPath As String, sFileName As String, sDate As String

sAppPath = "C:\Documents and Settings\" & UserName & "\Desktop\Excel Output\"
sDate = Replace(FormatDateTime(Now(), vbShortDate), "/", "_")
sFileName = sAppPath & "Excel Output - " & sDate & ".xlsm"

ActiveWorkbook.SaveAs Filename:= _
sFileName, FileFormat:= _
xlOpenXMLWorkbookMacroEnabled, Password:="", WriteResPassword:="",
ReadOnlyRecommended:=False _
, CreateBackup:=False

   'Select sheet
```

```
    Sheets("User Info").Select
    'Go to first cell
    Range("A1").Select

'End of PART 3
'*************************************************************************************
EndMacro:
On Error GoTo 0
Application.ScreenUpdating = True
Close #8


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' END ImportTextFile Macro
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
End Sub
```

Appendix D – AMS Site Deployment

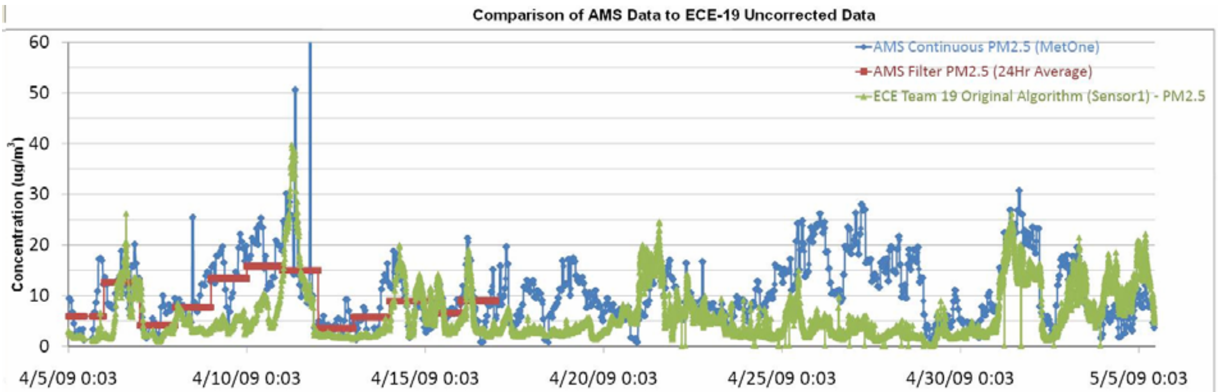


Figure 13: Comparison of AMS Data to ECE-19 Uncorrected Data

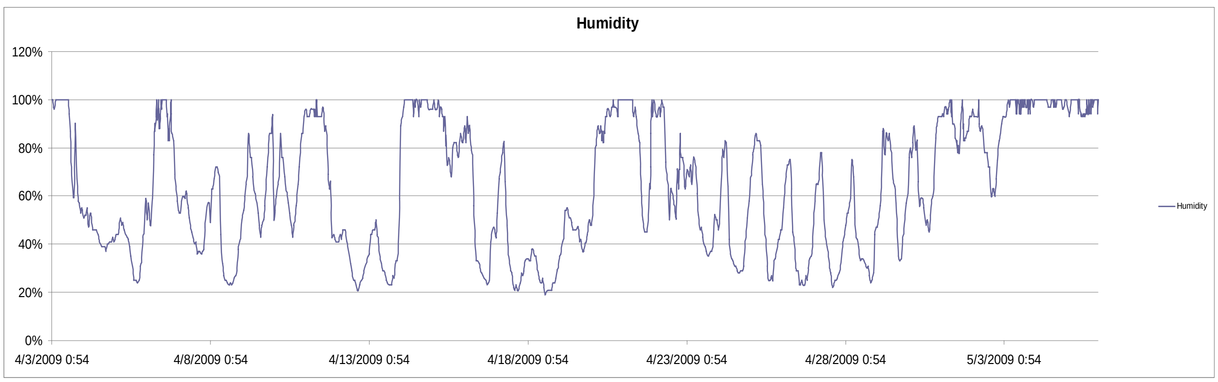


Figure 14: Humidity Data During Deployment

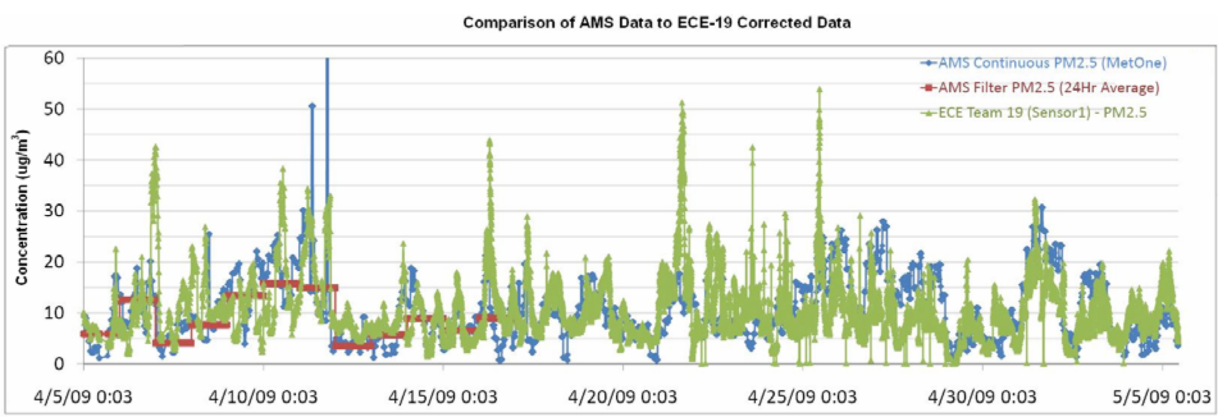


Figure 15: Comparison of AMS Data to ECE-19 Corrected Data

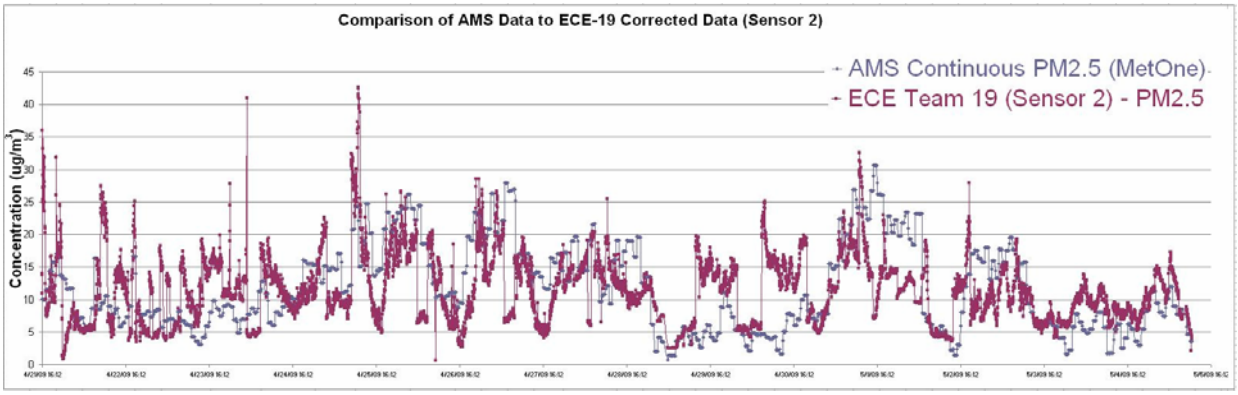


Figure 16: Comparison of AMS Data to ECE-19 Corrected Data (Sensor 2)

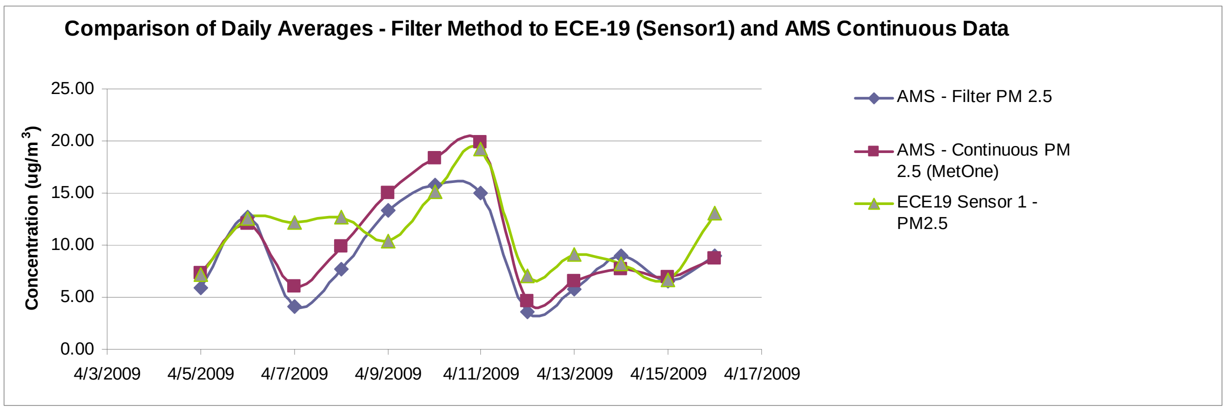


Figure 17: Comparison of Daily Averages – Filter Method to AMS and ECE-19 Continuous Data

						OVERALL
Date	**Filter**	**ECE 19 - Sensor '**	**AMS MetOne**	**ECE-Sensor 1 Dif**	**MetOne Diff.**	**Closest Method**
4/5/2009	5.88	7.13	7.35	1.25	1.47	ECE 19
4/6/2009	12.63	12.52	12.08	-0.11	-0.55	ECE 19
4/7/2009	4.08	12.15	6.05	8.07	1.97	MetOne
4/8/2009	7.67	12.74	9.82	5.07	2.15	MetOne
4/9/2009	13.38	10.38	15.02	-3.00	1.64	MetOne
4/10/2009	15.79	15.19	18.31	-0.60	2.52	ECE 19
4/11/2009	15.00	19.20	19.89	4.20	4.89	ECE 19
4/12/2009	3.54	7.06	4.63	3.52	1.09	MetOne
4/13/2009	5.75	9.05	6.57	3.30	0.82	MetOne
4/14/2009	8.92	8.21	7.70	-0.71	-1.22	ECE 19
4/15/2009	6.50	6.69	6.87	0.19	0.37	ECE 19
4/16/2009	8.96	13.13	8.74	4.17	-0.22	MetOne

Table 5: Comparison of Daily Averages – Filter Method to AMS and ECE-19 Continuous Data

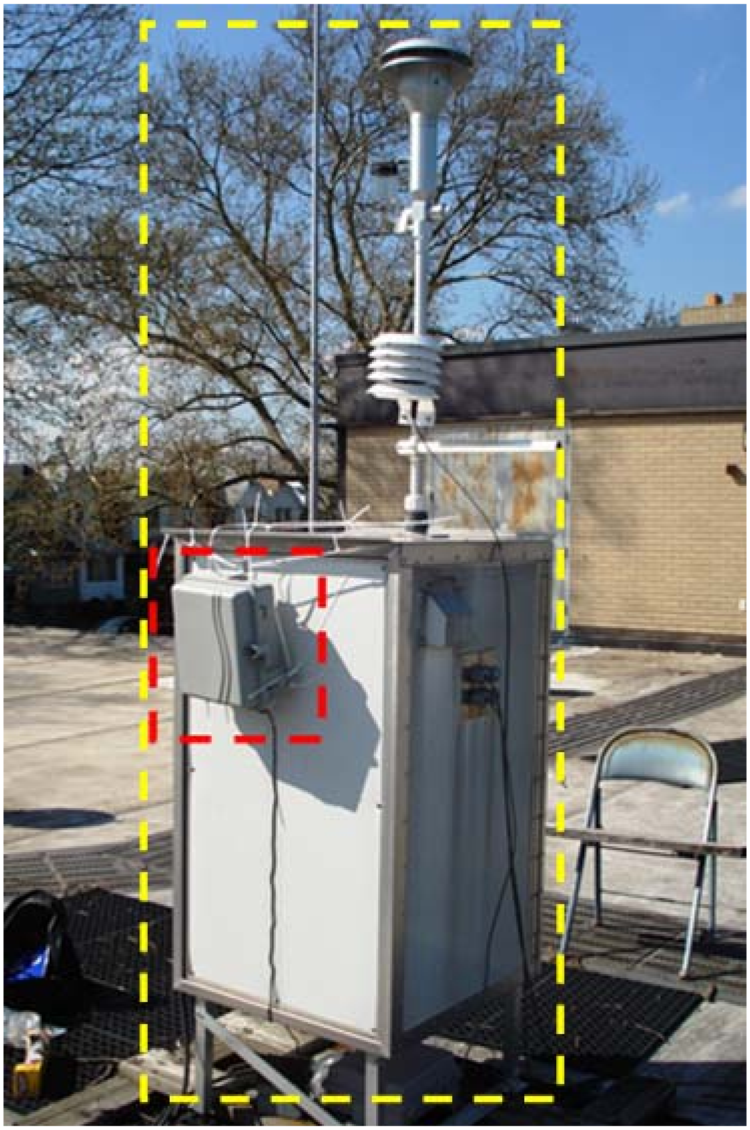


Figure 18: Size Comparison of System Node Size (Small Box) to FEM Monitor (Large Box)

Figure 19: AMS Filter Weighing System

Appendix E – Southeast Philadelphia Deployment

Figure 20: System Base Station Location

Figure 21: Southeast Philadelphia Deployed Node (Houston Community Center)

Appendix F – EPICS Mentoring

Figure 22: Senior Design Team Mentoring SLA Students

Figure 23: Senior Design Team Performs a Sensor Height Test with SLA Students

Appendix G – System Hardware Technical Discussion

Sensors

Sensor Decision Process

Several “off-the-shelf” components were evaluated for the actual capturing of ambient air. The options were considered based on several characteristics, including sensitivity (precision), price, and power consumption. As previously stated, the EPA utilizes a filtering system to test for particulate matter, thus this alternative was explored. In addition, lower cost alternatives were also discussed, including household air quality detectors similar to those used by design team ECE-17 in 2007-2008. However, because these sensors resulted in little success [11], a different “cost effective” solution was explored: the laser particle counter. Heralded for its mobility and ability to measure particle concentrations in short time intervals (in addition to its low cost) [12] this device was seen as a very attractive solution for the proposed design.

Upon researching the alternatives, it was determined that the Dylos DC1100 Pro laser particle counter would be most appropriate for the solution at hand. Though it is less precise than the Met-One Aerocet-212, this drawback was outweighed by the DC1100 Pro’s price per unit. Cost effectiveness played a large role in decision methodology, since the design is a “preliminary scan” of ambient air, [13].

Device Calibration

The original Dylos DC1100 Pro sensor was not calibrated to the necessary specifications. As stated on the company’s website, “The DC1100 Pro has increased lower sensitivity - detecting particles down to 0.5µm. The large particle size range is calibrated to 2.5µm and above [14].” However, for this specific system, the Dylos Corporation agreed to calibrate the laser particle counters with a greater degree of specificity, in which “one channel evaluates particle sizes between 0.5µm to 2.5µm and the other channel evaluates particle sizes between 2.5µm and 10µm [4].” By calibrating the device to these specifications, the data can be compared to the EPA standards for $PM_{2.5}$ and PM_{10}. The disadvantage to this process is that it is very time consuming for the technician to properly calibrate each device. Due to the complexity of this process, the 20% price discount (originally quoted in the proposed budget) could no longer be granted by the Dylos Corporation.

Serial Interface

In addition to the calibration of the device, the laser particle counter was customized to include a serial data connection. This simple interface allows the sensor to send out two particle counts, one for each channel, to the networking module. Once the data is sent to the networking component, it is then sent wirelessly to the coordinator at the base station. This process is completed by each of the sensors and the data is stored and then analyzed at a remote location.

Power Consumption

Due to the high current draw of the Dylos sensor, .250A at 9V, it was necessary to develop a voltage regulator circuit. This circuit would allow the sensor to be powered on while a reading was being taken, but powered off when a reading was not necessary. More specifically, a reading would be taken 6 times per hour for a total of 6 minutes per hour, and the device would be turned off for 54 minutes per hour. This would allow a significant amount of readings to be taken, but at the same time conserving battery power. For more information regarding this circuit, please see the "Power" section.

Networking

Network Decision Process

The decision to use a Xbee standard product was made due to its low power consumption when compared to Wi-Fi products (the second networking option evaluated for this system). Two Xbee options considered were the 2.4 GHz range or the 900 MHz range devices. The Digi XBee Pro DigiMesh 900 series utilized 900 MHz and was chosen mainly due to the low cost of the development kits. In addition to its low cost, the XBee Pro demonstrated an extended range, improved sleep modes, and mesh networking in comparison to the original XBee. [15]

Hardware

There are many advantages in utilizing the XBee Pro DigiMesh 900 development kit. The main advantage is that this kit contains all of the necessary hardware needed to begin deploying a wireless network, including the XBee Pro modules, interface boards, antennas, and cables. To interface a wireless module to the Dylos DC1100 sensor, an RS-232 interface board is used in combination with the XBee Pro Module. The mounting of the module is shown in Figure 8 (Appendix H) [16]. The serial interfaces of both the node module and the Dylos DC1100 Pro sensor, allow the transmission of data between the two devices through a standard DB-9 cable. Once the data is transmitted over the DB-9 cable, the module is able to transmit the information wirelessly to a coordinator module located at the base station. The coordinator module is similar to the node modules but contains a USB interface board to connect directly to an Asus Eee PC, which then collects and stores all of the data from each sensor.

Software

Another advantage of utilizing the XBee Pro DigiMesh 900 development kit is the use of free configuration and testing software. The software included with the kit is called X-CTU. By utilizing this software, each module can be programmed with several different configuration attributes [16]. The main configuration screen for each modem (or module) is shown in Figure 9 (Appendix H). The two main configurations used in this design are the cyclic sleep mode and the mesh capability.

The sleep mode is very important because it sets the module to sleep and wake for a programmed amount of time, allowing overall power consumption to be very low. In this design, the module sleeps for 9 minutes and wake for slightly longer than a minute. The wake time is set slightly longer than a minute to prevent error because the Dylos takes 60 seconds to transmit the data over a DB-9 cable to the module. The sleep mode must also be enabled to allow the voltage regulator circuit to work correctly at each node.

Power

Power Decision Process

In order to determine the power consumption of each system component, tests were run at 9V using a DC power supply. From this power supply, a supply current could be obtained for the air sensor and the XBee device. The XBee board was found to run at 82 mA in the on-mode, and 16 mA in sleep mode. The Dylos air sensor was found to run at approximately 250 mA in its on-mode, 128 mA in monitor mode, and 76 mA when the sensor was turned off but was still plugged in.

Battery Power

The power supply located at each node in the system is an Impact IMBPD8000 BPD-8000 Rechargeable Battery. This device was ideal, as it supplies power at a voltage of 9V, a requirement of both the Dylos air sensor and the XBee chip. The battery has an amp-rating of 8 Amp-hour, resulting in 72 Watt-hour for each node in the system. After extensive searching and comparison to other batteries on the market, this battery was selected due to its lower cost per Watt-hour.

Voltage Regulator Circuit

During both monitor and power off modes, the air sensor pulled far too much current. Thus, a voltage regulator circuit was designed to act as a switch for the air sensor in order to minimize power losses in the system. The voltage regulator acts as a switch turning on and off the current supplied to the air sensor. This is done by taking advantage of the XBee's sleep mode. The losses incurred from the voltage regulator are minimal to the system without it. The voltage regulator circuit uses only about 8 mA in the conducting state and only about 5 μA in the non-conducting state. A schematic of the power for each node can be seen in Figure 10 (Appendix H).

Node Placement

Enclosures

The nodes must be placed into a weatherproof enclosure to protect the hardware from damage. There are many enclosures available, but few that meet the specifications needed for this design. The large volume occupied by the Dylos air sensor (when placed face down in the enclosure) resulted in the need for a very large, weatherproof enclosure, measuring approximately 7 inches deep, 8-9 inches wide and 10-12 inches tall.

As a low cost solution, the use of Primex P136 enclosures, leftover from a previous system [11] was considered. Though this could have lowered the operating budget, the enclosures needed to be significantly altered to meet the specifications of the current system. These alterations would have left the system quite vulnerable to weather hazards; thus despite the possibility of cost savings, an alternative solution was sought.

The option utilized in the final design was the Primex P1000 enclosure. This product has a larger depth (4.27 inches), which would allow the sensor to be entirely encased within the enclosure. The enclosure also has sealable valves at its bottom, which are opened to allow for airflow from the ambient environment, into the box. In addition to the larger depth, this product also contains weatherproofing around the closing edges and is relatively inexpensive, costing approximately $38 [18]. A plastic cylinder was placed over the mouth of the Dylos sensor, to prevent the recirculation of air within the enclosure.

In addition to the node enclosures, a separate enclosure is needed for the base station. Since the base station does not include a Dylos sensor, the depth is not a problem. The chosen enclosure for the base stations was L-Com's weather proof model NBP141004-100. This model has dimensions of 14 x 10 x 4 and costs approximately $45. As an added benefit, a removable 120 VAC power module is included. This allows easy access to power outlets for the Asus Eee PC [19].

Appendix H – Additional Figures

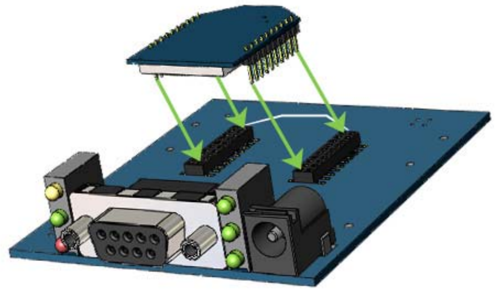


Figure 24: XBee Pro 900 Module Mounting on RS-232 Interface Board

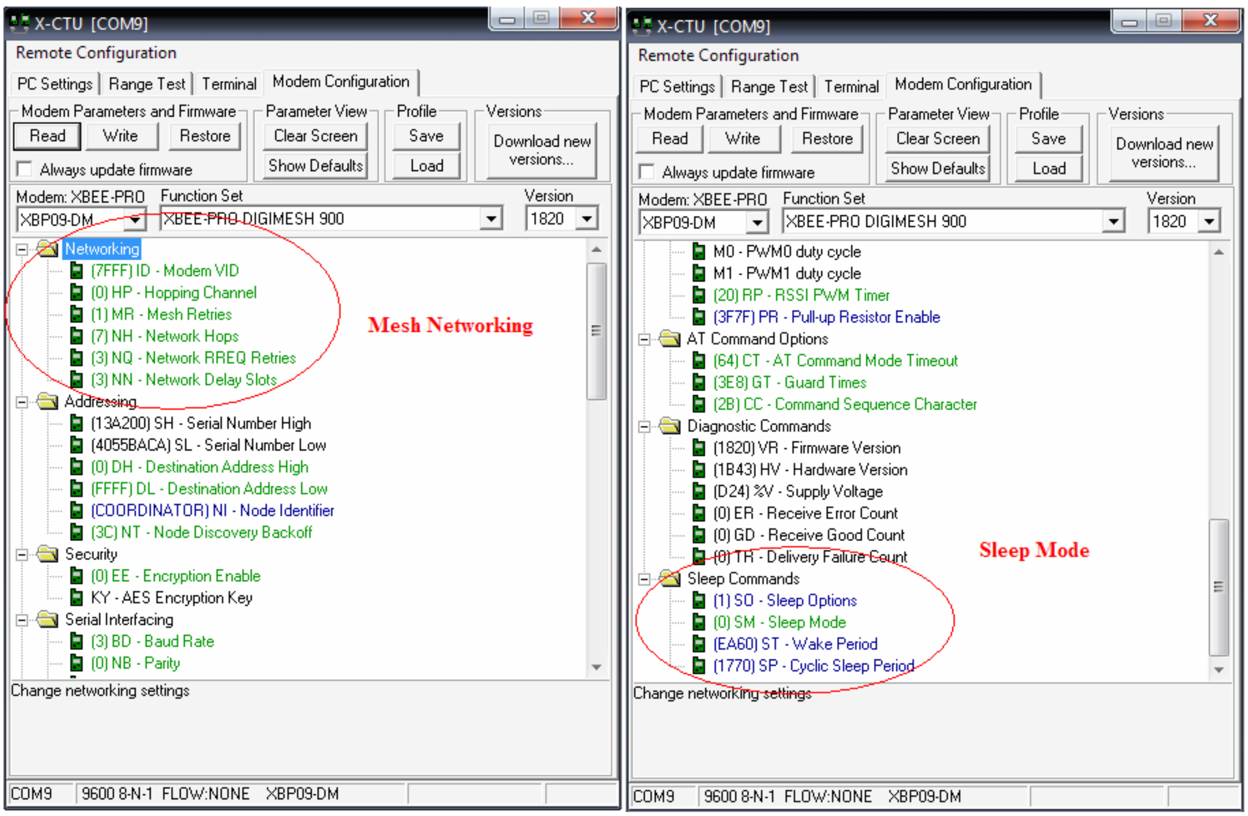


Figure 25: Modem Configuration Screen for X-CTU Software

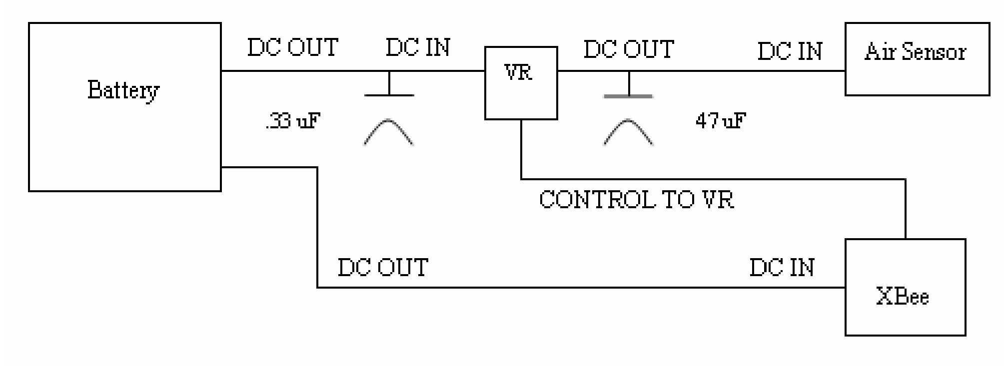


Figure 26: Air Sensor Node Power Circuit Schematic

Engineering Manual:

Hardware Development

Purpose of Manual

The engineering manual is meant to serve as a guide in the replication of a network of air quality sensors. The manual is broken up into several sections, each describing the details of a specific piece of hardware. These sections are listed below:

1. Introduction to the System Node
2. Dylos DC1100 Pro Laser Particle Counter
3. XBee Pro DigiMesh 900 Module and Interface Boards
4. Null Modem Serial Cable (RS-232 Interface)
5. Voltage Regulator Circuit
6. Power Source - 9V Battery
7. Completion of System Node

1. Introduction to the System Node

The completed system node is depicted in Figure 1 below and consists of a Dylos DC1100 Pro laser particle counter, an XBee Pro DigiMesh 900 wireless module, a voltage regulator circuit, and a 9V battery supply. The Dylos DC1100 sensor contains two channels for capturing particle counts. The Dylos also contains an RS-232 interface, allowing the data from each channel to be transmitted to the XBee Pro module. The XBee Pro module can wirelessly transmit the same data to a coordinator module to store the data on a PC. The second function of the XBee Pro module is to provide the logic for the voltage regulator circuit. This is accomplished by utilizing the programmable sleeping capabilities. The power supplied to the voltage regulator circuit is a 9V battery, which supplies power to each component in the system. Battery power has been utilized in order to minimize the need for AC outlets. However, if AC power is available, the battery supply can be replaced with an AC power adapter.

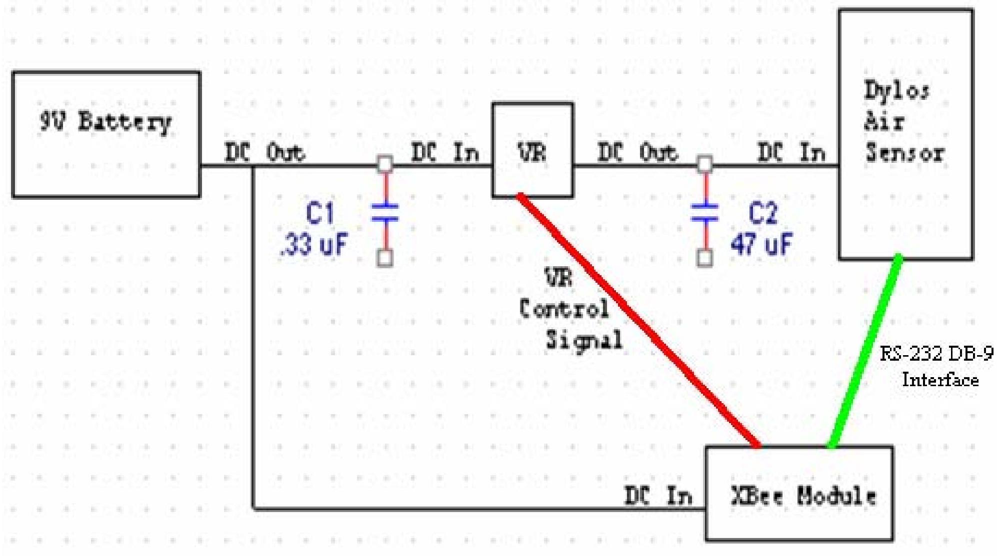


Figure 1: Interface Block Diagram of Sensor Node

2. Dylos DC1100 Pro Laser Particle Counter

The Dylos DC1100 Pro is an air quality sensor, shown in Figure 2, contains two channels for capturing particle counts. This device was mainly chosen due to its low cost, but also for its dual channel design. These channels have been calibrated by the Dylos Corporation to match the requirements of the system. More specifically, the calibration is important because it allows both PM2.5 and PM10 to be detected from a single sensor.

Device Details

- Manufacturer: Dylos Corporation (dylosproducts.com)
- Interface: RS-232 (DB-9 connector)
- Required Power: 9V DC
- Channel Calibration :
  - Channel 1: 0.5 to 2.5µm
  - Channel: 2 – 2.5 to 10 µm
- Cost: $300

Figure 2: Dylos Air Sensor

3. XBee Pro DigiMesh 900 Module and Interface Board

The XBee Pro DigiMesh 900 is a newer series of XBee modules. The Pro series features improvements in both range and power management. Due to these characteristics, as well as low cost, the XBee Pro DigiMesh 900 module was chosen as the wireless component in the system. In addition to the wireless module chip (which contains an antenna), two types of interface boards are needed in this system. The base station, which contains the PC for data collection, requires a USB interface board. The nodes, each containing a Dylos sensor, require a serial interface board. All of these components can be obtained by purchasing an XBee Pro DigiMesh 900 development kit from Digi. The circuit boards can be easily constructed by placing the XBee Pro module onto the receptacle of the interface board, as shown in Figure 3. Alternative XBee kits may be purchased, if they contain similar interface boards.

Device Details

- Supplier: Digi (digi.com)
- Interface Boards:
    - RS-232 (DB-9 connector)
    - USB
- Required Power: 9V DC (Serial Interface Board)

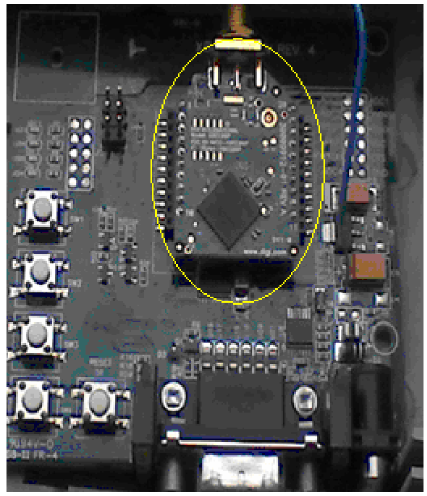


Figure 3: XBee serial interface board with module (circled)

4. Null Modem Serial Cable (RS-232 Interface)

In order to transmit the Dylos sensor output to the XBee Pro Module a serial communications link must be established. This can be done by using the supplied null modem cable in the XBee Pro development kit. Alternatively, a null modem cable can be easily constructed with two DB-9 male connectors by following the pin out in Figure 4.

a. Cable Components

   1. DB-9 Male Connectors
      - Manufacturer: Tyco Electronics
      - Part Number: 5747250-4
   2. Wiring

b. Null Modem Cable Construction

   a. Connect pin 5 (ground) from one header to pin 5 of the other header via wire wrapping or soldering
   b. Connect pin 2 (in) of header 1 to Pin 3 (out) of header 2 via wire wrapping or soldering
   c. Connect pin 3 (out) of header 1 to pin 2 (in) of header 2 via wire wrapping or soldering

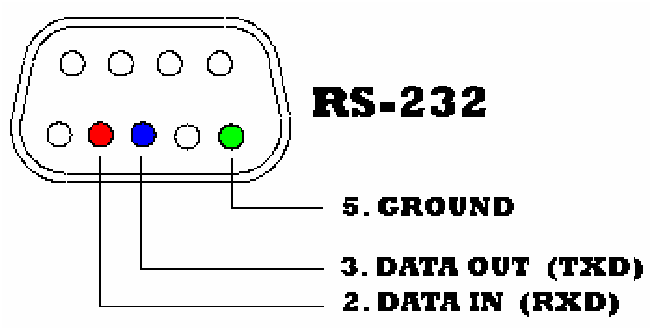


Figure 4: RS-232 (DB-9) Pin Out

5. Voltage Regulator Circuit

The voltage regulator circuit controls the power supplied to the Dylos air quality sensor. It allows power to be supplied to the sensor only when a reading is taken. For all battery powered nodes, this circuit must be utilized due to the high power consumption of the sensor.

a. Circuit Components

a. Voltage Regulator Chip: (See Figure 5)
   i. Manufacturer: Sharp Microelectronics
   ii. Part Number: PQ090RDA1SZH

b. 33 µF Capacitor
   i. Manufacturer: Panasonic - ECG
   ii. Part Number: ECA-1HHGR33

c. 47 µF Capacitor
   i. Manufacturer: United Chemi-Con
   ii. Part Number: EKY-250ELL470ME11D

d. 9-V power jack:
   i. Manufacturer: CUI Inc
   ii. Part Number: PJ-018H

e. 9-V adapter with cable assembly:
   i. Manufacturer: Tensility International Corp
   ii. Part Number: CA-2188

f. PC Board:
   i. Manufacturer: Vector Electronics
   ii. Part Number: 8015-1

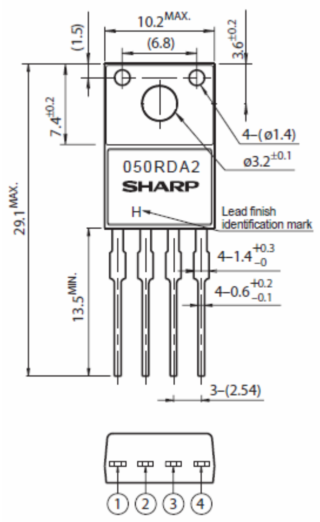


Figure 5: Voltage Regulator Schematic

b. Construction of Voltage Regulator Circuit

1. Prepare PC board by modifying the size to approximately 1in x 1 in.
2. Review circuit schematic in Figure 6 and before soldering components to the PC board
3. Review Figure 7 to see an example of proper placement of components on the PC board
4. Solder components (voltage regulator chip and capacitors) to PC Board
5. A connection to XBee Pro serial interface board must be made to control power supplied to the Dylos sensor.
   i. First, make a connection to pin 4 on the voltage regulator chip
   ii. Second, make a connection to the XBee Pro serial interface board to the right side of the 4^{th} resistor. (See Figure 7 for an example of proper placement)
6. Attach a 9V power adapters to the board by using the attached twisted pair cable assembly
   i. Dylos sensor 9V power connection – positive end of twisted pair should be connected to pin 2 of the voltage regulator chip
   ii. XBee Pro 9V power connection – positive end of the twisted pair should be connected to pin 1 of the voltage regulator chip
   iii. Negative ends of both twisted pairs must be grounded to a common ground on the PC Board
7. Mount the completed circuit to the XBee Pro module by using a screw.

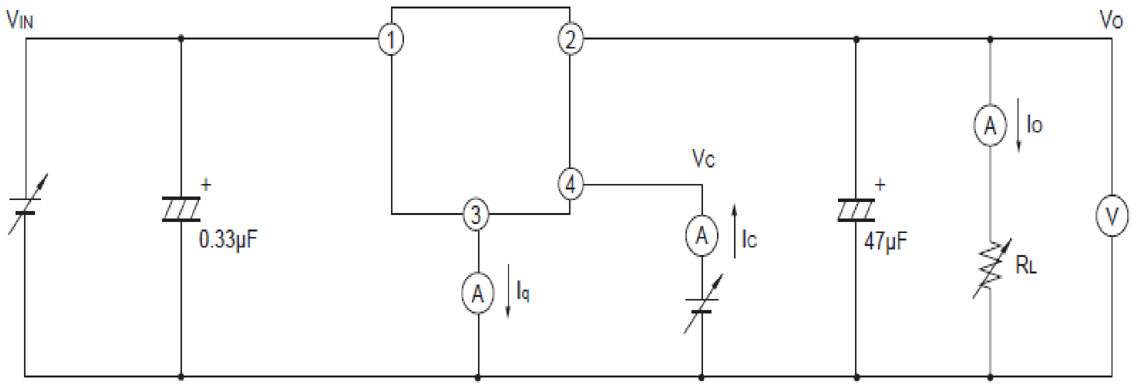


Figure 6: Complete Voltage Regulator Circuit Schematic

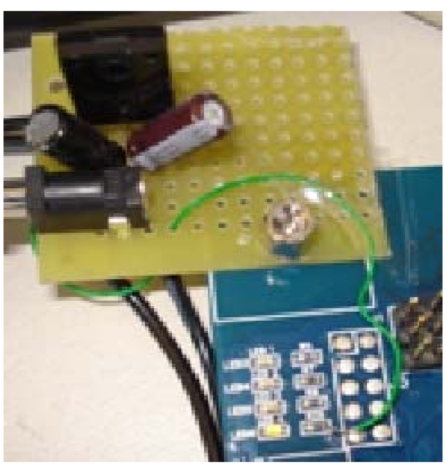


Figure 7: Voltage Regulator Circuit Attachment to XBee Interface Board

6. Power Source - 9V Battery

The power source chosen for this system is the Impact universal Li-Ion rechargeable battery, as shown in Figure 8. This battery was chosen due to its high capacity-to-cost ratio. An alternative battery may be selected if it supplies an output of 9V.

Device Details

- Supplier: B&H (bhphotovideo.com)
- Part Number: BPD-8000
- Capacity: 8000mAh
- Output Voltage: 9-10V
- Connections
    i. "Out" – connection between the battery and the voltage regulator circuit with supplied cable
    ii. "In" – used to charge the battery with supplied AC adapter

Figure 8: Impact 9V Battery

7. Completion of System Node

Once all of the parts are assembled, they must be placed securely within a weatherproof enclosure. The Primex P1000 enclosure has been used in this system due to its large depth and weatherproof hinges. A large depth is needed to completely enclose the Dylos sensor's irregular shape. The completed system is shown in Figure 9 below.

a. Overview of System Node Components

1. Dylos DC1100 Pro Laser Particle Counter
2. XBee Pro DigiMesh 900 Module and Interface Boards
3. Null Modem Serial Cable (RS-232 Interface)
4. Voltage Regulator Circuit
5. Power Source - 9V Battery

b. Mounting Components in Enclosure (See Figure 9)

1. Secure the voltage regulator circuit to the XBee Pro serial interface board by using a small screw or bolt. This is shown in Figure 7.
2. Fasten the XBee Pro interface board down at the top of the enclosure with the use of the Velcro.
   i. To do this, first attach the Velcro to the back side of the interface board and then attach to the enclosure
3. Place Velcro on the face of the Dylos sensor and then attach it to the bottom of the enclosure
4. Create a loop for the battery on the left side of the enclosure
   i. Take one piece of Velcro with two sides of equal length
   ii. Cut the middle of one of the two sides
   iii. Stick the two parts together making sure that the sticky exposed side of the uncut side is in the middle
   iv. Attach the sticky portion to the enclosure as shown in figure 9
   v. Place the battery inside the loop and close the loop by Velcroing the sides together
5. In order to ensure no air from the exhaust of the Dylos enters the intake a plastic tube needs to be Velcroed to the Dylos intake

i. Take a plastic tube and cut out a portion the size of the Dylos intake

ii. Place Velcro on the sides that were cut

iii. Attach to the outside of the Dylos intake

6. The node should now be complete and can be closed shut and ready for deployment.

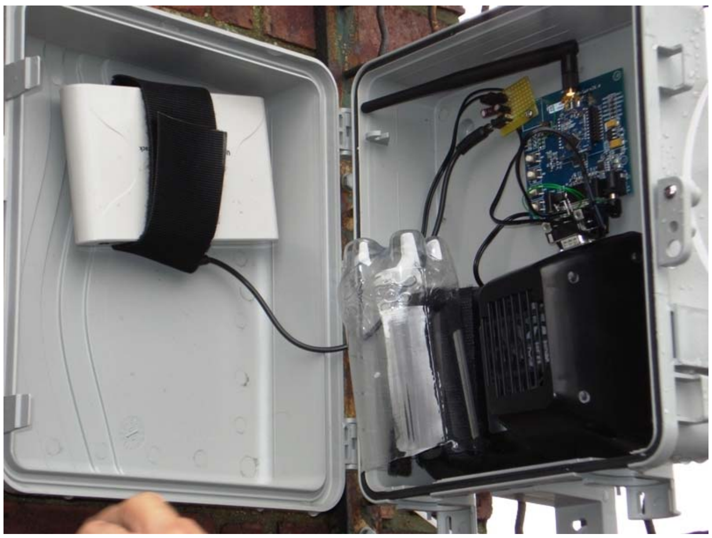


Figure 9: Complete System Node

User Manual

Part 1:

"Network Configuration Software Package"

Part 1 Contents:

A. Introduction to Network Configuration Software

This manual assumes that the system hardware is complete, as shown in Figure 1. If this is not the case, please refer to the "Engineering Manual: Hardware Development" for assistance. This software manual provides the details involved in properly installing the XBee Pro development kit supplied software and drivers. After installation is completed, this manual further describes the details in configuring a successful network.

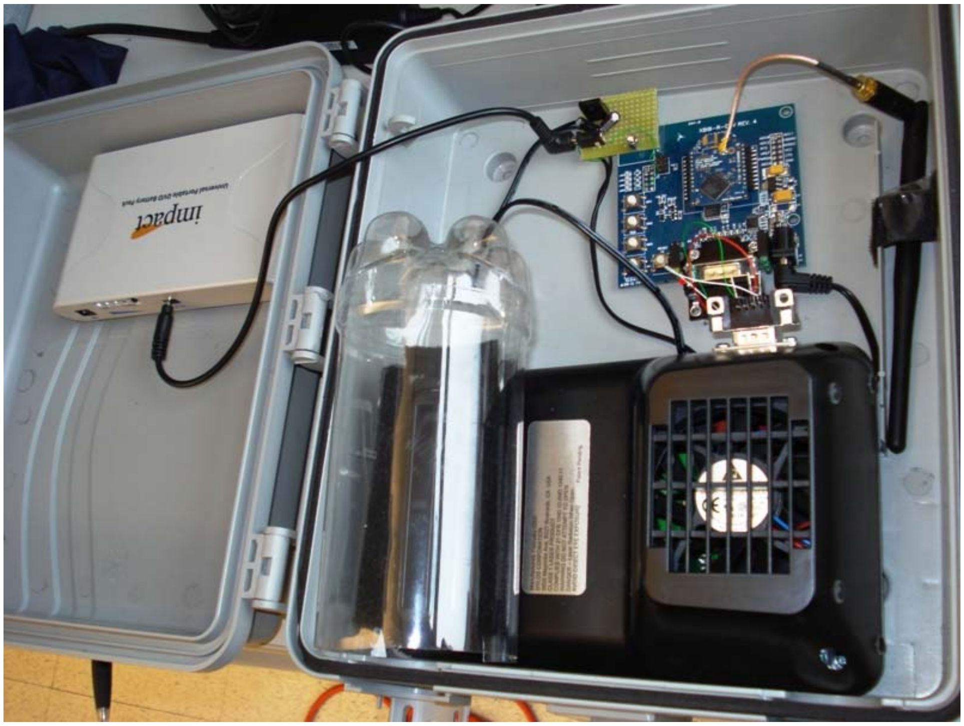


Figure 1: Node System Setup

B. Installation of X-CTU Software:

X-CTU is the network configuration software that is supplied with the XBee Pro DigiMesh 900 development kit. The software is contained on a CD labeled “Hardware and Software Setup”. This software must be installed to perform network configuration.

Installation Details

1) Insert the supplied CD and wait for the set-up screen to appear
2) Click on “Modules, Sensors, and Adaptors Documentation/Software”
3) Then click “XBee Modules”
4) Then click on “DigiMesh 900”
5) Then click on install X-CTU Software as shown in Figure 2
6) Follow the onscreen instructions to install the X-CTU Software

Figure 2: Installation of X-CTU Software

C. Installation of USB Drivers:

In order to utilize the supplied USB interface board, drivers must first be installed by using the supplied "Hardware and Software Setup" CD. The two drivers that need to be installed on the PC are a USB driver and a virtual COM port driver. (The COM port driver allows the USB port to perform like a physical COM port.)

<u>Installation Details</u>

1) First insert the CD is in the drive before connecting the USB interface board.
2) Connect the USB interface board and wait for the "Found New Hardware Wizard." (Plug and play device)
3) Click "Continue Anyway" when the alert box shows up
4) This will install the USB Driver
5) Follow the steps to install the virtual COM port driver which will show up after the first driver is installed
6) Click " Continue Anyway" when the alert box shows up
7) The necessary drivers will now be installed on the system

D. Configuration of Coordinator Module

After completing the software driver installations, the X-CTU software can be used to perform network configuration. This process begins with the configuration of the "Coordinator" module through a USB interface board. The coordinator is considered the data collector of the network and is directly connected to the Base Station (or PC) to allow proper data storage.

Installation Details

1) Double-Click the X-CTU software (there should now be an icon on the desktop)
2) Under the "PC Settings" Tab select the PC serial COM port that will be used. The serial ports used will be labeled as Digi PKG-U Serial Port Adapters
3) The settings should be as follows:
    - Baud Rate: 9600
    - Flow Control: None
    - Data Bits: 8
    - Parity: None
    - Stop Bits: 1
4) Once these are selected, make sure the "Enable API" box is checked and click the "Test/Query" button to determine if the correct COM Port was selected, as shown in Figure 3.

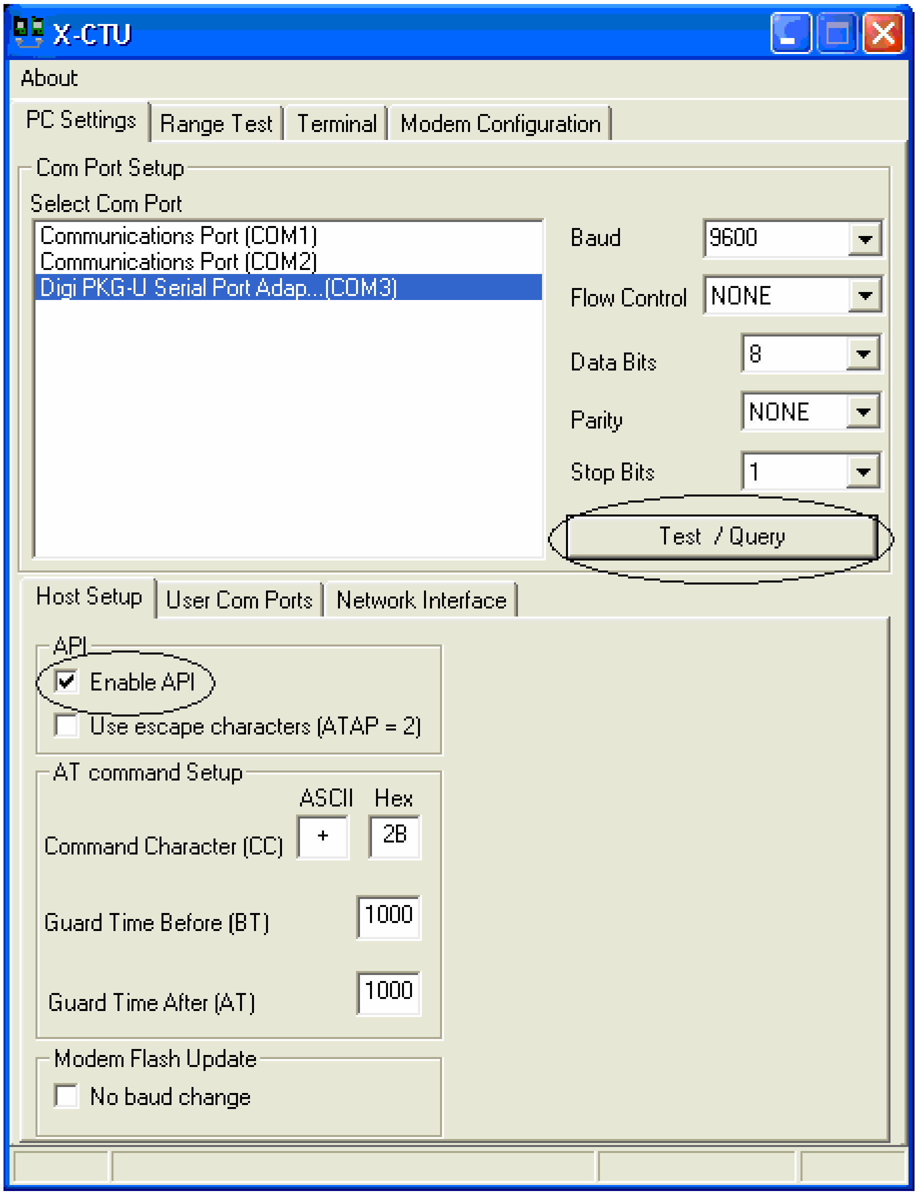


Figure 3: X-CTU Com Port Setup

Read/Set Modem Parameters

1) Go to the Modem Configuration Tab
2) Verify the newest versions are installed by clicking "Download New Versions"
3) Click "Read" to display the parameter, as shown in Figure 4
4) Module Identification
   - Under "Addressing" select NI-Node Identifier
   - Enter "COORDINATOR", as shown in Figure 5

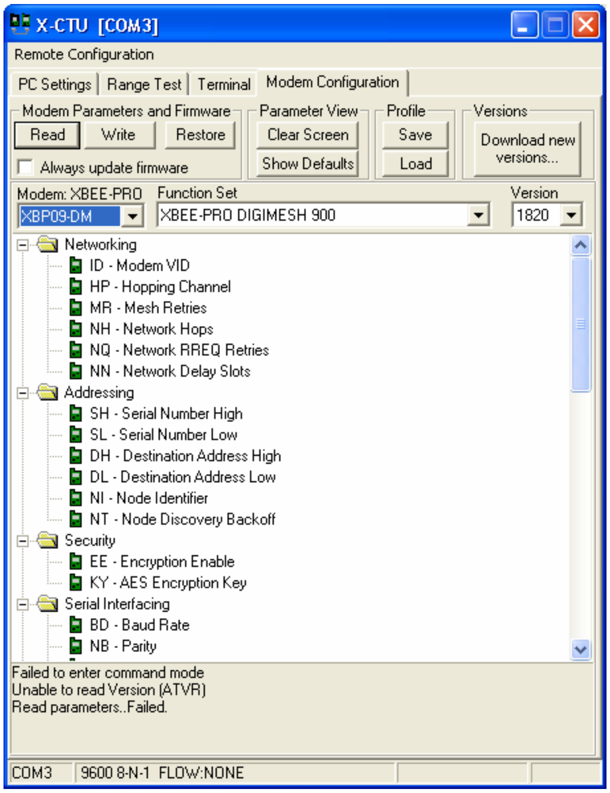


Figure 4: XBee Pro Module Parameters

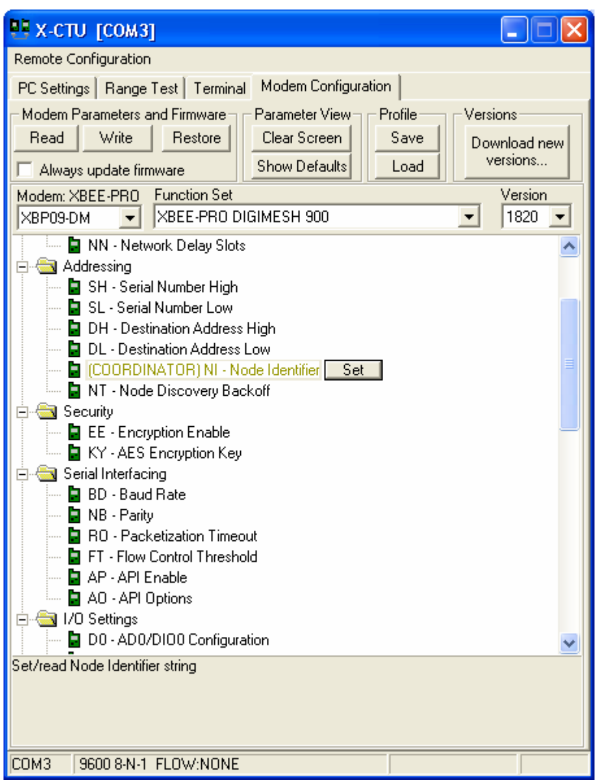


Figure 5: Enter "Coordinator" Identifier

<u>Sleep Configuration</u>

1) Click on the "Modem Configuration" Tab
2) Click on "Sleep Options"
3) To set this node to be the coordinator, select 1-PREFERRED SLEEP COORDINATOR – ENABLE (As shown in Figure 6)
4) The next thing that needs to be determined is whether the network will be in normal (continuous mode) or cyclic sleep mode
5) This option is found under "SM-Sleep Mode"
   - To set the network to normal select 0
   - To set the network to Cyclic Sleep select 4
   - The COORDINATOR will control the sleep cycles of the other modules in the network
   - To run the system as specified select option 4 (As shown in Figure 7)
6) Set sleep period (SP-Cyclic Sleep Period)
   - Value is in increments of 10 ms
   - Enter value in HEX format
   - For 8 min and 55 sec, enter "D0FC"
7) Set Wake Period (ST-Wake Period)
   - Value is in increments of 1 ms
   - Enter value in HEX format
   - For 1 min and 5 sec, enter "FDE8"
8) The parameters should now look like Figure 8
9) Click "Write" to save these parameters to the XBee Pro Module

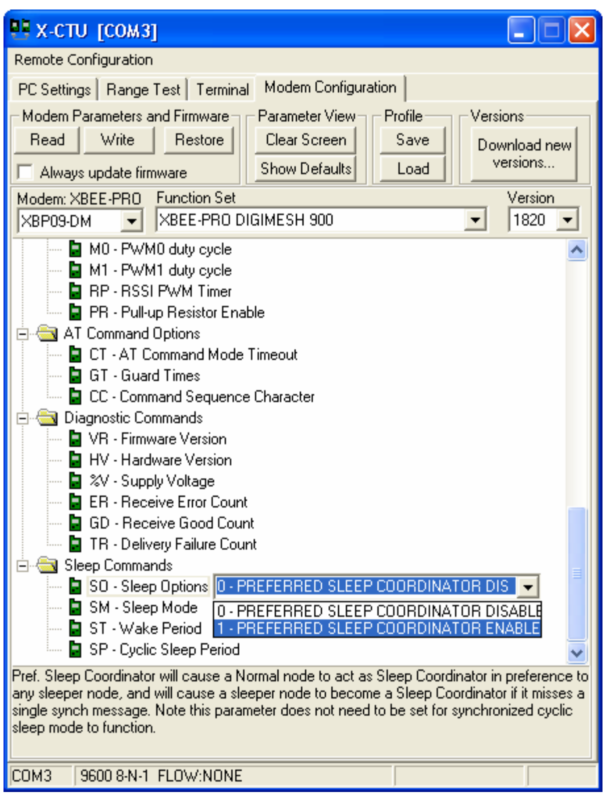


Figure 6: Enable Preferred Sleep Coordinator

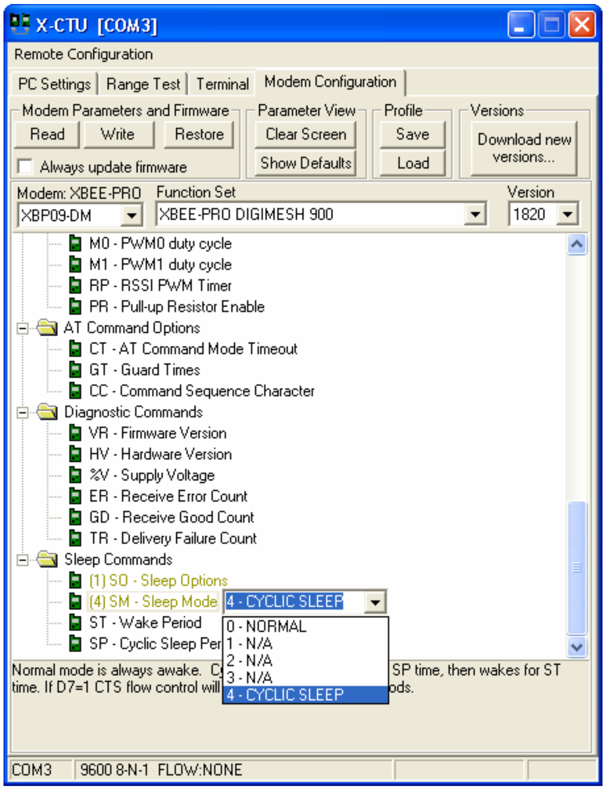


Figure 7: Enable Cyclic Sleep

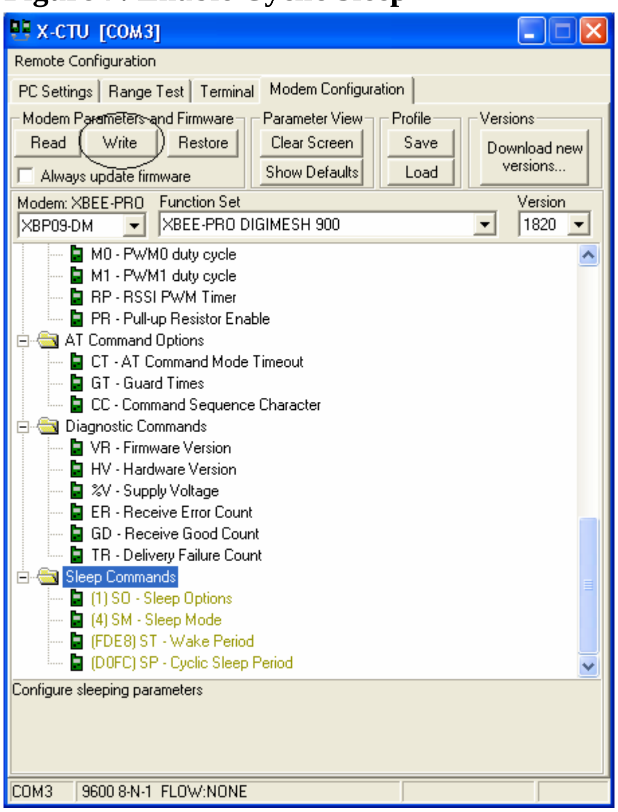


Figure 8: Sleep and Wake Parameters

E. Configuration of System Node

Completion of Cyclic Sleep Configuration

After the XBee Pro Coordinator module (USB interface) has been configured with the proper sleep parameters, each XBee Pro node module (serial interface) will need to be programmed individually by utilizing a serial to USB cable.

1) Connect the XBee Pro serial interface board via a serial-to USB cable
2) Re- open the X-CTU Software (first close all existing instances)
3) Go to the "Modem Configuration" tab and press "Read"
4) Scroll down to "Sleep Commands"
5) Set "SO-Sleep Options" to " 0 –PREFERRED SLEEP COORDINATOR-DISABLED"
6) Set "SM-SLEEP MODE" to "4-Cyclic Sleep" as shown in Figure 9
7) Click "Write"
8) The module will now be set to receive the sleep cycle from the coordinator
9) Repeat for all nodes in the network

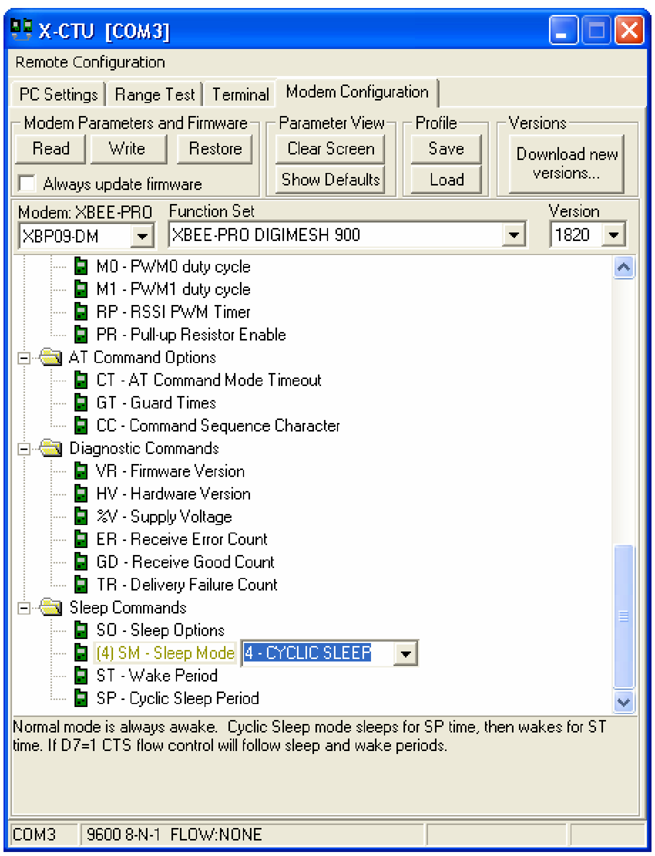


Figure 9: Node Sleep Mode Configuration

F. Test Network Range

A network range test should be done prior to deploying any hardware in the field. This will help in determining your network capabilities and reduce improper placement of equipment. In order to utilize this useful tool, a NULL is required for serial port of the interface board. This NULL is provided in the XBee Pro development kit. This is shown in Figure 10.

Perform Test

1) Connect coordinator to USB port
2) Open the X-CTU and select the associated COM-Port
3) If the "Enable API" button is checked, uncheck it
4) Verify the settings are as follows:
    - Baud Rate: 9600
    - Flow Control: None
    - Data Bits: 8
    - Parity: None
    - Stop Bits: 1
5) Make sure the modem configurations are set to the default parameters
6) If this is not done already, click the "Restore" button on the modem configuration page
7) Connect a NULL to the serial port of the serial interface board, as shown in Figure 10
8) Select the "Range Test" Tab
9) Check the "RSSI" Checkbox to enable Received Signal Strength Indicator (As shown in Figure 11).
10) Click "Start" to begin the test
11) Move the module away from the base station to find the maximum range
12) Click "Stop" to end the range test
13) The signal strength indicator is shown in Figure 12
14) Alternative to software - view the LEDs on the serial interface board
    - One glowing green if receiving data from the coordinator

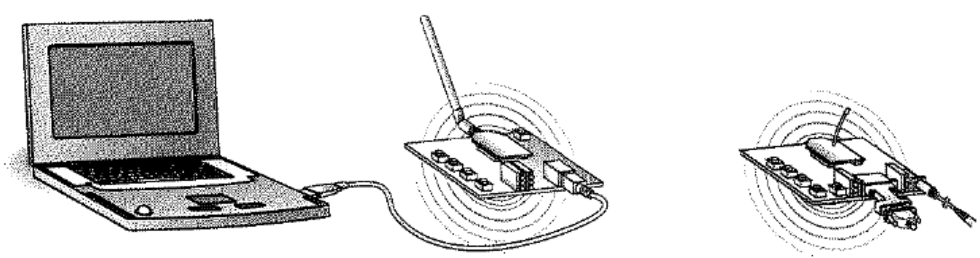


Figure 10: Range Test Setup

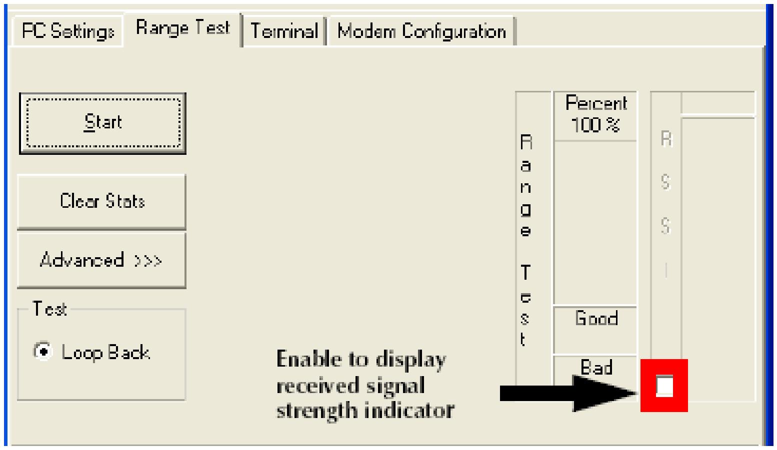


Figure 11: Check box for RSSI indicator

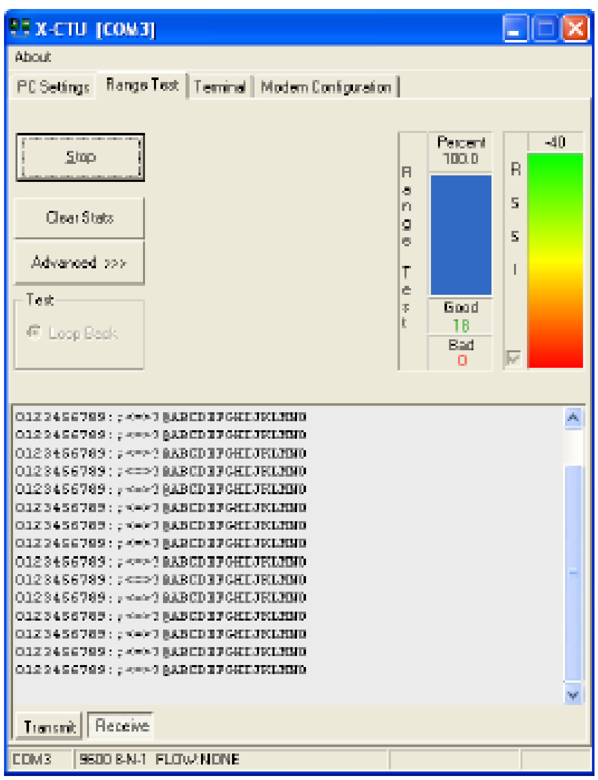


Figure 12: Range Test in Action

User Manual

Part 2:

"Algorithm Implementation Software Package"

Part 2 Contents:

Overview

The algorithm implementation software package is broken up into two phases. The first phase requires capturing and saving the data during system deployment. After the data has been collected, the data enters the second phase of the software process. In the second phase, the captured data is extracted, converted to the EPA scale ($\mu g/m^3$), and graphed for analysis. A block diagram of this process is shown in Figure 13.

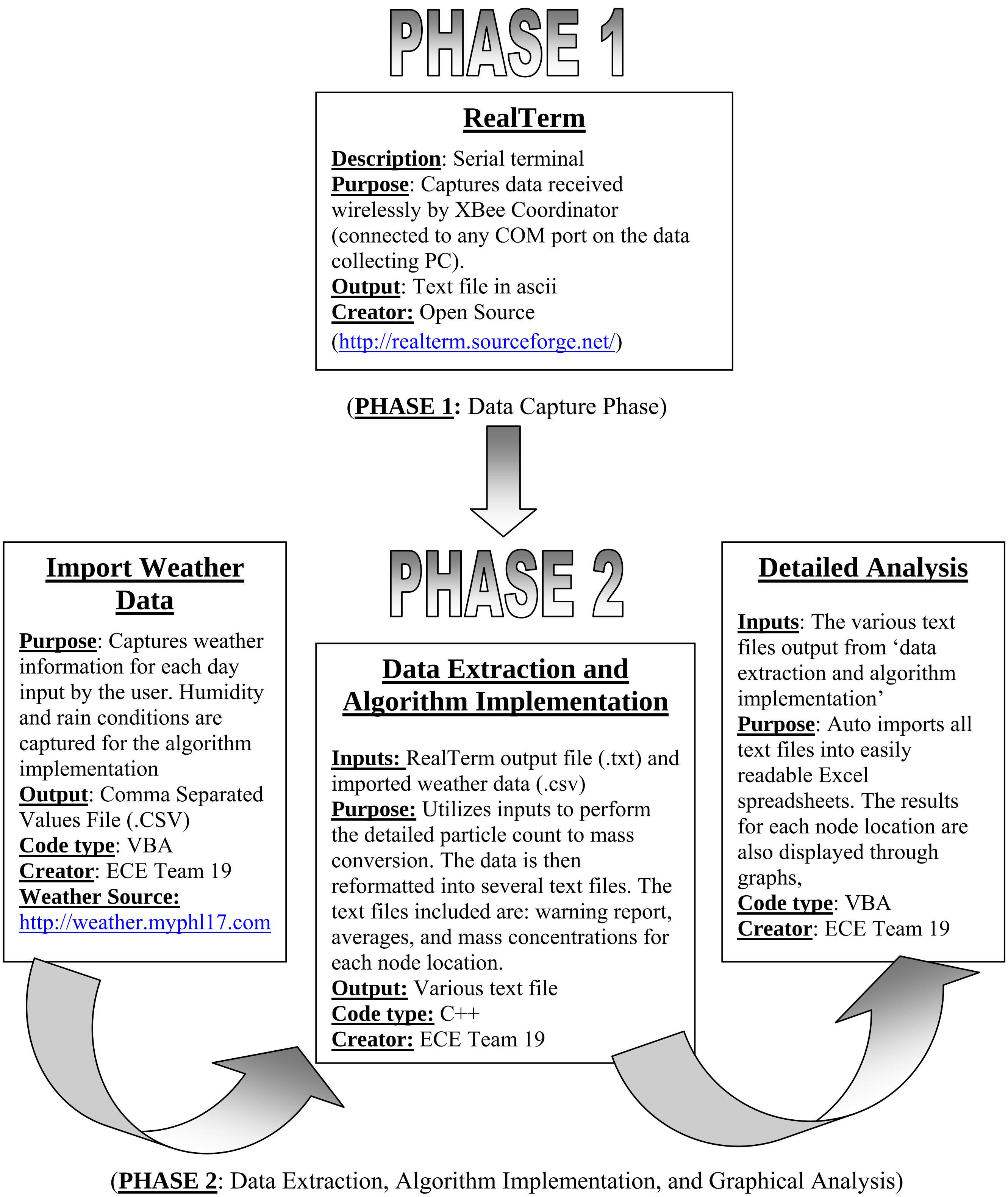


Figure 13: Software Block Diagram

A. <u>PHASE 1:</u> "Data Collection"

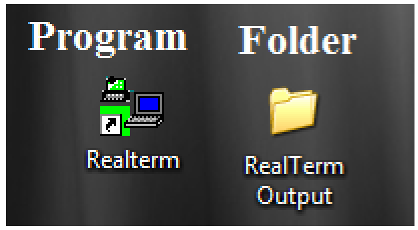


Figure 14: RealTerm Program and Output File (Located on the Desktop)

1. RealTerm

RealTerm is a serial terminal program that is used to capture the sensor data. This program allows the coordinator module to communicate through a COM port and output the received data to a text file.

Installation of Program

1) Download Realterm at http://realterm.sourceforge.net/index.html#downloads_Download
2) Run the setup file to install Realterm on the same computer where the coordinator is located

Configuration of Port

3) Open Realterm
4) On the "Port" Tab make sure the settings match those of the X-CTU software
   - Port: Same port that the USB drivers were installed on
   - Baud Rate: 9600
   - Parity: None
   - Bits: 8
   - Flow Control: None
   - Stop Bits: 1
5) Click "Change" to apply changes (As shown in Figure 15)

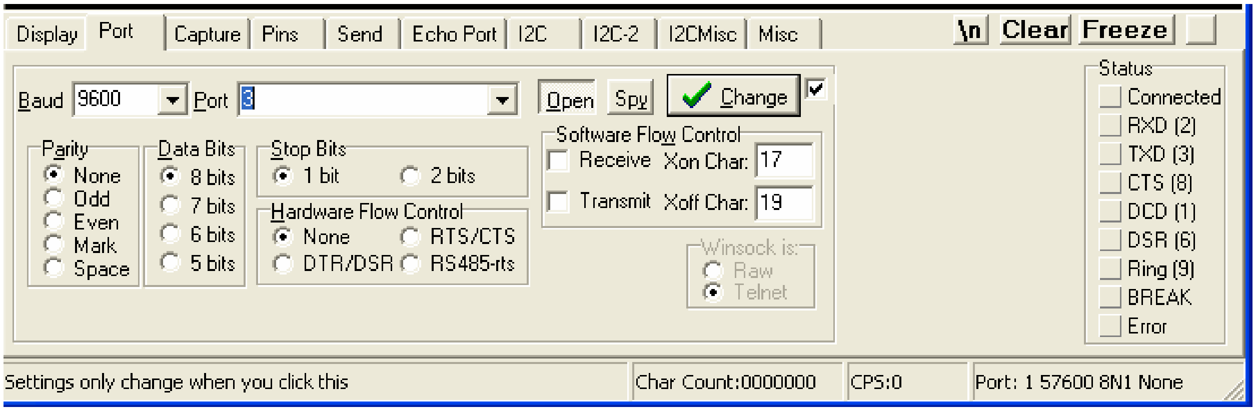


Figure 15: RealTerm - Port Tab

Configuration of Output File

1) Switch to the "Capture" Tab
2) In the file portion of the window type the destination of where the results text file is to be save
   - Current System: C:\Documents and Settings\Anthony\Desktop\RealTerm Output\results.txt
3) Make sure the "Unix timestamp" button is clicked so that a timestamp will appear in the final text file
4) To Capture the data click " Start Overwrite", (As shown in Figure 16)

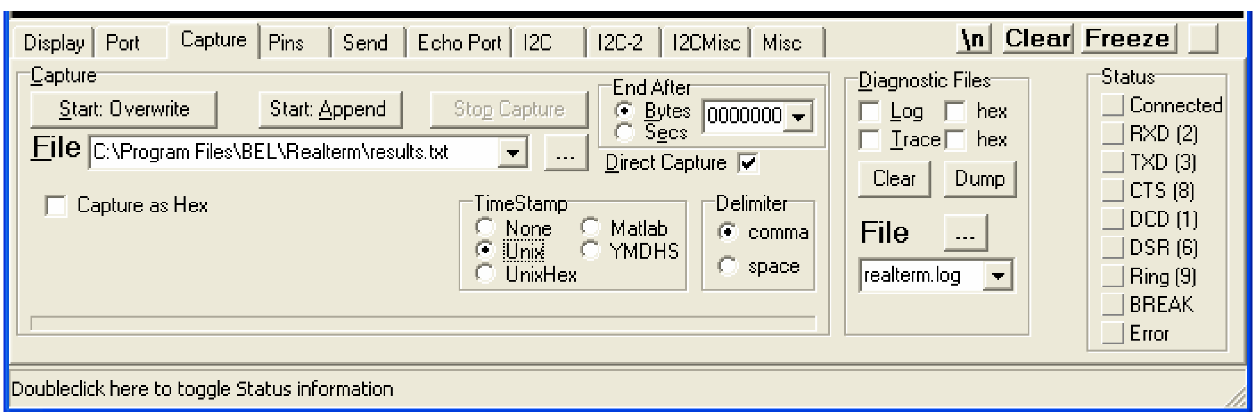


Figure 16: Realterm - Capture Tab

Details of Output File

1) The RealTerm output will be saved as a text file in specified location
2) The data will be in the following format:
   - 1236366080,f~ □ □¢, @H|Æÿþ□,1125,18
   - An example of the file is shown in Figure 17

3) Explanation of output
   - 1236366080 – Unix Timestamp which can be converted to readable time
   - @H – Node Identifier, this will be different for each XBee module
   - 1125,18 - Small particle Count, Large Particle Count

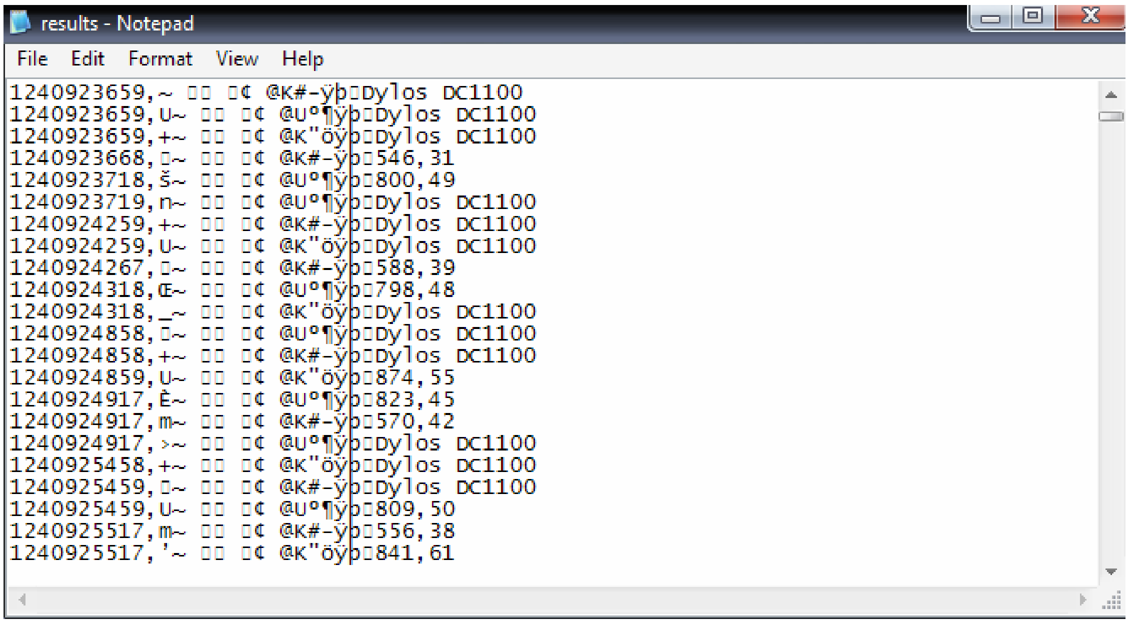


Figure 17: RealTerm Output File (result.txt)

<u>PHASE 2:</u> "Data Conversion and Analysis"

IMPORTANT: Before proceeding, please verify that the RealTerm output file (results.txt) is located in the folder "RealTerm Output", which should be located on the desktop.

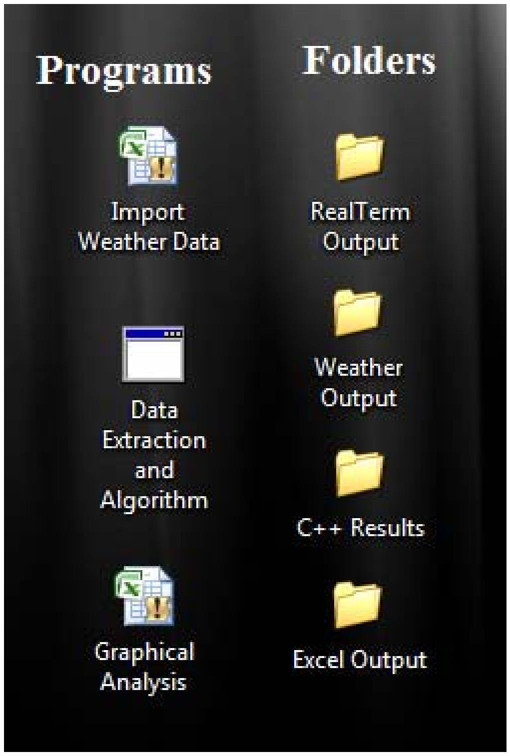


Figure 18: All Phase 2 Programs and Output Folders (Located on the Desktop)

1. Import Weather Data

This is the first software that must run before analyzing the RealTerm output file (results.txt). The Import Weather Data software is an important tool that automatically retrieves weather information from an internet source. (weather.myphl17.com) This is an Excel based tool that utilizes a VBA Macro.

User Input

The user must first open the file on the desktop labeled "Import Weather Data." The user must choose the start and end dates, as well as, the closest location in order to capture the best weather information. The user must make each choice through the use of pull-down menus. The user input page is shown in Figure 19.

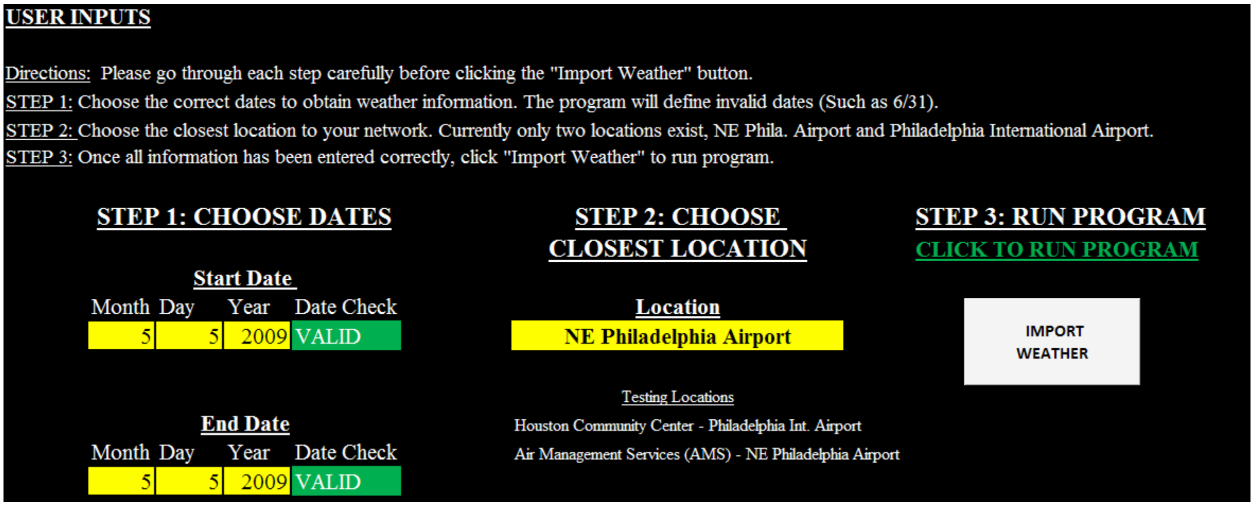
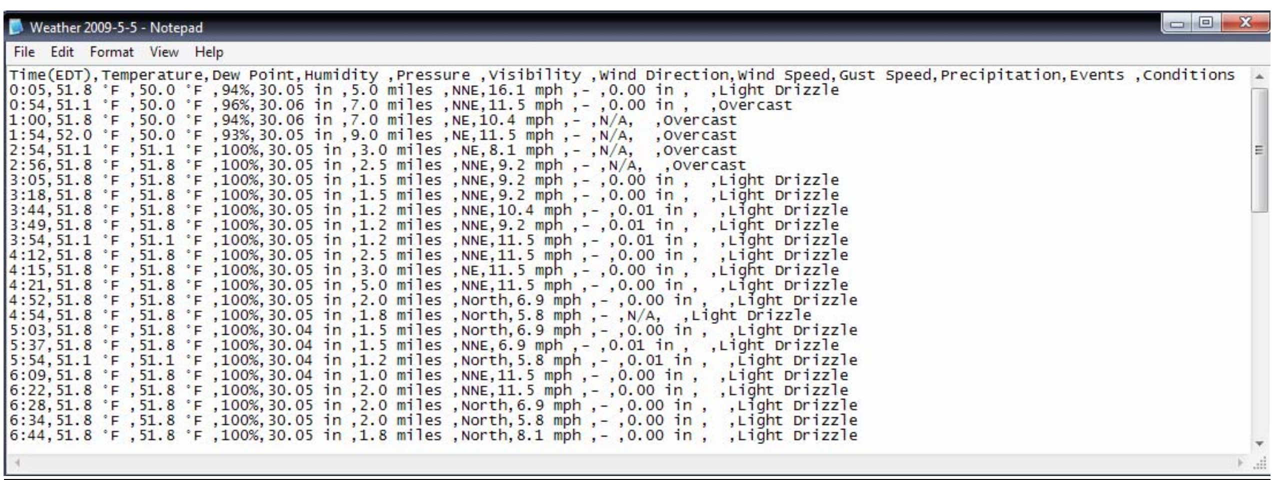


Figure 19: User Inputs for “Import Weather Data” Program

Output Files

The outputs of this program are numerous comma separated value (.csv) files. A file is created for each day and is saved in the “Weather Output” folder on the desktop. An example of the output file is shown in Figure 20.



```
Weather 2009-5-5 - Notepad
File  Edit  Format  View  Help
Time(EDT),Temperature,Dew Point,Humidity ,Pressure ,Visibility ,Wind Direction,Wind Speed,Gust Speed,Precipitation,Events ,Conditions
0:05,51.8 °F ,50.0 °F ,94%,30.05 in ,5.0 miles ,NNE,16.1 mph ,- ,0.00 in ,  ,Light Drizzle
0:54,51.1 °F ,50.0 °F ,96%,30.06 in ,7.0 miles ,NNE,11.5 mph ,- ,0.00 in ,  ,Overcast
1:00,51.8 °F ,50.0 °F ,94%,30.06 in ,7.0 miles ,NE,10.4 mph ,- ,N/A,  ,Overcast
1:54,52.0 °F ,50.0 °F ,93%,30.05 in ,9.0 miles ,NE,11.5 mph ,- ,N/A,  ,Overcast
2:54,51.1 °F ,51.1 °F ,100%,30.05 in ,3.0 miles ,NE,8.1 mph ,- ,N/A,  ,Overcast
2:56,51.8 °F ,51.8 °F ,100%,30.05 in ,2.5 miles ,NNE,9.2 mph ,- ,N/A,  ,Overcast
3:05,51.8 °F ,51.8 °F ,100%,30.05 in ,1.5 miles ,NNE,9.2 mph ,- ,0.00 in ,  ,Light Drizzle
3:18,51.8 °F ,51.8 °F ,100%,30.05 in ,1.5 miles ,NNE,9.2 mph ,- ,0.00 in ,  ,Light Drizzle
3:44,51.8 °F ,51.8 °F ,100%,30.05 in ,1.2 miles ,NNE,10.4 mph ,- ,0.01 in ,  ,Light Drizzle
3:49,51.8 °F ,51.8 °F ,100%,30.05 in ,1.2 miles ,NNE,9.2 mph ,- ,0.01 in ,  ,Light Drizzle
3:54,51.1 °F ,51.1 °F ,100%,30.05 in ,1.2 miles ,NNE,11.5 mph ,- ,0.01 in ,  ,Light Drizzle
4:12,51.8 °F ,51.8 °F ,100%,30.05 in ,2.5 miles ,NNE,11.5 mph ,- ,0.00 in ,  ,Light Drizzle
4:15,51.8 °F ,51.8 °F ,100%,30.05 in ,3.0 miles ,NE,11.5 mph ,- ,0.00 in ,  ,Light Drizzle
4:21,51.8 °F ,51.8 °F ,100%,30.05 in ,5.0 miles ,NNE,11.5 mph ,- ,0.00 in ,  ,Light Drizzle
4:52,51.8 °F ,51.8 °F ,100%,30.05 in ,2.0 miles ,North,6.9 mph ,- ,0.00 in ,  ,Light Drizzle
4:54,51.8 °F ,51.8 °F ,100%,30.05 in ,1.8 miles ,North,5.8 mph ,- ,N/A,  ,Light Drizzle
5:03,51.8 °F ,51.8 °F ,100%,30.04 in ,1.5 miles ,North,6.9 mph ,- ,0.00 in ,  ,Light Drizzle
5:37,51.8 °F ,51.8 °F ,100%,30.04 in ,1.5 miles ,NNE,6.9 mph ,- ,0.01 in ,  ,Light Drizzle
5:54,51.1 °F ,51.1 °F ,100%,30.04 in ,1.2 miles ,North,5.8 mph ,- ,0.01 in ,  ,Light Drizzle
6:09,51.8 °F ,51.8 °F ,100%,30.04 in ,1.0 miles ,NNE,11.5 mph ,- ,0.00 in ,  ,Light Drizzle
6:22,51.8 °F ,51.8 °F ,100%,30.05 in ,2.0 miles ,NNE,11.5 mph ,- ,0.00 in ,  ,Light Drizzle
6:28,51.8 °F ,51.8 °F ,100%,30.05 in ,2.0 miles ,North,6.9 mph ,- ,0.00 in ,  ,Light Drizzle
6:34,51.8 °F ,51.8 °F ,100%,30.05 in ,2.0 miles ,North,5.8 mph ,- ,0.00 in ,  ,Light Drizzle
6:44,51.8 °F ,51.8 °F ,100%,30.05 in ,1.8 miles ,North,8.1 mph ,- ,0.00 in ,  ,Light Drizzle
```

Figure 20: Output File from “Import Weather Data” Program

2. Data Extraction and Algorithm Implementation

The “Data Extraction and Algorithm Implementation” program is the most important portion of the software package. This program reads in both the RealTerm output and the weather data. These files are used to effectively perform the particle count to mass conversion algorithm. More specifically, the weather data is used to apply correction factors to the original algorithm. This program will output multiple files, including a warning report, an average report, and data reports for each location.

<u>User Input</u>

The user must first run the executable file “Data Extraction and Algorithm” located on the desktop. The user will then be asked to enter the deployment location – AMS or Houston Community Center. (Additional deployment locations can be added, but this would require updating the node locations within the source code.)

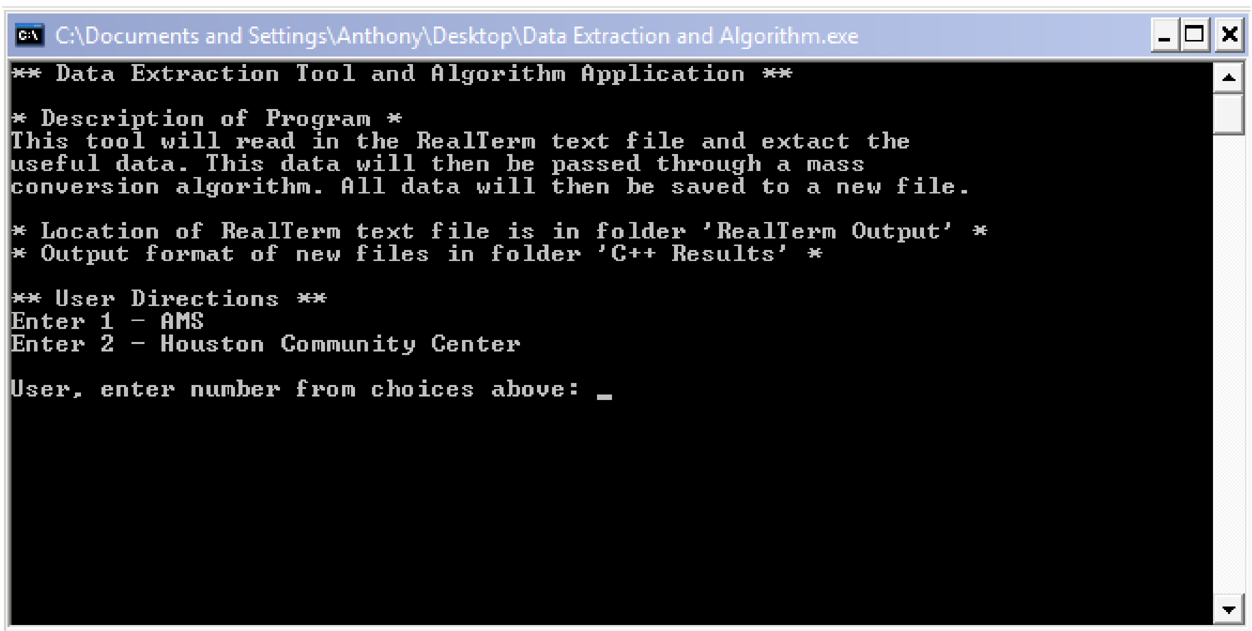


Figure 21: User Input for “Data Extraction and Algorithm App.”

<u>Output Files</u>

The outputs of this program are numerous text (.txt) files. These files are located in the “C++ Results” folder, which is located on the desktop. The output files can be seen in Figure 21. An example of the output file containing “All Data” is shown in Figure 22.

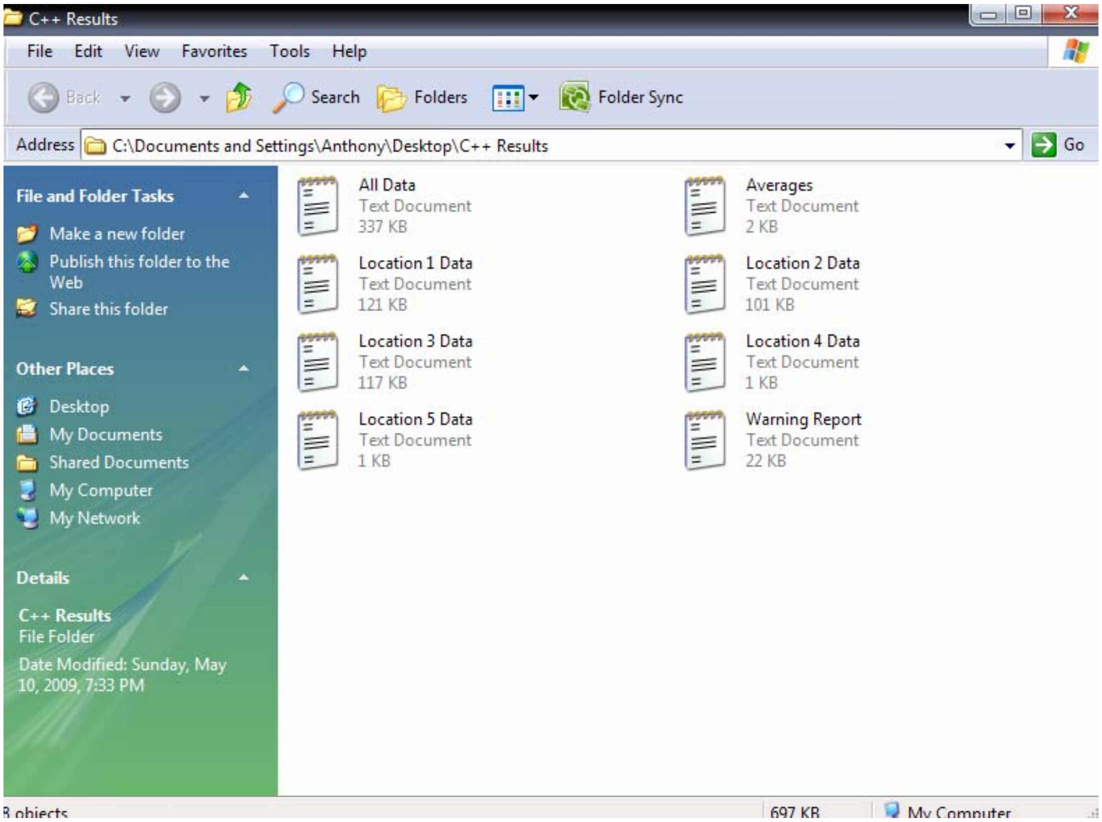
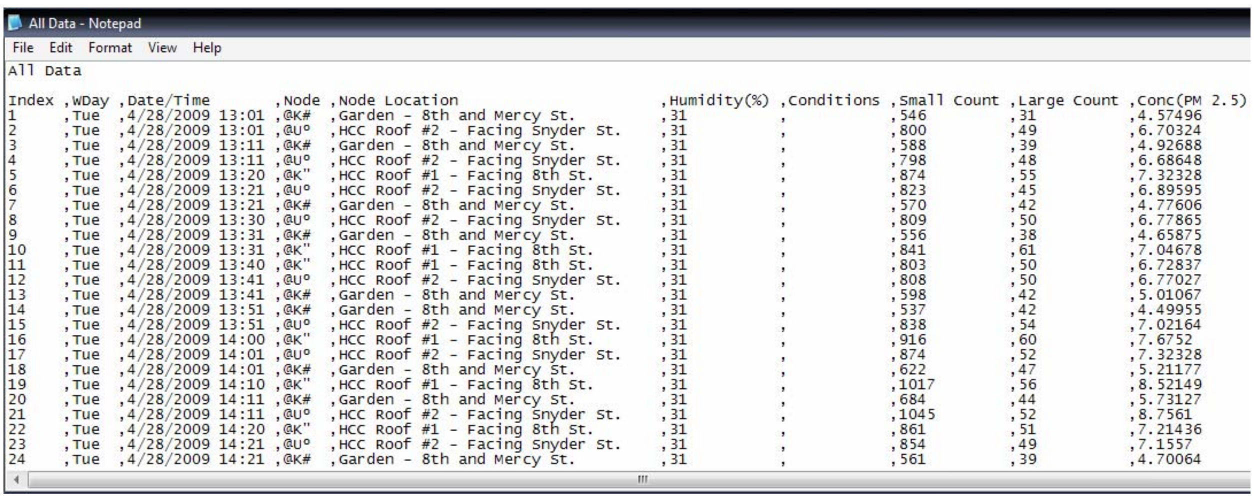


Figure 21: C++ Results Folder



All Data

Index	WDay	Date/Time	Node	Node Location	Humidity(%)	Conditions	Small Count	Large Count	Conc(PM 2.5)
1	Tue	4/28/2009 13:01	@K#	Garden - 8th and Mercy St.	31		546	31	4.57496
2	Tue	4/28/2009 13:01	@U°	HCC Roof #2 - Facing Snyder St.	31		800	49	6.70324
3	Tue	4/28/2009 13:11	@K#	Garden - 8th and Mercy St.	31		588	39	4.92688
4	Tue	4/28/2009 13:11	@U°	HCC Roof #2 - Facing Snyder St.	31		798	48	6.68648
5	Tue	4/28/2009 13:20	@K"	HCC Roof #1 - Facing 8th St.	31		874	55	7.32328
6	Tue	4/28/2009 13:21	@U°	HCC Roof #2 - Facing Snyder St.	31		823	45	6.89595
7	Tue	4/28/2009 13:21	@K#	Garden - 8th and Mercy St.	31		570	42	4.77606
8	Tue	4/28/2009 13:30	@U°	HCC Roof #2 - Facing Snyder St.	31		809	50	6.77865
9	Tue	4/28/2009 13:31	@K#	Garden - 8th and Mercy St.	31		556	38	4.65875
10	Tue	4/28/2009 13:31	@K"	HCC Roof #1 - Facing 8th St.	31		841	61	7.04678
11	Tue	4/28/2009 13:40	@K"	HCC Roof #1 - Facing 8th St.	31		803	50	6.72837
12	Tue	4/28/2009 13:41	@U°	HCC Roof #2 - Facing Snyder St.	31		808	50	6.77027
13	Tue	4/28/2009 13:41	@K#	Garden - 8th and Mercy St.	31		598	42	5.01067
14	Tue	4/28/2009 13:51	@K#	Garden - 8th and Mercy St.	31		537	42	4.49955
15	Tue	4/28/2009 13:51	@U°	HCC Roof #2 - Facing Snyder St.	31		838	54	7.02164
16	Tue	4/28/2009 14:00	@K"	HCC Roof #1 - Facing 8th St.	31		916	60	7.6752
17	Tue	4/28/2009 14:01	@U°	HCC Roof #2 - Facing Snyder St.	31		874	52	7.32328
18	Tue	4/28/2009 14:01	@K#	Garden - 8th and Mercy St.	31		622	47	5.21177
19	Tue	4/28/2009 14:10	@K"	HCC Roof #1 - Facing 8th St.	31		1017	56	8.52149
20	Tue	4/28/2009 14:11	@K#	Garden - 8th and Mercy St.	31		684	44	5.73127
21	Tue	4/28/2009 14:11	@U°	HCC Roof #2 - Facing Snyder St.	31		1045	52	8.7561
22	Tue	4/28/2009 14:20	@K"	HCC Roof #1 - Facing 8th St.	31		861	51	7.21436
23	Tue	4/28/2009 14:21	@U°	HCC Roof #2 - Facing Snyder St.	31		854	49	7.1557
24	Tue	4/28/2009 14:21	@K#	Garden - 8th and Mercy St.	31		561	39	4.70064

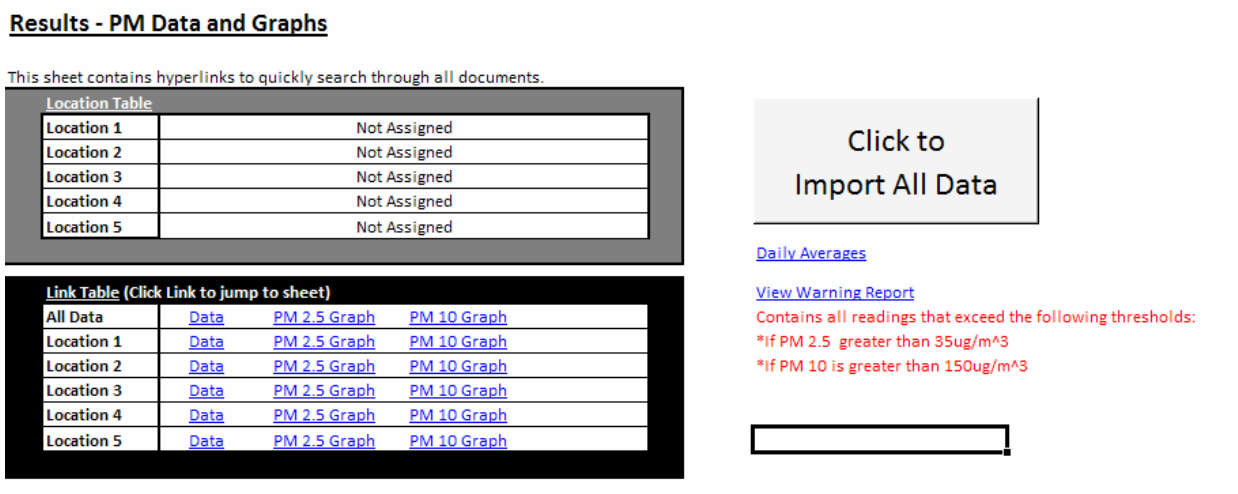
Figure 22: "All Data" Output File

3. Graphical Analysis

The graphical analysis program is the final program to execute. This program will automatically update an excel template file with the “C++ Results” files shown in Figure 21.

User Input

The user must first open the “Graphical Analysis” program located on the desktop. The user must then run the program by clicking the macro enabled button “Import All Data”. This is shown in Figure 23.



Results - PM Data and Graphs

This sheet contains hyperlinks to quickly search through all documents.

Location Table

Location 1	Not Assigned
Location 2	Not Assigned
Location 3	Not Assigned
Location 4	Not Assigned
Location 5	Not Assigned

Link Table (Click Link to jump to sheet)

All Data	Data	PM 2.5 Graph	PM 10 Graph
Location 1	Data	PM 2.5 Graph	PM 10 Graph
Location 2	Data	PM 2.5 Graph	PM 10 Graph
Location 3	Data	PM 2.5 Graph	PM 10 Graph
Location 4	Data	PM 2.5 Graph	PM 10 Graph
Location 5	Data	PM 2.5 Graph	PM 10 Graph

Click to Import All Data

Daily Averages

View Warning Report

Contains all readings that exceed the following thresholds:

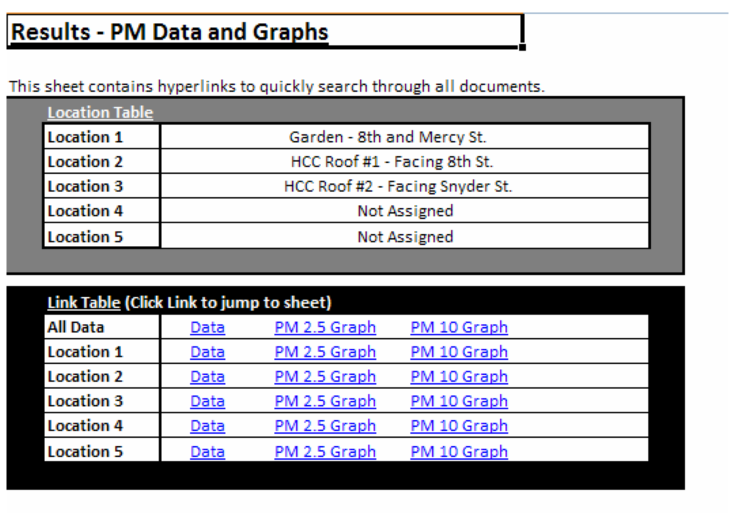
*If PM 2.5 greater than 35ug/m^3

*If PM 10 is greater than 150ug/m^3

Figure 23: “Graphical Analysis” Program User Controls

Output Files

The outputs of this program are numerous Excel worksheets within one workbook. The first sheet contains the location names, as well as, hyperlinks for easy access to other sheets. This is shown in Figure 24. These other sheets include a warning report (Figure 25), an average report (Figure 26), and also an array of graphs for each location.(Figure 27)



Results - PM Data and Graphs

This sheet contains hyperlinks to quickly search through all documents.

Location Table

Location 1	Garden - 8th and Mercy St.
Location 2	HCC Roof #1 - Facing 8th St.
Location 3	HCC Roof #2 - Facing Snyder St.
Location 4	Not Assigned
Location 5	Not Assigned

Link Table (Click Link to jump to sheet)

All Data	Data	PM 2.5 Graph	PM 10 Graph
Location 1	Data	PM 2.5 Graph	PM 10 Graph
Location 2	Data	PM 2.5 Graph	PM 10 Graph
Location 3	Data	PM 2.5 Graph	PM 10 Graph
Location 4	Data	PM 2.5 Graph	PM 10 Graph
Location 5	Data	PM 2.5 Graph	PM 10 Graph

Figure 24: “Graphical Analysis” Output – Node Location Names

Warning Report							
If PM 2.5 > 35ug/m^3							
If PM 10 > 150g/m^3							
WDay	Date/Time	Node Location	Humidity(%)	Conditions	Conc(PM 2.5)	Conc(PM 10)	Message
Fri	5/1/2009 1:10	Garden - 8th and Mercy St.	94		10.5441	164.553	Warning PM 10 too high
Fri	5/1/2009 1:20	Garden - 8th and Mercy St.	94		11.5037	181.814	Warning PM 10 too high
Fri	5/1/2009 1:31	Garden - 8th and Mercy St.	94		11.3356	181.217	Warning PM 10 too high
Fri	5/1/2009 1:31	HCC Roof #2 - Facing Snyder St.	94		12.307	164.6	Warning PM 10 too high
Fri	5/1/2009 1:40	HCC Roof #1 - Facing 8th St.	94		14.4041	150.395	Warning PM 10 too high
Fri	5/1/2009 1:41	Garden - 8th and Mercy St.	94		12.7526	199.794	Warning PM 10 too high
Fri	5/1/2009 1:41	HCC Roof #2 - Facing Snyder St.	94		13.0985	169.252	Warning PM 10 too high
Fri	5/1/2009 1:50	HCC Roof #1 - Facing 8th St.	94		15.3813	161.668	Warning PM 10 too high
Fri	5/1/2009 1:51	Garden - 8th and Mercy St.	94		13.079	203.552	Warning PM 10 too high
Fri	5/1/2009 1:51	HCC Roof #2 - Facing Snyder St.	94		13.3369	166.059	Warning PM 10 too high
Fri	5/1/2009 2:00	Garden - 8th and Mercy St.	88		17.6005	191.343	Warning PM 10 too high
Fri	5/1/2009 2:01	HCC Roof #2 - Facing Snyder St.	88		19.7933	195.252	Warning PM 10 too high
Fri	5/1/2009 2:01	HCC Roof #1 - Facing 8th St.	88		20.7155	159.71	Warning PM 10 too high
Fri	5/1/2009 2:10	HCC Roof #1 - Facing 8th St.	88		20.9158	178.786	Warning PM 10 too high
Fri	5/1/2009 2:11	Garden - 8th and Mercy St.	88		17.197	199.091	Warning PM 10 too high
Fri	5/1/2009 2:11	HCC Roof #2 - Facing Snyder St.	88		19.8345	196.151	Warning PM 10 too high
Fri	5/1/2009 2:20	HCC Roof #1 - Facing 8th St.	88		20.0541	170.202	Warning PM 10 too high
Fri	5/1/2009 2:21	Garden - 8th and Mercy St.	88		16.4835	201.809	Warning PM 10 too high
Fri	5/1/2009 2:21	HCC Roof #2 - Facing Snyder St.	88		17.8859	192.916	Warning PM 10 too high
Fri	5/1/2009 2:30	Garden - 8th and Mercy St.	88		16.983	206.169	Warning PM 10 too high

Figure 25: "Graphical Analysis" Output – Warning Report

Daily Averages (All Locations)							
Location	WDay	Year	Month	Day	PM 2.5 Average	PM 10 Average	Message
Garden - 8th and Mercy St.	Wed	2009	4	29	4.61978	19.2471	
HCC Roof #2 - Facing Snyder St.	Wed	2009	4	29	5.95159	24.5803	
HCC Roof #1 - Facing 8th St.	Wed	2009	4	29	5.37765	22.5017	
HCC Roof #1 - Facing 8th St.	Thu	2009	4	30	4.15855	15.892	
Garden - 8th and Mercy St.	Thu	2009	4	30	3.47878	16.6734	
HCC Roof #2 - Facing Snyder St.	Thu	2009	4	30	3.73218	15.5493	
HCC Roof #1 - Facing 8th St.	Fri	2009	5	1	7.12313	50.4017	
Garden - 8th and Mercy St.	Fri	2009	5	1	5.81711	51.772	
HCC Roof #2 - Facing Snyder St.	Fri	2009	5	1	6.55181	53.7101	
Garden - 8th and Mercy St.	Sat	2009	5	2	19.1732	143.84	
HCC Roof #2 - Facing Snyder St.	Sat	2009	5	2	18.4497	117.262	
HCC Roof #1 - Facing 8th St.	Sat	2009	5	2	19.2174	108.316	
HCC Roof #1 - Facing 8th St.	Sun	2009	5	3	8.59338	22.7369	
Garden - 8th and Mercy St.	Sun	2009	5	3	8.43104	37.4068	
HCC Roof #2 - Facing Snyder St.	Sun	2009	5	3	7.67917	23.1048	

Figure 26: "Graphical Analysis" Output – Average Report

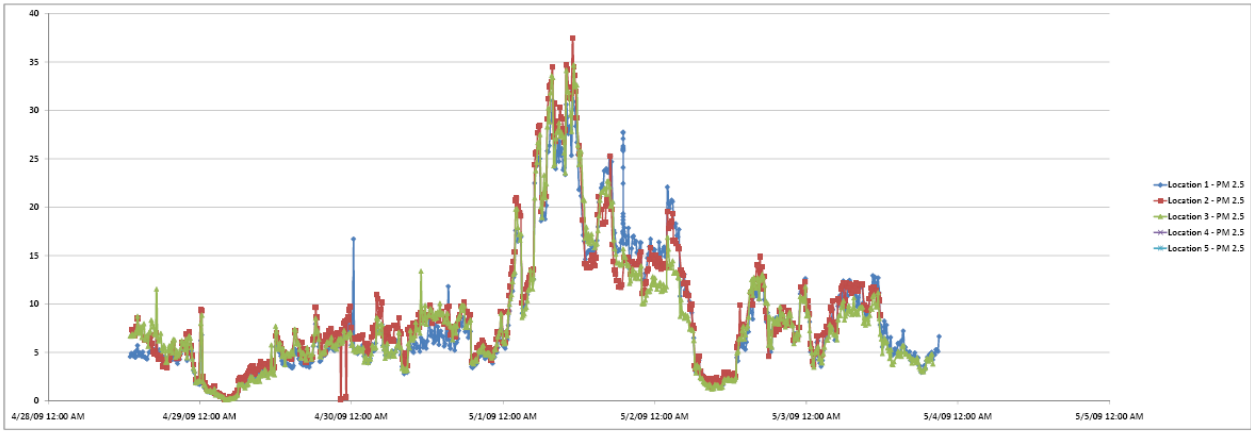


Figure 27: "Graphical Analysis" Output – All Locations (PM2.5)

Anthony McClellan

15 Elmwood Avenue
Norwood, PA 19074
(C) 484-574-2164
anthony.s.mcclellan@drexel.edu
anthony.mcclellan@gmail.com

Education
Drexel University, Philadelphia, PA - Graduation June, 2009
Bachelor of Science in Electrical Engineering, Minor in Business Administration
Cumulative GPA - 3.4

Skills
- Software: Microsoft Office, Microsoft Visio, AutoCad, MatLab, PSpice, LabView, Maple, Zuken E3 Cable Tool
- Programming Languages: Basics of HTML, C++, Java, and VDHL
- Engineering Design Projects: Improved various skills including leadership, organization, and research
- Relevant Coursework: Communication Systems, Lightwave Engineering, Wireless and Optical Electronics, Wireless Communication Systems, Modulation and Coding, Digital Signal Processing, numerous Design Labs

Experience

L-3 Communications, Camden, NJ
Hardware Engineering Co-op, April 2008 - Present
- Received clearance in order to work on government contracts
- Assisted various engineers (EE,ME, Systems) in the design of L-3's Integrated Communication System
- Gained experience with numerous types of equipment (designed by both L-3 and outside vendors):
  - Internal Comms - voice terminals, broadcast and alarm systems
  - External Comms - Radios (HF, UHF, VHF)
  - Networks - Cameras, Computer Workstations, Entertainment and Training Systems
- Experienced broad view of L-3's design processes:
  - High level system view of all equipment connections
  - Lower level view of the equipment distribution by room location
  - Lowest level view of the cable design
- Utilized E3 Cabling Tool to designate appropriate mating connectors, pins, signal types/functions
- Participated in CFR (Code of Federal Regulations) testing procedures on an assembled communication's rack
- Created various organizational tools - including an obsolete parts listing and a voice switch planning tool

Comcast, Philadelphia, PA
Voice Engineer (Comcast Digital Voice), April 2007 to September 2007
- Assisted Market Design Engineers with design/upgrading processes and equipment purchasing
- Gained experience with various types of equipment through site surveys
  - Switches (Cisco BTS and Cedar Point), MGX, DACS, routers, servers, test gear, and a variety of fibers
- Interacted with outside vendors/integrators - including Cisco, Agilent, Motorola, Tellabs, Sun, Cedar Point, etc.
- Acted as Project Planner for test server upgrades for all switch locations around the United States
- Developed a nationwide network diagram to demonstrate all current and planned switch locations
- Created various organizational tools for documenting unused equipment to prevent budget complications

V-Comm Telecommunications Engineering, Blue Bell, PA
Network Engineering Co-op, April 2006 to September 2006
- Assisted engineers in the design process of public switched telephone networks (PSTN)
- Supported engineers in various Switch Interconnection Configuration Plans (SICP) for Comcast's network
- Obtained an understanding of the general principals of telephony from a switching perspective
- Gained experience in researching existing network designs (As-Builts), subscriber forecasts, and LERG databases
- Utilized Microsoft Office and Visio, for organizing design data and drawing network design diagrams

Honors and Awards
- Drexel University Dean's Scholar Award, 2004-09
- Drexel University Dean's List- Winter 2004-05, Winter 2006-07, Fall 2007-08, Winter 2007-08, Fall 2008-09
- Pennsylvania Higher Education Assistance Agency (PHEAA) Academic Excellence Scholar, 2004-09
- Member of Phi Eta Sigma National Honor Society, Drexel University Chapter, 2004-05

Robert J Falcone
387 Westbourne Dr
Broomall, PA 19008-3739
robert.john.falcone@drexel.edu

Education

Drexel University, Philadelphia, PA
Bachelor of Science in Electrical Engineering, Degree Expected June 2009
Cumulative 3.71 GPA

Relevant Coursework

Electromagnetic Fields and Waves
Transform Methods and Filtering
Programming for Engineers
Introduction to Modulation and Coding
Electrical Engineering Laboratory
Engineering Ethics and Professionalism

Work Experience

Ewingcole, Philadelphia, PA
Electrical Engineering Co-op, Sports and Entertainment Group, April 2007-September 2007
- Assisted in the design of electrical systems for the $1.6 billion New Meadowlands Stadium
- Performed load and demand calculations for various electrical products
- Analyzed panel capacities and designated circuits accordingly
- Created layout drawings and aiming patterns based on vendor specifications for Stadium sports-lighting
- Drafted schematic diagrams of Stadium electrical rooms

Lockheed Martin Maritime Systems and Sensors, Moorestown, NJ
Co-op Technical Senior, Ship Electrical Systems, April 2006-September 2006
- Created and maintained technical drawings supporting various Naval combat systems
- Executed quality control and verification checks of various program electrical drawings
- Collaborated on the development of a baseline drawing to be used across several US Navy Cruiser hulls

Drexel University, Philadelphia, PA
Executive Ambassador, September 2004- Present
- Promote Drexel University to prospective students and their families through guided tours
- Participate in interviewing candidates for the Ambassador position

All Star Baseball Academy, Broomall, PA
Instructor, May 2001-April 2007
- Instructed youth players in the fundamentals of baseball
- Directed several of the Academy's instructional programs
- Scheduled appointments and managed accounts for Academy clients

Leadership and Organizational Experience

The DAC Pack, Drexel University Athletics
President, April 2007-Present
- Increased revenue over 400% during tenure through advertising and partnership agreements
- Manage a budget of over $50,000
- Build a portfolio of corporate and University sponsors
- Draft, negotiate, and sign partnership contracts
- Design and implement creative marketing strategies and campaigns

Drexel University, Philadelphia, PA
Overnight and Day Visit Program Coordinator, February 2006-March 2007
- Restructured the previous Overnight and Day Visit Program in place
- Planned visits to Drexel University for selected high school seniors
- Fielded calls and questions from prospective students and families
- Trained and direct a team of Drexel University Student Hosts

Engineering Design and Publication Experience

The Smart Borders Engineering Design Team, Drexel University
Project Leader, January 2005-June 2005

- Researched several technologies considered for use in securing the United States' borders
- Delegated tasks, integrated separate design components, and assembled final document for submission
- Presented design before a panel of faculty and students
- Selected for presentation at the 85th Annual Meeting of the Transportation Research Board in Washington, DC
- Published in the 2006 edition of the Transportation Research Record

High Profile Public Speaking Experience

Democratic Presidential Debate, Drexel University
Co-Host, October 2007

- Emceed viewing party attended by students and world renowned political figures
- Greeted and introduced Democratic Presidential Candidates on stage

Anthony J. Drexel Society Gala, Drexel University
Student Ambassador, November 2007

- Selected to participate in Drexel University's annual gala event
- Dined and interacted with prominent university benefactors and donors

Discover Drexel Day Open House , Drexel University
Emcee and Keynote Speaker, October 2007

- Delivered welcoming speech to over 500 high ability visiting students
- Introduced guest speakers and provided direction for execution of tasks

Software Skills and Proficiencies

- Microsoft Office
- AutoCAD
- Lab View
- MATLAB
- Maple 9.5
- PSPICE

Mark Uva
3175 JFK Blvd, Apt. 814
Philadelphia, PA 19104
973-727-3321
mark.a.uva@drexel.edu

Drexel University, Philadelphia, PA
Bachelor of Science in Electrical Engineering, Anticipated Graduation - June, 2009
Cumulative GPA: 3.67

Experience

Drexel University ACIN Program, Camden, NJ
Modeling and Simulation Co-op, April 2008 to Present
- Developed Diffraction code to estimate loss due to obstructions in terrain
- Developed scenarios in JCSS for comparison with COMPOSER
- Orchestrated comparison between JCSS and COMPOSER with Lockheed Martin
- Authored white paper describing issues with Terrain in JCSS as well as quarterly reports describing progress
- Attended annual JCSS conference at the Pentagon

L-3 Communications CS-East, Camden, NJ
Systems Engineering Co-op, April to September, 2007
- Administrated and assisted engineers with Systems Engineering tools, including Doors, as well as day to day operations.
- Received security clearance to work on projects requiring clearance.
- Tracked day to day usage of Doors by all users to determine if the amount of licenses was adequate.
- Developed scripts, using Perl and DXL languages to assist in day to day operations and maintenance.
- Updated Processes, including training powerpoints, tables, and Visio flow charts to reflect newest process updates.

V-comm, L.L.C., Blue Bell, PA
RF co-op, April to September, 2006
- Assisted in design, implementaion, and operation of cellular, PCS, and DVB-H networks.
- Fine-tuned existing wireless networks to provide better service.
- Prepared coverage plots and documents using MapInfo, NetPlan, EDX and Terrain Navigator Pro.
- Researched up and coming technologies such as WiMAX and DVB-H to so the company could become more adept in those fields

Centralized Odor System (Freshman Design)

- Conducted direct market research & online research analysis
- Presented final report to faculty and fellow students

Skills

Software: Microsoft Office, DOORS, NetPlan, EDX, MapInfo Professional, Terrain Navigator Pro, Maple, MATLAB, PSpice, LaTex writing
Programming Languages: Basics of Java, MATLAB, and C++

Honors and Awards

- Drexel Dean's List: Fall, Winter, Spring 04-05; Winter 05-06; Winter 06-07; Fall, Winter 07-08
- 2006/2007 Inter Fraternity Council Athlete of the Year
- AJ Drexel Scholar - June 2004-present
- Drexel University Pennoni Honors College Member - September 2004-present
- Edward J. Bloustein Scholar

Leadership and Organizational Experience

Sigma Phi Epsilon fraternity
-Chaplain: Oversaw standards board, ruled on judicial matters, and authored new bylaws
-Internal Relations Chair: Oversaw executive board, made sure all members were doing their job and set goals for each e-board member
-Community Service Chair: Worked with community groups and other student organizations to set up worthwhile community service opportunities
-Athletic Chair: Set up practices and determined teams for each sport

Dac Pack
-Executive Board member: Worked with other members to plan trips and events for students

Relevant Coursework

Calculus I,II,III	Physics I,II,III	Programming for Engineers (Java)
Fundamentals of Intellegent Systems	Electric Circuits	Energy I, II
Systems I, II	Linear Modeling	Vector Analysis
Transform Methods	Digital Signal Processing	Electronic Devices
Analog Electronics	Intro to Modulation and Coding	Electromagnetic Fields and Waves
Wireless Communications	Wireless and Optical Electronics	ECE Lab I,II,III,IV
Communications I,II,III	Deterministic Signal Processing	Statistical Signal Processing

Edward W Ostapowicz
3867 Marsh Rd
Garnet Valley, PA 19061
610-494-2664
edward.w.ostapowicz@drexel.edu

Education:

Drexel University, Philadelphia, PA
Bachelor of Science in Electrical Engineering, Anticipated Graduation - June, 2009
GPA: 3.73

Relevant Coursework:

Fundamentals in Intelligent Systems
Electrical Engineering Lab I, II, III, IV
Electronic Devices
Energy Management Principles
Power Systems
Basic Java Programming
Fundamentals in Signals and Systems
Motor Control Principles
Power Electronics
Systems and Control

Computer Skills:

- Basic Java
- Basic SKM Power Tools
- Basic Maple
- Basic MATLAB
- Basic AutoCAD
- Microsoft Office
- Microsoft Operating Systems
- Basic Microstation

Work Experience:

Sunoco, Inc., Philadelphia, PA
Instrument/Electrical Reliability Engineer Co-op, October 2007 to April 2008
- Updated database using EMPAC software
- Designed relay rack mount using Microsoft Visio
- Assisted in design of outdoor Class A Div II relay mount
- Assisted in visual inspections of electrical equipment in the refinery
- Attended many vendor sessions to learn about new equipment

KlingStubbins, Philadelphia, PA
Electrical Engineering Intern, September 2006 to April 2007
- Assisted in buildings' short circuit analyses
- Assisted in CADD design of buildings' electrical systems
- Traveled to Ft. Lee, VA for on site surveying
- Coordinated group of three to finish CADD drawings

Drexel University, Philadelphia, PA
Aide Engineer/Lab Tech, October 2005 to April 2006
- Prepared laboratories for daily classes
- Tested devices used in the labs
- Created supplemental components for particular labs
- Maintained laboratory equipment
- Controlled order within the laboratories

Eastern Electric, Mantua, NJ
Blueprint Analyst, June to August, 2000 and 2001
- Did the "take-offs" on blueprints for various future projects
- Formulated organized system of double checking the "take-offs"
- Interpreted different symbols from table to drawing
- Tabulated amounts of light fixtures, receptacles, and other equipment
- Managed time effectively to meet deadlines
- Reported final numbers to supervisor

Honors and Awards:

- Freshman Design Project Published (2006)
- Intramural Campus Team of the Year Co-Captain (2005)
- Perfect Attendance (2000-2004)
- A.J. Drexel Scholarship (College)
- CYO Basketball All-Star Selection (2004)
- Treasurer of the DAC Pack (2007-2008)
- Dean's List (2005-2007)
- Neumann Scholarship (High School and College)
- Football Student-Athlete (2004)