Portland State UNIVERSITY

Maseeh College of Engineering & Computer Science
Department of Electrical and Computer Engineering

Senior Capstone Project
Final Report

Open Bike Initiative Air Quality Sensor Hub

June 10, 2015

Team Members:
Ali Alavi
Robert Flory
Meng Lei
Pedro Munoz

Faculty Advisor: Malgorzata Chrzanowska-Jeske

Sponsors: Open Bike Inc. and Intel Corporation

Abstract:

Open Bike Initiative (OBI) is a bike sharing model in development that will incorporate a “smart-lock” capable of communicating with other devices. Every day, the road is shared by various commuters choosing different methods of travel such as OBI. While bikers and pedestrians gain the added benefits of physical activity, they also have the risk of increased pollution uptake with no method of identifying their level of exposure. This document outlines a low cost and mobile air quality sensor hub to be mounted on OBI bicycles. It will collect and store air quality and GPS data and is capable of reporting the information to the upcoming models of the OBI smart-lock via RJ45. The outline features the first two prototypes of the system which incorporate sensors for detecting different sizes of particulate matter, carbon monoxide, temperature, humidity, and location to be logged onto an SD card for analysis. They can be used to provide the data online and used to create a map of pollution for the city along paths of travel at different times of the day. These standalone prototypes were road tested and data was collected. Graphs of the sensor data were made and the GPS locations were used to create a map of the rides so air quality data could be referenced to location. Clear patterns in rise of Carbon Monoxide are observed in at-risk areas. Higher levels of particulate matter were also visible near a cement plant and high traffic areas. The cost of each unit is approximately $180. Next steps include further testing, communication programming, and optimizing size.

Complete specifications for the project including schematics, board layout, microcontroller code, test plans, among other documentation can be found on the github repository and wiki at:
https://github.com/alaviali/OBI/wiki

Table of Contents:

List of Figures:

List of Tables:

Introduction & Motivation

Open Bike initiative (OBI) is a developing bike sharing model launched in 2013 with open hardware and open source software. The design implements a locking mechanism unit called a “Smart-Lock” with wireless and cellular modules capable of attaching to any bicycle. The units communicate through a client and server side software in order to manage the OBI system. The smart-lock units are also capable of RS232 communication via RJ45 connections. This allows various sensors to be connected to the system in order to upload their data for cloud storage.

Primary outdoor air pollutants listed by the Environmental Protection Agency (EPA) are particulate matter (PM), carbon monoxide (CO), ground level ozone (O_3), nitrogen oxides (NO_x), sulfur oxides (SO_x) and lead (Pb)[1]. Every day, commuters such as bicycle riders and pedestrians are exposed to traffic-related pollution. While the government monitors the level of pollution in specific locations with stationary sensors, no reliable method of measuring pollution levels in all paths of travel exist. Considering the evidence suggesting the negative effects of ambient air pollution on human health (Katsouyanni 2003), it’s essential to have a method of evaluating pollutant exposure to commuters. Having this data could allow bicyclists and pedestrians to choose the optimal paths and times to travel in order to minimize their exposure.

By making low cost and mobile air quality sensing systems, creating a map of pollutant concentrations within paths of travel in a city could be created based on time of day. This would require a fairly large scale distribution in a non-invasive manner to encourage their use.

For our capstone project, our team was tasked with designing, building, and testing a small, low cost, and mobile, system with sensors used to measure real-time ambient air pollution correlated with location. It should be capable of collecting, storing, and sending the gathered data via an RJ45 connection to OBI Smart-Locks for cloud storage and access by all. With the data available for public access, this Internet of Things model would assist in creating a smarter city by allowing programmers, researchers, students, etc. to create maps, graphs, or apps that will provide commuters with the information they need to minimize their exposure to pollution. Additionally, our system was to provide a method of transferring power from a dynamo hub to the OBI Smart-Lock in order to charge the battery.

A similar project had been constructed by Alex Bigazzi called the Portland Ace. We used is experience to begin our search for suitable sensors. His project transfered air quality data to a app running on the rider’s cell phone.

Initially, our system was to be completely dependent on the OBI Smart-Locks for power and location data as the locks had both a battery and GPS modules. However as the project progressed, due to scheduling conflicts with the development of the OBI Smart-Lock as well as interest regarding our air quality system and its potential for use in many other avenues, the project evolved into a self contained system with its own power source that can either store the data locally, or send the data to future models of the OBI Smart-Lock for cloud storage.

Sensor Selection

Sensors for ground level ozone and its precursors cost hundreds of dollars and are beyond the budget of a low cost project such as this. Thus, the focus of this project was aimed at sensors for detecting Carbon Monoxide (CO) and Particulate Matter (PM). Additionally, as ambient pollutant concentration is affected by environmental factors such as climate, a temperature and humidity sensor was also chosen to be incorporated into the system. In order to gain a better understanding of the riders’ exposure to pollutants, it was decided that it would be beneficial to monitor the level of air intake by the rider. Therefore, a pulse sensor was also ordered and tested as pulse rate can be correlated with breathing patterns and subsequently the amount of pollutant laden air taken in by the bicycle rider. In order to choose the best

[1] http://www.epa.gov/oaqps001/urbanair/

sensor for each purpose, extensive research regarding each pollutant and the ideal sensors for their detection was conducted. Several projects with similar goals were reviewed and evaluated and the following sensors (summarized in Table 1) were chosen to be tested and potentially implemented into the system.

Quantity measured	**Sensor**	**Range**	**Power consumption**	**Output**	**Cost**
Small particulate matter	Sharp GP2Y1010AU0F	~<1um 0-0.5 mg/m^3	20mA-40mA	Analog	$11.95
Large particulate matter	Grove PPD42NS	~> 1um 0 - 8000 pcs / 0.01ft^3	90mA	Digital	$15.90
CO concentration	MQ-7	50-4000 ppm	42mA-150mA	Analog	$7.25
Temperature	DHT22	-40 - 80 celsius	Low	Digital	$9.95
Humidity		0-100% RH			
Pulse	PulseSensor	-	Low	Analog	$24.95

Table 1: Sensor summary

Two individual sensors were used for detecting PM. The Sharp sensor is used to detect smaller PM (~<1um) and consumes 20mA-40mA at 5V. It operates by counting the number of particles passing through an infrared beam supplied by an LED which is powered on for detection and outputs an analog value relative to the particle count. The reading time is 320 microseconds.

The Grove sensor is used to detect larger PM (~>1um) and consumes 90mA at 5V. It's operation is similar to that of the sharp sensor but uses a comparator to drive its output low based on particle size and concentration. Low pulses of 10-90ms are generated and particle concentration is calculated using the low pulse occupancy time of the sampling period. We used 1 second samples. If the bicycle is traveling 15mph this reading occurs during 22 feet of travel.

The MQ7 CO sensor is a metal oxide gas sensor that requires a constant 5V reference signal and outputs an analog voltage proportional to the ambient CO concentration. This CO concentration is detected using a CO sensitive coil with two phases of operation. In the "burn-off" or cleaning phase of operation, the coil uses 150mA at 5V for 60s to clear the accumulated CO particles from the coil. The data in this phase is not usable. In the detection phase of operation, the coil uses 42mA at 1.4V for 90s to accumulate CO which affects the resistance of the coil. Based on this resistance an output voltage is supplied proportional to the amount of ambient CO. In order to calibrate the output of the sensor to actual PPM values, a vacuum chamber environment with exactly 100 PPM of CO is required to detect the exact resistance and voltage of the sensor at that point, and to determine the correlation between output voltage and actual CO PPM. Since we did not have access to the necessary tools required for this type of calibration, we are only able to provide the relative concentration of CO.

The DHT22 detects both the ambient temperature and humidity in the air with very low power usage at 5V with a digital output. It is capable of detecting temperatures between -40 to 80 degrees celsius and detecting the relative humidity from 0 to 100%. It has a hardware limitation that only allows a reading once every two seconds. This period determined the maximum sampling rate for the project.

The PulseSensor is an open source sensor design used to detect the user's heart rate. It functions by using an LED to shine light through the user's finger or earlobe and measuring the reflection of the light to create an analog signal proportional to the user's pulse. After testing and evaluation, the pulse sensor was determined to be far too unreliable for practical use especially in constant movement such as on a bike ride. Alternatives to the PulseSensor for measuring heart rate were too expensive or too invasive (such as a wireless chest harness) and thus, the idea of using such a device was abandoned.

Initial Project Specification and Design Requirements

Initially, the sensor hub requirements were:

1. Be powered by a 3.3V line from the OBI Smart-Lock which also turns the system on.
2. Have a real time clock.
3. Take air quality readings at regular intervals and record this data, including the time the reading was taken. A sample rate of approximately 0.5Hz was proposed.
4. Store 10-15 minutes of records.
5. Have a micro USB connector to pass power to the Smart-Lock for charging batteries.
6. Have a two terminal connector to pass AC+ and AC- power from a dynamo hub to the Smart-Lock.
7. Have an RJ45 jack to connect to the Smart-Lock. The 8 signals to be passed through the connection were:
    a. RX and TX for serial communication
    b. 3.3V power from the Smart-Lock.
    c. 5V battery recharging line from micro USB.
    d. Cable cut detect.
    e. AC- and AC+ transmitting dynamo hub power to the Smart-Lock.
    f. Ground
8. Implement serial communication with the Smart-Lock through the RX/TX lines including:
    a. A command to set the Sensor Hub real time clock.
    b. Report the number of records stored in the Sensor Hub to the Smart-Lock.
    c. Provide a record to the Smart-Lock and subsequently delete the record from Sensor Hub storage.
9. Be contained in an enclosure suitable for mounting on the front of a bicycle with excellent airflow.
10. Be as cost effective as possible. Initial discussion included a $400 budget for the project.

Revised Project Specification and Design Requirements

As the project progressed, it became increasingly evident that the OBI team designing the Smart-Lock would not be able to successfully meet our schedule for various reasons. Consequently, our sponsors proposed that the scope of our project be expanded to include a simplified stand-in or "dummy" Smart-Lock system using an Arduino UNO with an RJ45 jack for minimal serial communication, GPS for providing location data, and SD card storage for recording the transmitted sensor data. Additionally, various power limitations further discussed in the "power" section of the report proved that the use of the 3.3V line from the Smart-Lock for powering the entire system would prove to be impossible. Thus, a battery became a necessary requirement for providing power.

As development progressed, further limitations with hardware and software began to emerge. GPS, memory storage, and serial communication working in conjunction proved to corrupt the transmitted data due to limitations with the Atmega328P microcontroller or Arduino serial communication protocols. These limitations are further discussed in the "Serial Communication" section of the report. Since the Arduino UNO uses the same microcontroller, it quickly became evident that the data would not be able to be transmitted to the Smart-Lock and stored to the SD memory. Thus, communication with the Smart-Lock was eliminated from the design for the time being and the GPS and SD memory were moved to the sensor hub in order to create a completely stand alone system. However, the RJ45 jack and necessary connections are still available for future implementation by simply adjusting the programming of the microcontroller.

The adjusted requirements for the system were determined to be:

11. Be powered by a rechargeable battery pack which also turns the system on when connected.
12. Have a real time clock.

13. Take air quality and GPS location readings at regular intervals and record this data, including the time the reading was taken. A sample rate of approximately 0.5Hz was proposed.
14. Large amount of internal storage for records with easy access.
15. Have a two terminal connector to pass AC+ and AC- power from a dynamo hub to the Smart-Lock.
16. Have an RJ45 jack to connect to future Smart-Locks. The 8 signals to be passed through the connection were:
    a. RX and TX for serial communication
    b. 3.3V power from the Smart-Lock.
    c. 5V battery recharging line from micro USB.
    d. Cable cut detect.
    e. AC- and AC+ transmitting dynamo hub power to the Smart-Lock.
    f. Ground
17. Be contained in an enclosure suitable for mounting on the front of a bicycle with excellent airflow.
18. Be as cost effective as possible. Initial discussion included a $400 budget for the project.

System Design and Component Selection

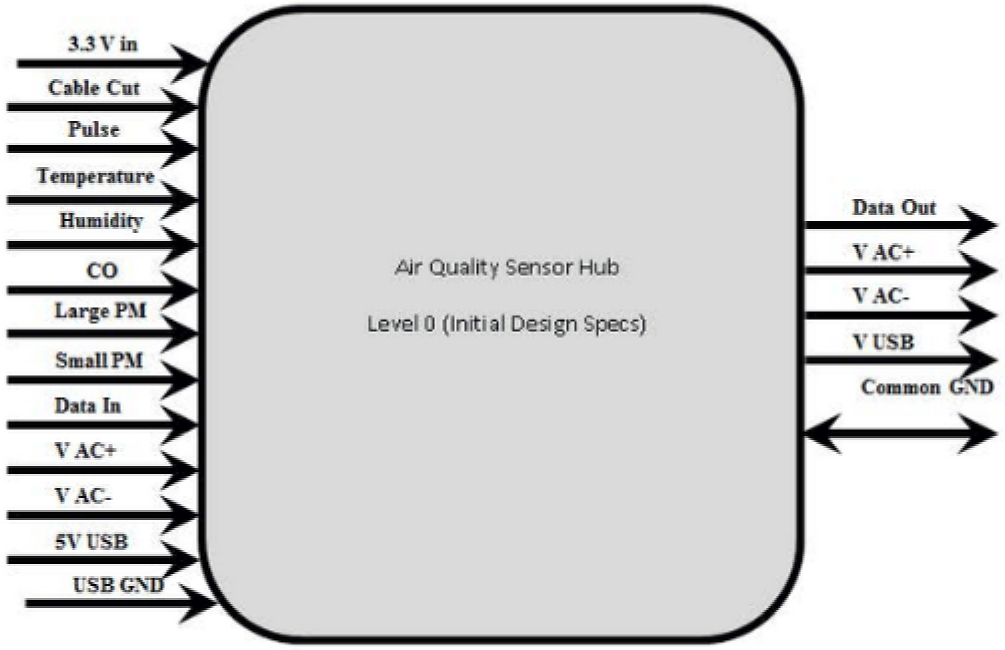


Figure 1: Initial Design (Level 0)

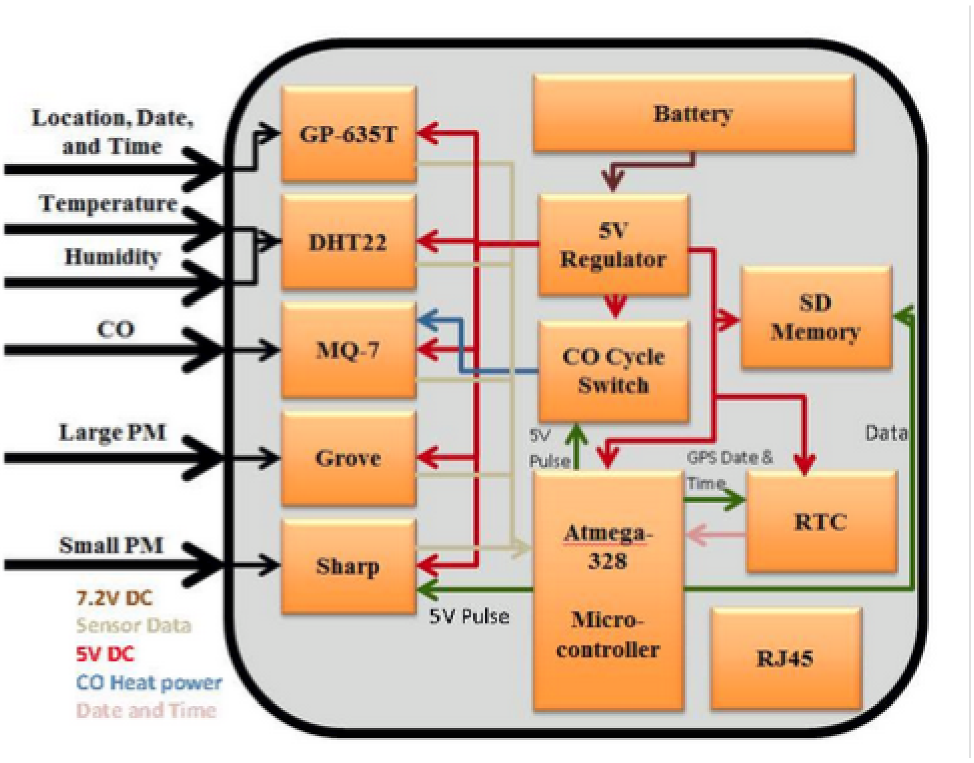


Figure 2: FInal Design (Level 1)

Figure 1 features the initial design of the system highlighting the inputs and outputs in a level 0 model. Figure 2 features the final design of the self contained system in a block diagram (level 1 model), highlighting the components, their connections, and the system inputs.

The entire system is powered via a 5V regulated signal from the battery. All sensor data is fed to a microcontroller for processing and writing to an SD card along with the time and date which is read from the RTC. The microcontroller is also used to control the CO cycle switch, Sharp sensor LED power, and setting the RTC.

In order to achieve the necessary specifications, it was evident that a low cost microcontroller would be necessary as the heart of the system. Thus, an ATmega328P 8-bit microcontroller was chosen. This microcontroller has a two-wire interface (TWI) for I2C communication, a serial peripheral interface (SPI), and USART for RX/TX serial communication making it ideal for our design. It has several other capabilities such as pulse width modulation, analog to digital conversion and external interrupts that were not used in this design. The 328P has two 8-bit and one 16-bit timer for timer interrupts. It provides 32kB of flash memory for bootloader and program storage and 2kB SRAM for data. Since sample Arduino code using the 328P is available for all sensors, the Arduino development environment was used for this project. In order to save space on the final board, rather than including a programming header it was decided that the microcontroller would be programmed on an Arduino Uno board and transferred to a socket on the fabricated PCB. The PCB required a 16MHz crystal (microcontroller clock) and filtering capacitors to support the Arduino environment.

Circuit design for the system was a fairly straightforward process. Most sensors were capable of being connected directly to the microcontroller with no additional circuitry necessary. Primary external components included a DS1307 real time clock and 23LCV1024 memory chip with battery backup for non-volatile storage (later replaced by the SD card). The memory chip provided 1024k bits (128k bytes) of byte-addressable memory. The non-volatile feature of this memory chip was not used in this project.

Since the MQ7 requires both a 5V and a 1.4V phase for proper operation, an LM317 voltage regulator is used to regulate the 5V signal to 1.4V. The output voltage of the linear regulator was controlled by a variable resistor on the feedback portion of the circuit to allow for fine tuning. Additionally, a smoothing capacitor of 0.1uF was used at the input of the regulator, and a smoothing capacitor of 1uF was used at the output.

To transition between the two power cycles of the MQ7, various SPDT relays were researched and considered. However, a more efficient method using a custom transistor circuit was designed, tested, and implemented (figure 3). The custom switch uses 2 p-channel mosfets and an npn transistor. A 5V pulse from the microcontroller (CO_PWR_SEL) controls the switch by setting the output to 1.4V when high, and 5V when low. One mosfet needed a turn on voltage below ~1.7V while the other required a turn on voltage below 5V and drain to source on resistance as low as possible so the 5V did not drop significantly across the fet.

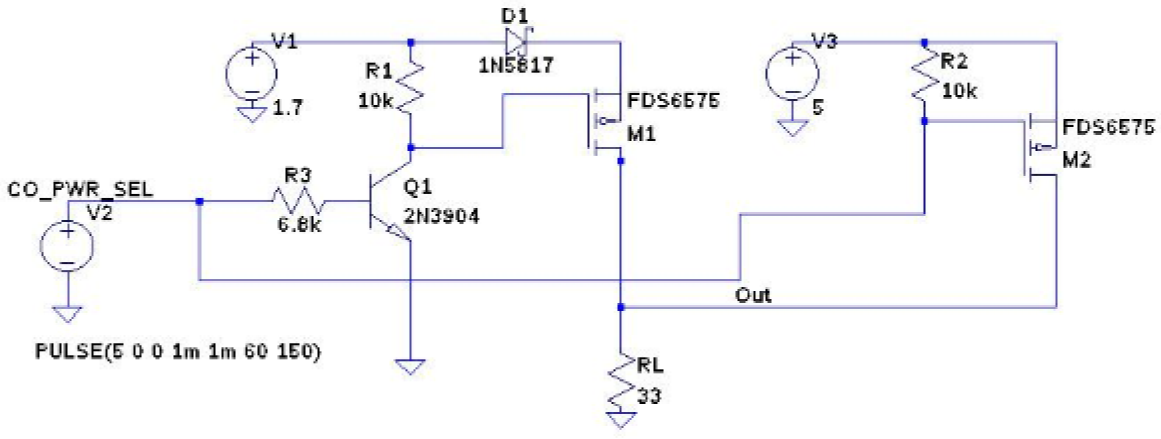


Figure 3: Transistor switch circuit

In Figure 3 resistors R1 and R2 pull the gates of M1 and M2 high holding the p-channel mosfets M1 and M2 off. RL models the resistive load of the CO sensor. The CO_PWR_SEL signal is inverted by the 2N3904 before reaching the gate of M1. The circuit will have either M1 or M2 conducting depending on the high or low level of CO_PWR_SEL. Diode D1, a Schottky diode with a forward voltage drop of approximately 0.3V, prevents the 5V source from driving current back through M1. M1 and M2 are LP0701 transistors with -16.5 volts breakdown voltage, and a -500mA drain current capacity. The turn on voltage is -0.7V and on resistance is 1.7 ohms. Switching speed was not a factor in their selection.

The relatively low number of components inspired use of through-hole parts on the board, making hand assembly and possible modifications easier. Sensors were connected to the board using screw terminals with 0.350 inch pitch.

Major components and blocks of the circuit are:

Component or circuit element	Purpose
Atmega 328P	Microcontroller
DS1307	Real time clock
CR2032	3.0V backup battery for RTC and NVSRAM.
23LCV1024	Memory expansion (nvsram).
Voltage regulator	Reduce 5V supply to ~1.7 volts
Transistor switch	Alternately provide 1.4V or 5V to CO sensor

Table 2: Circuit board components

EagleCAD was used to create the schematic and board layout. The PCBs were fabricated by OSH Park. The complete schematic and board layout can be found in the github repository.

After revision of the project specification, a GPS receiver and Arduino SD shield were added to the project. The SD board was not used as a shield, the memory chip was removed and signal lines were soldered to the SD board from where the memory chip had been.

A complete system as built would require the following components:

1. Fabricated PCB
2. Atmega 328p microprocessor flashed with program code.
3. Air quality sensors
   a. MQ-7 CO sensor.
   b. Sharp GP2Y1010AU0F particulate matter sensor.
   c. Grove (Shinyei) PPD42NS particulate matter sensor.
   d. DHT22 temperature / humidity sensor.
4. DS1307 real time clock
5. CR2032 3.0V battery and holder.
6. GP-635T GPS receiver
7. SD shield for Arduino
8. Micro Reset switch.
9. Several values of 1/4 watt resistors.
10. 100uF and 200uF electrolytic and several values of ceramic capacitors.
11. 16 MHz crystal.
12. 32.768 kHz crystal.
13. Screw terminals to attach signal and power lines from the sensors to the board.
14. 7.2V 3300mAh rechargeable battery.
15. On/Off switch.
16. Two voltage regulators, LM317 and KA78R05
17. LP0701 p-channel mosfets, two of them
18. 2N3904 npn transistor
19. RJ45 jack (for future use)

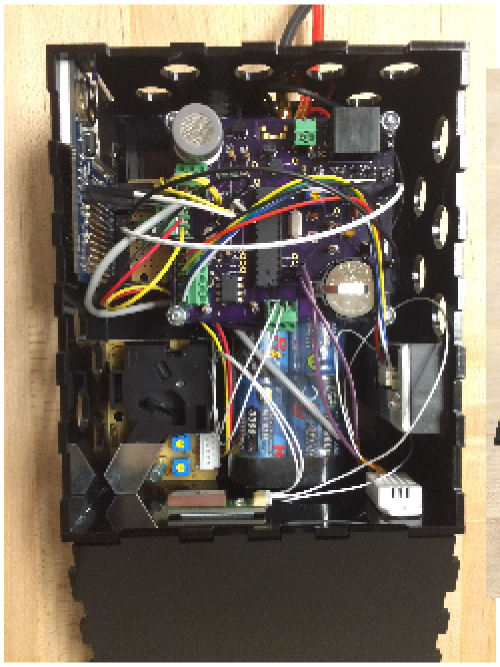


Figure 4: Complete System Internals

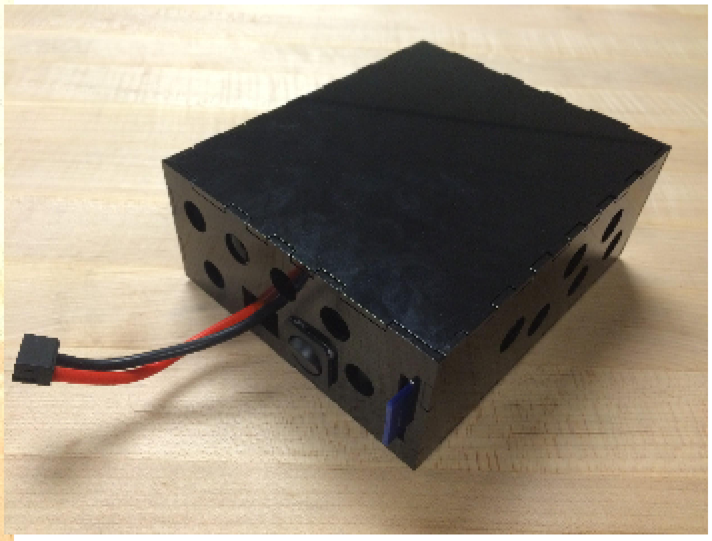


Figure 5: Complete System Box

Power

Power requirements for the Sensor Hub were calculated at 150mA – MQ7, 90mA – Grove sensor, 20mA – Sharp sensor, and 50-100mA for the microcontroller, RTC, memory chip, and other components, resulting in a total of approximately 360mA. A 500mA supply was considered a sufficient margin for reliable operation.

Originally, the Sensor Hub was to be powered using the 3.3 volt line from the OBI Smart-Lock. Due to constraints of the sensors, it was necessary to run the system at 5V. In particular, the analog outputs of the MQ7 CO sensor and Sharp PM sensor required a constant 5V reference. The MQ7 burn-off cycle also required a 150mA 5V signal. As a consequence of the Sensor Hub operating at 5V, logic-level shifters on the RX and TX lines were necessary for serial communication with the Smart-Lock, whose microcontroller was specified to operate at 3.3V.

An LT1302 switching regulator was ordered and tested. It is able to boost voltages from as low as 2 volts up to 5 volts with an efficiency of approximately 80%. Upon testing, it quickly became evident that it was not a suitable solution for our system. The maximum current output before the output voltage began to drop was 300mA, which was well below the data sheet specification of 600mA. Moreover, it was discovered that this boost converter needed a momentary start-up current of approximately 1.5A. The 3.3 volt line from the Smart-Lock however, is supplied by a TPS63031 buck converter with a maximum output current of 800mA. Thus, it was not possible to draw enough current from the Smart-Lock for the start-up phase of the LT1302. Consequently the boost converter was abandoned early in the design process and removed from the schematic before the boards were manufactured.

Various batteries were researched as alternative power sources for the system. Since the system required exactly 5V, the idea of using 5V rechargeable USB battery packs was evaluated. These solutions were especially attractive as they were extremely cost effective, lightweight, and advertised to be capable of supplying up to 10,000 mAh. Two different battery packs were tested with undesirable results. The first problem encountered was that both battery packs would shut off after approximately 30 seconds with loads drawing less than 1A. We were able to design a transistor circuit that would have momentary pulses of high current draw every few seconds to trick the battery pack into remaining powered on without wasting much battery life. The second problem encountered which led to abandoning this solution was the fact that neither battery pack was capable of supplying the advertised current of 1.0A or 2A at 5V. With a current draw of approximately 100mA the supply voltage would begin to drop below 5V. Presumably the switching regulators inside these pack are operating in constant current mode. The test was conducting using various resistors in parallel as loads and testing the voltage across them.

Finally, a 7.2 volt rechargeable Ni-Mh battery rated at 3300mAh was purchased. These batteries typically have a 1C discharge rating, meaning the battery can continuously provide up to 3.3A of current. A daughter board using a 5.0V KA278R05 linear regulator to supply the necessary voltage with 100uF smoothing capacitors at the input and output. The regulator was measured to have an efficiency (power out divided by power in) of about 66% with an input voltage of 7.2V. Using the peak power of the Sensor Hub, it should be able to run for: 3300mAh / 360mA * 0.66 = 6.05 hours. The battery is by far the largest component in the finished box.

Serial Communications and Primary Prototype

The GPS unit uses serial communication to interact with the microcontroller. Programming the microcontroller for the GPS unit using the Arduino software serial library was a straightforward process. However, establishing serial communication between the Sensor Hub and stub Smart-Lock proved to be difficult. After many hours of troubleshooting and attempting various options such as altsoftserial protocols, the problem was finally isolated to a conflict between the use of the memory expansion interface (SPI) and software emulation of serial communication together. During testing and troubleshooting, the hardware serial interface was being used to send debugging messages to a serial console in conjunction with an Arduino library for software emulation of a serial interface being used for communication with the Smart-Lock. This combination resulted in corruption of both the hardware serial interface and the data written to or read from the memory expansion chip.

Due to scheduling and approaching deadlines, getting the system to function and provide usable data became top priority. In the interest of providing a working prototype capable of storing data, serial communication was abandoned for this version of the prototype. The decision was made to move the GPS to the front box, and for the SD shield to replace the memory chip as it provides a large amount of non-volatile storage. Since the memory chip and SD shield both use the ATmega SPI interface, this transition was easy and straightforward. This transition also allowed the system firmware to be adjusted to have the GPS receiver function using hardware serial. The working prototype was completed to meet the necessary deadline with the added benefit of the unit functioning as a standalone system without being dependent on any other hardware.

Problems encountered using both the SPI interface, whether for communication or memory storage, and the software serial library still have not been resolved and are under investigation.

Programming

The Arduino IDE and C++ were used to program the microcontroller. All the sensors had sample Arduino code which allowed them to be tested individually. However, each sample code operated under the assumption that only that particular sensor was connected. Creating a merged program for all sensors required some attention as the sensors required specific timing parameters. The Grove sensor data is processed by measuring low pulse occupancy of the input signal. The sample code has a 30 second loop which does nothing but repeatedly check for the input signal being low, and timing that low pulse. Ideally the microprocessor could perform other tasks while waiting. Such a solution could potentially be implemented using a pin-change interrupt. As an alternative, our programs reduced the sample time to 1 second.

Initially, the programming was designed to follow an interrupt based routine. However, after encountering several issues with merging sensor codes into one subroutine and the complexity of sensor interrupts building and waiting, it became evident that a polling method would be a superior alternative for this system.

Before the transition to a standalone system, the program was modified to store the readings in the memory expansion chip. This was done as compactly as possible with 1 byte each for temperature and humidity, and two bytes each for the PM and CO sensors. The date and time from the RTC were also stored using 6 bytes total for month, day, two-digit year, hour, minute and second. Each record used a total of 14 bytes of memory. Had the pulse sensor not been eliminated, an additional byte could have been used to store the beats per minute data from the pulse sensor. The memory expansion chip can store 2184 15-byte records. At one record every 2 seconds this would be 72.8 minutes of data.

The serial communication code was also implemented and tested using a serial console. Command codes typed into the serial console could be used to set the time and date of the RTC, request the number of stored records, or retrieve a record. Error and acknowledge codes were implemented and work began on coding a 16-bit CRC checksum to ensure data was transmitted correctly through the RJ45 connection.

As discussed earlier, serial communication with the stand in Smart-Lock had to be abandoned. Had this not been the case, an additional, subtle problem was also encountered. The original specification called for the Sensor Hub to store 10-15 minutes of time-stamped sensor readings to which GPS data would be appended by the Smart-Lock by having the Smart-Lock read the NEMA strings from the GPS. This would require the Smart-Lock to store 10-15 minutes of GPS data and use processor cycles to match the nearest GPS time with the sensor record time.

When serial communication was eliminated by moving the GPS receiver and SD card to the Sensor Hub, work on the previous Sensor Hub code stopped and a new program called frontbox-solo was created.

Since the SD card uses the same SPI interface as the memory expansion chip, the memory chip was removed from the board and the SD signal and power lines were soldered in its place.

Prior to deciding to abandon the PulseSensor due to its unreliability it had already been integrated into the program though the operation was fairly complicated. It was implemented using timer interrupts to take an analog reading every 2 milliseconds. The readings are treated as a wave and an algorithm finds the peaks and troughs. While the results for the PulseSensor were usable when the subject was fairly still, this did not fit with the system model of measuring the heart rate of an active biker.

The frontbox-solo firmware operates successfully. Due to a full second being spent in the loop reading NEMA strings from the GPS receiver, and a full second of time spent reading the Grove sensor, the maximum frequency for sensor readings is once every two seconds. This is acceptable as the temperature / humidity sensor can only be read once every two seconds. At this rate, a bicycle traveling 15 mph (24kph) will record a reading every 44 feet (13.4m).

Having the data written to an SD card rather than into the SRAM memory chip eliminated the need to compact the data into bytes for saving space. It is now written as tab-delimited text strings to the SD card so the data is easily imported into a spreadsheet for analysis. The data is appended to the file so even if the system loses power no data is lost.

The code for the individual sensors, GPS receiver, RTC, merged sensor code and frontbox-solo can be found in the github repository at: https://github.com/alaviali/OBI

Test Plan

The following test plan outlines an integral test designed for validating the functionality of the full, assembled Sensor Hub system. For a more detailed test plan outlining each individual block of the system in detail with individual block tests, please refer to the wiki at: https://github.com/alaviali/OBI/wiki

Test Writer: Ali Alavi						
Test Case Name:		All Sensor Reading Test		**Test ID #:**		OBI_IT_#01
Description:		This integral test is a combination of all previous tests and should ensure that all sensors are functioning correctly and the data is effectively written to the SD shield.		**Type:**		White box
Tester information						
	Name of tester:			**Date:**		
Hardware Version:				**Time:**		
Setup:		Program chip, ensure the system is outside or directly next to a window.				
Step	**Action**	**Expected result**	**Pass**	**Fail**	**N/A**	**Comments**
1	Turn system on, wait 3 to 5 minutes.	The system should turn on and the red LED on the SD shield should begin blinking every 2 to 3 seconds.				

2	Light incense and place box over the system allowing smoke to build up. Wait 5 to 10 minutes, remove box, wait 2 to 3 minutes, turn system off, and check SD data.	The data on the SD card should have the correct date and time. After a satellite link was established by the GPS, it should also display the correct latitude and longitude (verifiable using mapping website). The values should have fluctuated according to the smoke buildup with PM and CO levels rising as more smoke built up.				
3	Import text file to spreadsheet program.	The spreadsheet should automatically be populated in the correct format. The top row should be the headers, while each following row should be a line of sensor records.				
4	Create a graph for Sharp values.	The graph should show a rise in output values as the box filled with smoke and a drop as the box was removed.				
5	Create a graph for Grove values.	The graph should show a rise in output values as the box filled with smoke and a drop as the box was removed.				
6	Check CO levels by creating a graph for CO read values vs. CO heat values.	The graph should show the CO Heat values as consistently higher than the CO read values. CO read values should be low until the box was placed over the system and smoke built up. All values should rise as the smoke filled the box and drop as the box was removed.				
7	Check temperature with thermometer.	The temperature should be close to the thermometer reading.				
8	Check humidity	The percent humidity should stay fairly constant throughout the experiment.				

Robustness and Durability Test
Test ID#: OBI_IT_#02

- The robustness and durability test can include several aspects. Bike riding, even on smooth tarmac can cause moderate vibrations and shocks. The OBI front box is designed to be functional under the stress of an average city bike ride. The testing procedure is to install the front box on a bike and take it to a test run. During the test run, make sure to stop every now and then to check if the battery is still powering the system, if there's still data being written to the SD card (the red LED on the SD card shield should be blinking every few seconds). After the test run, check if there are any irregularities in the record which would indicate one or more sensors have malfunctioned, or other components haven't been functioning very well.

To test the effects of having the box in motion, a fan could be used with incense burning in the box, and data could be compared with the fan on and the fan off.

Test Results

Each of the sensors was tested individually to verify they were reporting the data they were designed to detect and to get a relative range of values for this data. The PM sensors are calibrated at the factory to report their values accurately. The small particle sensor reports values in milligrams per liter and the large particle sensor reports values in particles for 0.01 cubic foot. An equation obtained by researcher Chris Nafis was used to convert the mg/l values to particles per cubic foot (Nafis 2012). He compares simultaneous data from the Sharp sensor with data with a Dylos DC1100 dust sensor in order to fit a curve and develop an equation converting the Sharp analog voltage to particles per 0.01ft^3 (the units reported by the Dylos sensor). This equation was adopted for our data. Testing shows the maximum reading from the sensor to be approximately 425000.

The CO sensor needs to be calibrated by the end user in a 100 ppm CO environment. Since we did not have the facilities or resources to perform such a calibration, the relative output voltage will be provided.

The particle and CO sensors were tested by burning incense in a closed cardboard box as outlined by the test plan. Results are shown in Figures 6 and 7. The first three minutes represent “clean” air. The box was opened and the incense extinguished midway through the test as reflected in the decrease in levels in Figure 6. Both the GPS location and the reported time for each record were found to be correct.

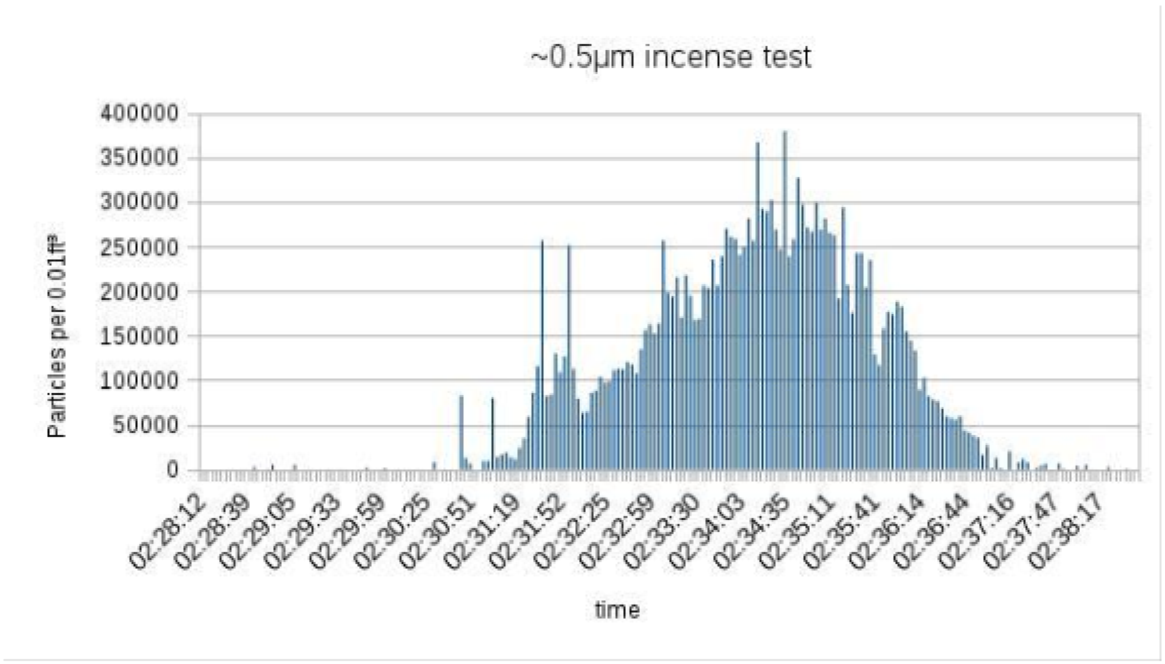


Figure 6: Sharp small particle incense test.

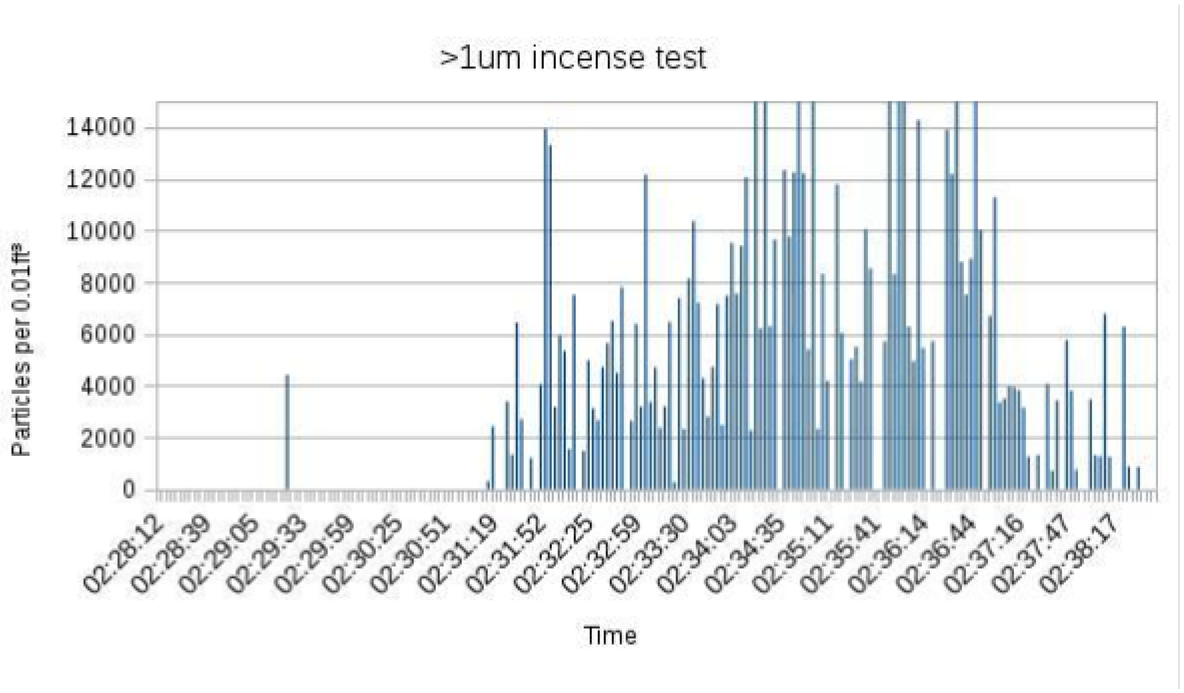


Figure 7: Grove large particle incense test.

The incense tests are for the same time period as reflected in the ordinate axis labels. There are 209 data points in the Figures. The buildup of smoke in the box is reflected more smoothly in Figure 6, which measures small, smoke sized, particles. Figure 7 shows the incense also produces larger particles, although their distribution is not as smooth. The vertical axis for the large particle sensor has been truncated at 15,000 particles per 0.01ft^3 since the sensor data sheet specifies 8000 as its maximum capability. Figure 6 shows particle concentrations upwards of 300,000 per 0.01ft^3.

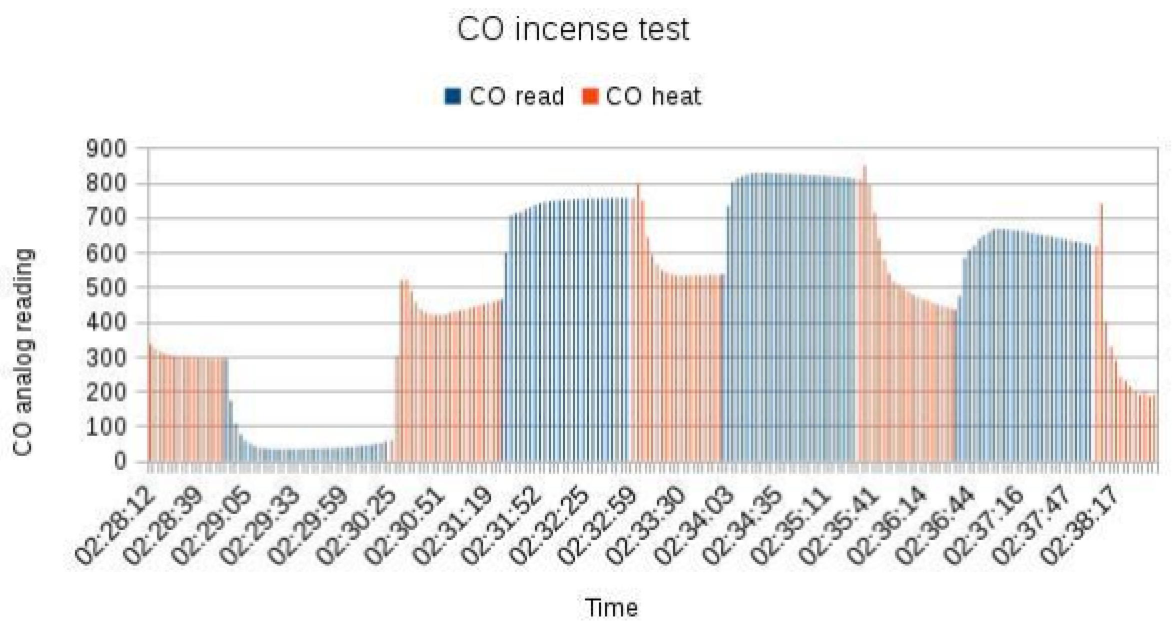


Figure 8: CO readings during incense test.

In Figure 8 CO data can be seen from the same incense test. The red lines are the analog levels during the 60 second cleaning phase while the blue lines show the levels during the 90 second valid data phase. The vertical axis shows maximum detected levels of nearly 850. This scale reflects the analog voltage on the CO read pin in values from 0-1023 mapped from voltages of 0-4.9V.

Test data from the temperature and humidity sensors are not shown but the values over dozens of lab and road tests indicate their accuracy in reflecting environmental conditions.

Durability and reliability were tested by taking the box on several test rides of 5 to 7 miles. The prototype successfully collected data and all electrical connections were maintained.

Figure 9: System mounted on the front rack of Robert's Bike

Sample Road Data

Data from a sample ride is shown in Figures 10-15.

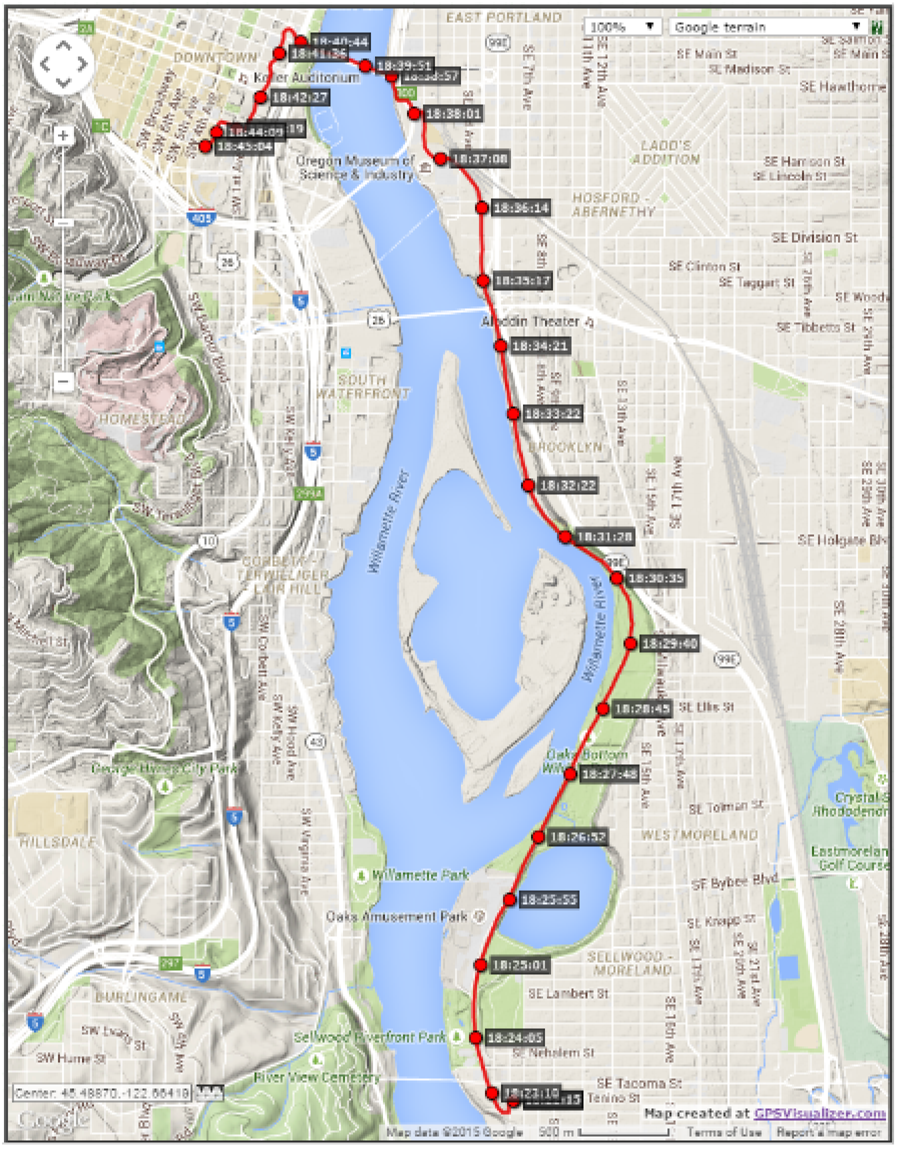


Figure 10: Map generated from GPS data.

The GPS data was used to create the map In Figure 10. The route was from south (Oaks Bottom) to north (Portland State University). Every 20th data point is marked with a timestamp, using UTC time. This ride occurred on June 9th, 2015. There are 638 total data points which were used to generate the map and following Figures.

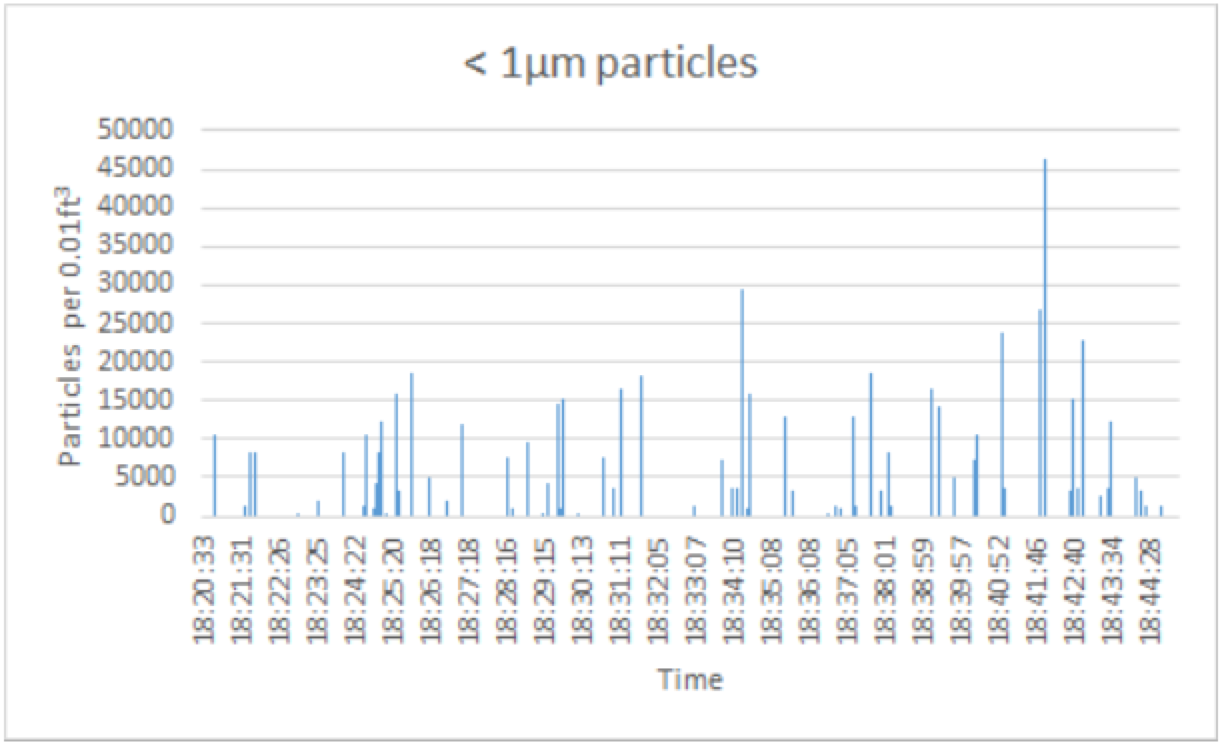


Figure 11: Data from Sharp sensor

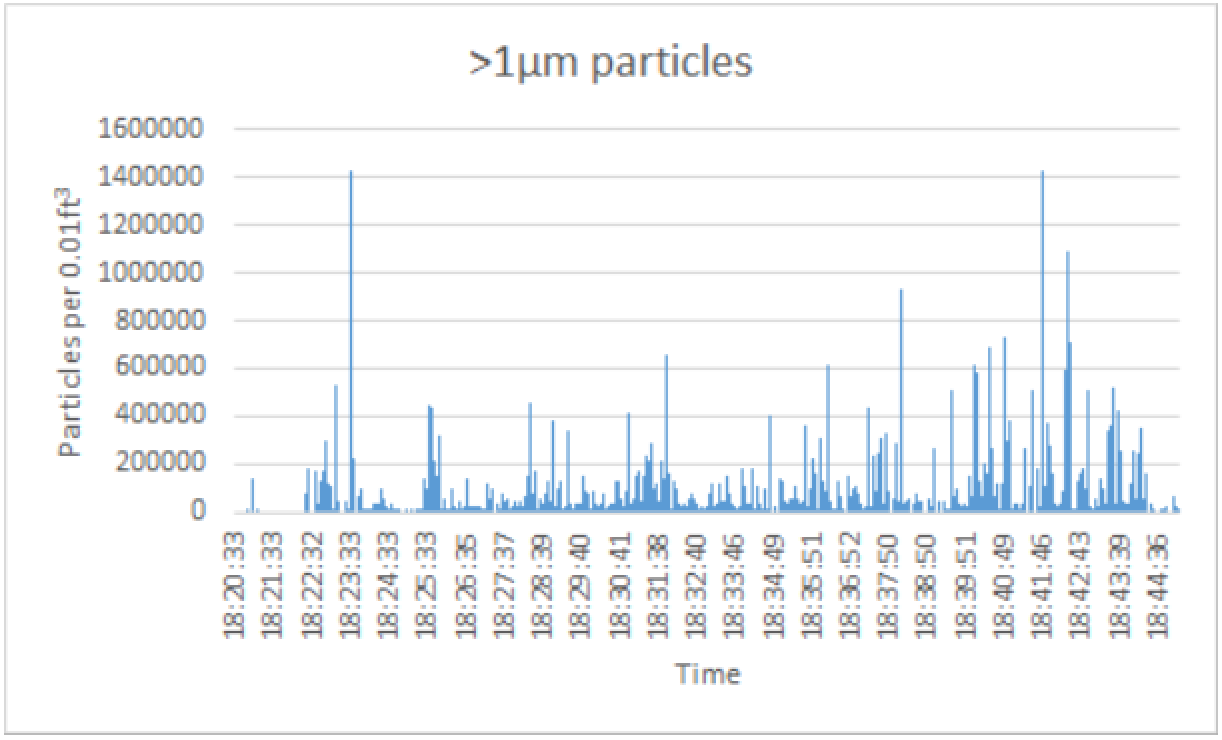


Figure 12: Data from Grove sensor.

Figures 11 and 12 show data from the PM sensors. The particle counts for small PM (<1um) and large PM (>1um) are shown. A curve draped over the data would have relatively the same shape for both charts, except for a spike at 18:23:33 in Figure 12. Both plots show more particles starting at 18:39:51 when the ride left the river and entered the city center. The particle count tapers off at the end upon arrival at the Portland State University campus.

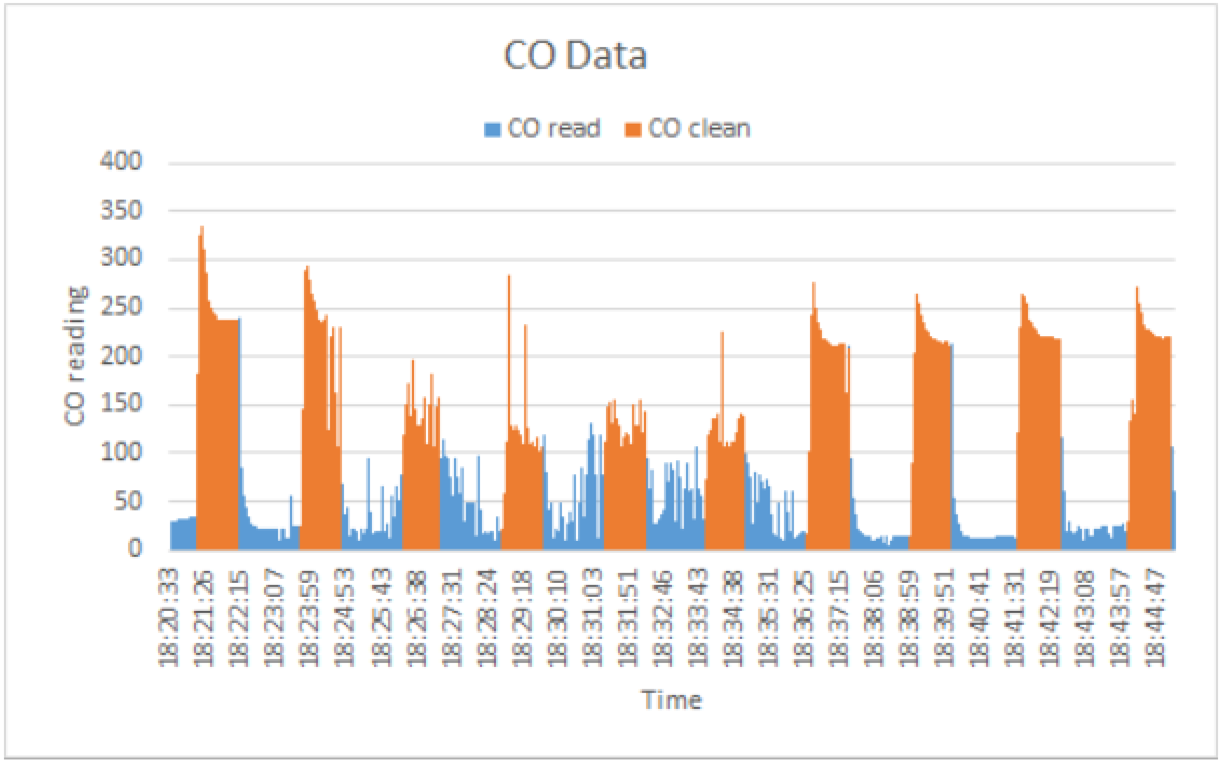


Figure 13: Data from CO sensor.

In Figure 13 we find elevated CO levels during the read phase from about 18:25 to 18:36. Referencing the timestamps on the map, this is the section of the bicycle path along the Willamette River adjacent to Ross Island. Seven data sets for this ride have been collected (some traveling north, some south, at various times of day). In many data sets the increased CO concentration is visible for that section of the path. While we can only speculate, the cause of this increase could be a result of the fact that the path is near several Ross Island Sand and Gravel Co. operations and below Highway 99E which can be high in traffic at times.

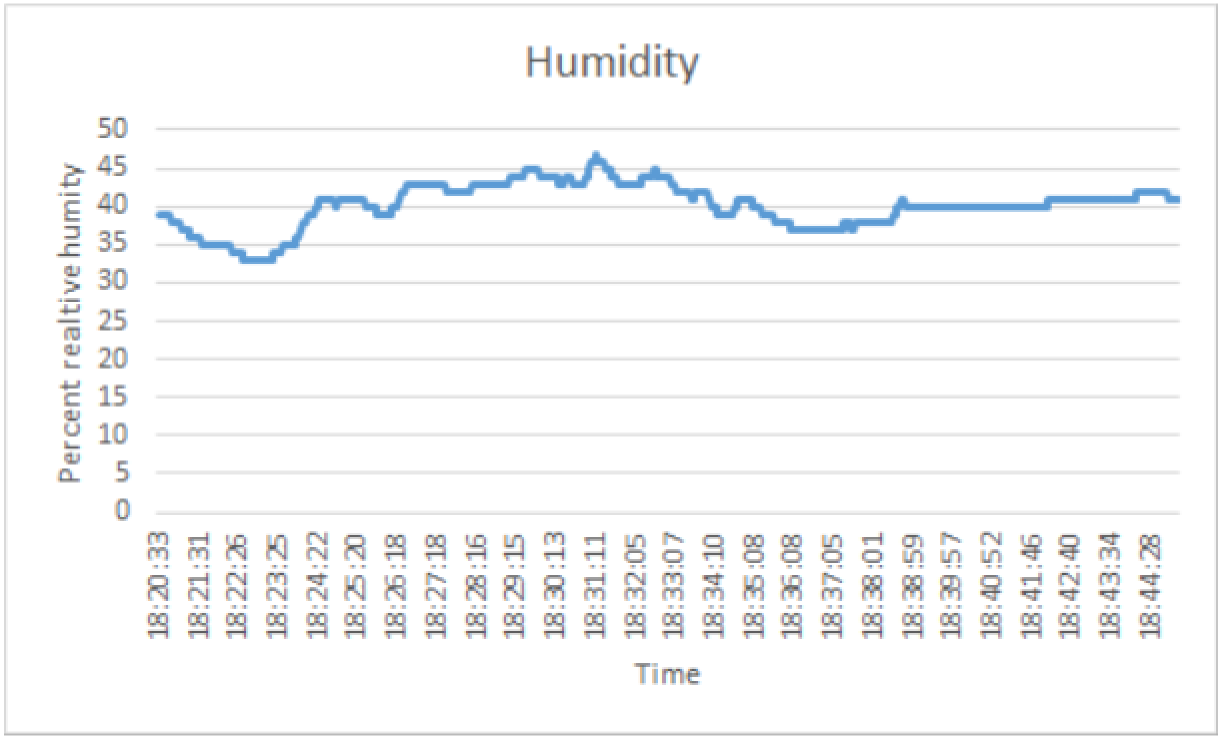


Figure 14: Humidity

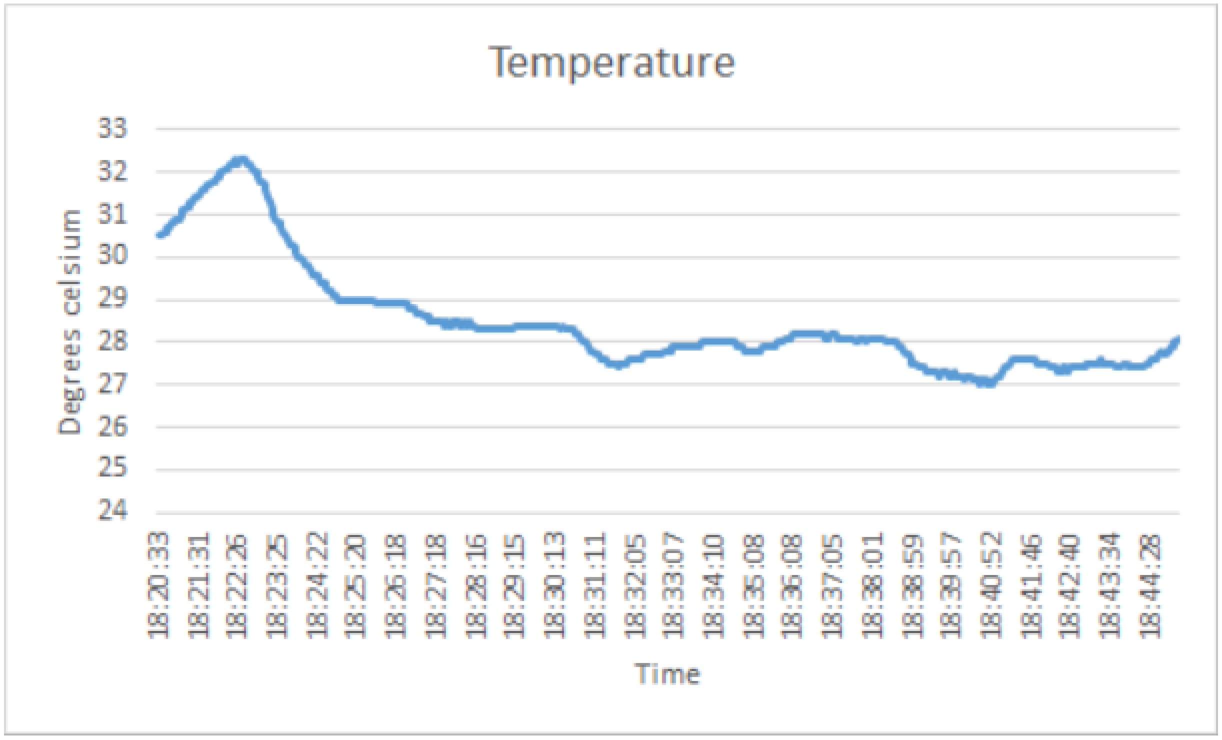


Figure 15: Temperature (celsius)

Humidity and temperature data are shown in Figures 14 and 15. This ride occurred on a warm day as indicated by the temperature. The times are in UTC and subtracting 7 hours to convert to Pacific Standard Time yields a 25 minute ride from 13:20 to 13:45. The initial high temperature likely represents residual heat from the box being in a car immediately prior to the ride.

Cost

Cost for the project is summarized in Table 3.

Original Specification		
	Total	Per set
AQ Sensors -- 4 systems	$241.55	$60.88
Components -- 4 boards	$152.87	$38.22
PCB fabrication -- 3 boards	$46.40	$15.47
Arduino Unos (2)	$49.90	N/A
Original specification total:	$490.72	$114.57
Additional costs		
GPS receivers (2)	$80.00	$40
SD shields (2)	$40.00	$20
7.2V 3300mAh NiMh batteries (2)	$40.00	$20
Grand Total:	$650.72	$194.57

Table 3: Cost summary

A complete system as built should cost about $180. Some of the ordered components were not used in the final design, thus the estimate amount of $38.22 for each set is disproportionately large.

Ethical and Legal

OBI operates under open source licensing. They use Creative Commons Licenses for documentation and the Mozilla Public Licenses for code. As they are our sponsors, our code and schematics fall under these licenses also, i.e they are available for anyone to use and modify.

Ethical considerations are also straightforward. There are no adverse effects from the project aside from incidental environmental costs associated with component and PCB manufacture. The project aims to collect air quality data which could be used to improve air quality or at least cycling routes, with consequent benefits to health. Having said that, in order to ensure the data is accurate further testing should be incorporated with professionally calibrated and verified systems.

OBI is currently in the process of determining the necessary actions to enable these devices to be used for government action without legal or ethical repercussions.

Discussion

While this project was a success in many ways and not the most technically challenging, it proved to have many other real world challenges and learning opportunities for the group. The most notable challenge was the fluidity of the scope of the project. The project initiated with a set of concrete goals that the group aimed to accomplish. As the design process progressed and the system started to take shape, the OBI team was faced with various issues regarding their Smart-Lock design leading to a large delay in their schedule. As a result, the added task of creating a stand-in system with GPS that would be capable of communicating with the sensor hub was added to the scope of the project. Attempts at this proved to have their own challenges with regards to communication and storage which ultimately led to the incorporation of storage and GPS in the sensor hub making the project completely self-contained.

Prior to that, the system was faced with power limitations which also contributed to design modifications. Upon testing components from the initial system design, it became evident that the power source (the OBI Smart-Lock) would not be capable of running this system. This led to the integration of a battery as a power source further contributing to the independent nature of the final product at a late stage in the design process. Earlier testing could have avoided this issue. Earlier testing and debugging could have also provided a potential for incorporation of the PulseSensor.

While some data has been collected, many more data sets are needed to draw meaningful conclusions. Additional data might also suggest modification to the code or changes to the data format. An ideal modification would be the calibration of the CO sensor and the potential integration of a second CO sensor that would be in detection phase when the first sensor is in burn-off phase. This would eliminate any dead zones in CO data and would provide more valuable information at the cost of increased power consumption. An alternative would be to further research CO sensors to possibly replace the MQ7.

Additional research on air quality sensors would also provide potential other pollutant sensors to be incorporated into the system. For example, a low cost sensor for nitrogen oxides would give valuable data as nitrogen oxides are precursors to ground level ozone, an important pollutant.

It is also important to note that on a bicycle, these sensors would be moving at a rather quick pace causing a disturbance in the ambient air. Since the typical application of these sensors is to be stationary, it is not known how the airflow will affect their output and results. By using professional grade sensors and conducting more tests, better results could likely be provided.

These challenges provide a series of valuable lessons, first of which is to plan ahead. This will prepare one for potential challenges that could arise and will allow for possible solutions to be formed ahead of time. Another important lesson is to test early and thoroughly. While the documentation may provide useful details, it is still important to conduct tests in advance due to the possibility of unforeseen circumstances and issues, as well as potential pitfalls and inaccuracies in the documentation. Finally, it's important to always be able to adjust as projects often change and limitation always arise.

Conclusion

The Air Quality Sensor Hub, part of the Open Bike Initiative (OBI) bike sharing model, is a low-cost, portable, and self-contained system capable of being mounted to any bicycle. It is able to detect Carbon Monoxide, Particulate Matter in two size ranges, temperature, humidity, and GPS location for storage and analysis. Future models of the OBI Smart-Lock will be able to communicate and get data from the Sensor Hub via an RJ45 connection in order to gather the data for cloud storage and open access. This will open the door for the creation of detailed maps capable of outlining pollution along paths of travel within a city relative to time and day, apps to show real-time changes in pollution, as well as many other applications which can assist bikers, pedestrians, and all kind of commuters in choosing the best routes and times for travel. This data will also be valuable for cities to reduce pollution by determining high-pollution areas on a finer scale.

These sensor hubs were designed to be able to withstand a typical city bicycle ride and tested to show a clear sensitivity to the detected pollutants. Road tests were able to outline a clear rise in carbon monoxide and particulate matter in industrial and high-traffic areas.

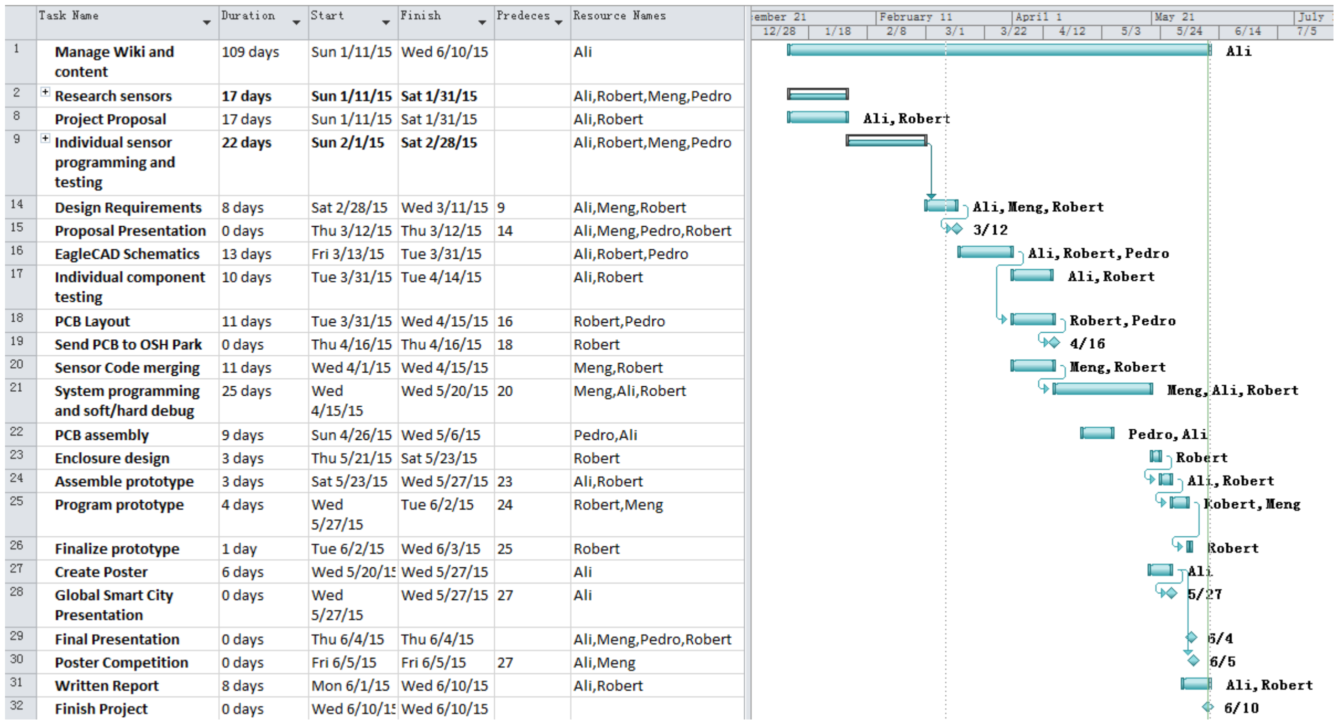
In the future, adjustments will be made to potentially reduce the size of these units, and to provide better, more accurate data with the potential for the addition of sensors for other pollutants or even the ability to monitor human physiology.

Project Management

A project schedule was created and maintained using Microsoft Project. Each member assumed the role of team leader for a one month shift. At the beginning, each member was tasked with testing a particular sensor. As the project progressed roles emerged. One member took over schematic design and board layout, another was primarily responsible for programming. The roles were fluid so everyone contributed to the project.

As previously explained, this project was a moving target which required a large amount of adjusting by the group. As a result, the timing of the schedule had to be modified constantly. Despite such conflicts, the workflow remained the same.

The final schedule as followed and each members contributions are displayed below:



	Task Name	Duration	Start	Finish	Predeces	Resource Names
1	Manage Wiki and content	109 days	Sun 1/11/15	Wed 6/10/15		Ali
2	+ Research sensors	**17 days**	**Sun 1/11/15**	**Sat 1/31/15**		Ali,Robert,Meng,Pedro
8	Project Proposal	17 days	Sun 1/11/15	Sat 1/31/15		Ali,Robert
9	+ Individual sensor programming and testing	**22 days**	**Sun 2/1/15**	**Sat 2/28/15**		Ali,Robert,Meng,Pedro
14	Design Requirements	8 days	Sat 2/28/15	Wed 3/11/15	9	Ali,Meng,Robert
15	Proposal Presentation	0 days	Thu 3/12/15	Thu 3/12/15	14	Ali,Meng,Pedro,Robert
16	EagleCAD Schematics	13 days	Fri 3/13/15	Tue 3/31/15		Ali,Robert,Pedro
17	Individual component testing	10 days	Tue 3/31/15	Tue 4/14/15		Ali,Robert
18	PCB Layout	11 days	Tue 3/31/15	Wed 4/15/15	16	Robert,Pedro
19	Send PCB to OSH Park	0 days	Thu 4/16/15	Thu 4/16/15	18	Robert
20	Sensor Code merging	11 days	Wed 4/1/15	Wed 4/15/15		Meng,Robert
21	System programming and soft/hard debug	25 days	Wed 4/15/15	Wed 5/20/15	20	Meng,Ali,Robert
22	PCB assembly	9 days	Sun 4/26/15	Wed 5/6/15		Pedro,Ali
23	Enclosure design	3 days	Thu 5/21/15	Sat 5/23/15		Robert
24	Assemble prototype	3 days	Sat 5/23/15	Wed 5/27/15	23	Ali,Robert
25	Program prototype	4 days	Wed 5/27/15	Tue 6/2/15	24	Robert,Meng
26	Finalize prototype	1 day	Tue 6/2/15	Wed 6/3/15	25	Robert
27	Create Poster	6 days	Wed 5/20/15	Wed 5/27/15		Ali
28	Global Smart City Presentation	0 days	Wed 5/27/15	Wed 5/27/15	27	Ali
29	Final Presentation	0 days	Thu 6/4/15	Thu 6/4/15		Ali,Meng,Pedro,Robert
30	Poster Competition	0 days	Fri 6/5/15	Fri 6/5/15	27	Ali,Meng
31	Written Report	8 days	Mon 6/1/15	Wed 6/10/15		Ali,Robert
32	Finish Project	0 days	Wed 6/10/15	Wed 6/10/15		

Figure 16: Project Schedule

References

Bigazzi, A. (2013). Portland ACE. Retrieved 2015.

Katsouyanni, K. (2003). Ambient air pollution and health. British Medical Bulletin, 68(1), 143-156. doi:10.1093/bmb/ldg028

Nafis, C. (2012). Automatically measuring and graphing Air Quality with an inexpensive device. Retrieved April 26, 2015.

Open Bike Initiative: http://openbikeinitiative.org/