Nobi
Wildfire Prevention Smart Sensor
Final Report

Team 1
Yil Verdeja, ECE Box 347
Yeggi Lee, ECE Box 181
Shannon McCormack, ECE Box 607
Lab Section B01
Dec 15, 2017

TABLE OF CONTENTS

1 Introduction **6**
- 1.1 Problem Statement 8
- 1.2 Market Research 8

2 Product Requirements **10**
- 2.1 Customer Requirements 10
  - 4.1.1 Explicit Requirements 10
  - 4.1.2 Implicit Requirement 11
- 2.2 Product Specifications 11
- 2.3 Competitive Value Analysis 12
  - 2.3.1 Customer's Essential Criteria and Weights 13
  - 2.3.2 Vigily's FireALERT MK 1 15
  - 2.3.3 Insight Robotics' Wildfire Detection System 16
  - 2.3.4 Libelium's Fire Detecting System 18

3 Design Approach **19**
- 3.1 System Architecture 20
  - 3.1.1 Energy Harvesting and Battery Charging System 21
  - 3.1.2 Bluetooth and Environment Sensing System 21
- 3.2 Module Description & Analysis 23
  - 3.2.1 Solar Photovoltaic Module 23
    - 3.2.1.1 Input Signal Characteristics and Response 24
    - 3.2.1.2 Functional Description 25
    - 3.2.1.3 Output Signal Characteristics 25
  - 3.2.2 Voltage Regulator 25
    - 3.2.2.1 Input Signal Characteristics and Response 26
    - 3.2.2.2 Output Signal Characteristics 26
  - 3.2.3 Charge Controller 27
    - 3.2.3.1 Input Signal Characteristics and Response 28
    - 3.2.3.2 Output Signal Characteristics 28
  - 3.2.4 Battery 29
    - 3.2.4.1 Module Inputs 30
    - 3.2.4.2 Output Signal Characteristics 30
  - 3.2.5 RH, Temperature, Gas & Pressure Sensor 30
    - 3.2.5.1 Input Characteristics 31
    - 3.2.5.2 Functional Description 34
    - 3.2.5.3 Output Signal Characteristics 34

3.2.6 Microcontroller and Bluetooth Module 35
3.2.6.1 Input and Output Characteristics 35
3.2.6.3 Functional Description 38
3.3 Defining Product Changes and Additions 40
3.3.1 Software Modifications 40
3.3.2 Installation 43
3.3.3 Data Visualization 45

4 Product Results 47
4.1 Product Functionality 47
4.2 System Testing and Results 47
4.2.1 Sensor Testing 47
4.2.2 Calibration and State Testing 48

5 Business Analysis 48
5.1 Initial Investment 49
5.2 Return on Investment 50

6 Failure and Hazard Analysis 50
6.1 Internal Failures and Hazards 51
6.2 External Failures and Hazards 51

7 Recommendations 51

8 Conclusion 52

Appendix A: Computer Code 53
Nobi Code 53
BluefruitConfig.h 61
floatToString.h 63

References 66

LIST OF FIGURES

Figure 1	Energy Harvesting and Battery Charging System Block Diagram	6
Figure 2	Bluetooth and Environmental Sensing System Block Diagram	7
Figure 3	BLE and Sensor System Flow Diagram	8
Figure 4	Final Block Diagram of the Nobi	9
Figure 5	Schematic of the Nobi Prototype	9
Figure 6	Solar Panel (9 V 220 mA)	10
Figure 7	Tilt Formulas	11
Figure 8	Solar Cell I-V Characteristic	12
Figure 9	7805 TO-220 Voltage Regulator Diagram	12
Figure 10	Pin Assignment of CN 3083	15
Figure 11	Lithium Battery 3.7 V, 1 Ah	17
Figure 12	BME680 Integrated Environmental Unit Bosch Sensortec	18
Figure 13	Adafruit BME680 Breakout Board	19
Figure 14	Forced Mode Operation in the BME680	22
Figure 15	Adafruit Feather 32u4 Bluefruit LE	23
Figure 16	Active Supply Current vs Frequency Graph	25
Figure 17	Idle Supply Current vs Frequency Graph	25
Figure 18	Flow Diagram of Software Modules	27
Figure 19	Lithium Battery 3.7 V, 1 Ah	28
Figure 20	BME680 Integrated Environmental Unit Bosch Sensortec	30
Figure 21	Adafruit BME680 Breakout Board	30
Figure 22	Forced Mode operation in the BME680 with TPHG cycle	33
Figure 23	Adafruit Feather 32u4 Bluefruit LE	34

Figure 24 Active Supply Current vs Frequency Graph 36

Figure 25 Idle Supply Current vs Frequency Graph 36

Figure 26 Flow Diagram of Software Modules 38

Figure 27 Alarm Interrupt of the Nobi Flow Chart 39

Figure 28 Calibration Flow Chart 40

Figure 29 Updated Overall Nobi Flow Chart 41

Figure 30 Updated Bluetooth Mesh Network Interrupt Flowchart 42

Figure 31 Heavy Duty Custom Bolt 43

Figure 32 Screenshot of UART Bluefruit LE App Receiving Data from the Nobi 44

Figure 33 Oracle Data Visualization 45

Figure 34 Break-even Analysis Graph 49

LIST OF TABLES

Table 1 Competitive Value Analysis Legend 13

Table 2 Value Analysis of FireALERT MK I 15

Table 3 Value Analysis of Insight Robotics' Wildfire Detection System 16

Table 4 Value Analysis of Libelium Fire Detection 18

Table 5 Electrical Characteristics of 7850 26

Table 6 Electrical Characteristics of the CN3083 28

Table 7 Gas Sensor Specifications 32

Table 8 Power supply current of the ATmega32u4 MCU 35

Table 9 Input Voltage of the Bluetooth Module 37

Table 10 Unit Cost Break-down 47

Table 11 Unit Cost Break-down 48

1 Introduction

Wildfires in the United States have always been devastating. However, with the scorching temperatures of today, they are becoming alarmingly more frequent and often times longer. More and more people are being displaced while thousands, if not billions, of dollars are being used to combat this problem. In fact, the Forest Service is using more than 50% of its budget to suppress wildfires.[1]

Due to climate change, moisture and precipitation levels are changing, leaving wet areas wetter and dry areas drier. The high temperatures has caused snow to melt earlier leading to a longer summer and drier soil. As a result, drought conditions ensue causing trees and undergrowth to become essentially kindling. With the winds coming from the northeast, this provides the perfect environment for wildfires to develop from a single spark whether it is from a cigarette or a strike of lightning. The smallest fires transform into deadly, fast-moving infernos burning everything in their path. [2]

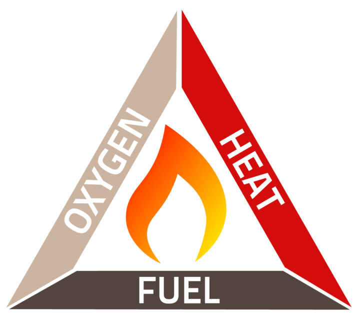

Figure 1. Fire Triangle Fundamentals[3]

No state knows this better than California who has suffered a devastating loss of resources, people, and infrastructure even before the recent December fires. At least 40 people have been confirmed dead though the number is expected to rise as officers explore the ravaged areas. With a staggering 5,700 structures lost, almost a 100,000 people have been forced to evacuate, leaving their belongings and homes behind. An expected $65 billion in property damage has left

[1] (n.d.). The Rising Cost of Fire Operations - Forest Service. Retrieved October 31, 2017, from https://www.fs.fed.us/sites/default/files/2015-Fire-Budget-Report.pdf
[2] (2016, October 10). Climate Change Blamed for Half of Increased Forest Fire Danger - The .... Retrieved October 31, 2017, from https://www.nytimes.com/2016/10/11/science/climate-change-forest-fires.html
[3] (n.d.). Back to Basics with the Fire Triangle - Elite Fire Protection. Retrieved October 31, 2017, from http://www.elitefire.co.uk/basics-fire-triangle/

California in shambles.[4] The recent wildfire outbreak is the second deadliest in this century with the Oakland hills fire in 1991 taking first place. [5]

Figure 2. Map of the locations of wildfires in California [6]

The costs of wildfires on human life, property damage, and federal dollars is crippling and increasing exponentially. More and more acres are being burned each year leading to more budget being spent on wildfires. Even more so, wildfires cause secondary effects and disasters such as landslides, flash floods, and smoky air condition that affect people's health and aircraft visibility. Shown in the graph below is the projected growth on fire suppression spending. Wildfires will continue to spread and grow until there are no more trees to sustain them.

---

[4] (2017, October 11). Wildfire property damage could reach $65 billion in Northern California. Retrieved October 31, 2017, from http://money.cnn.com/2017/10/11/news/economy/california-wildfire-cost-estimate/index.html
[5] (2017, October 11). California's Wildfires - The New York Times. Retrieved October 31, 2017, from https://www.nytimes.com/2017/10/11/us/california-fires-questions.html
[6] (n.d.). California fires: 36 dead as 'ashes and bones' found in ruins - CNN. Retrieved October 31, 2017, from http://www.cnn.com/2017/10/13/us/california-fires-updates/index.html

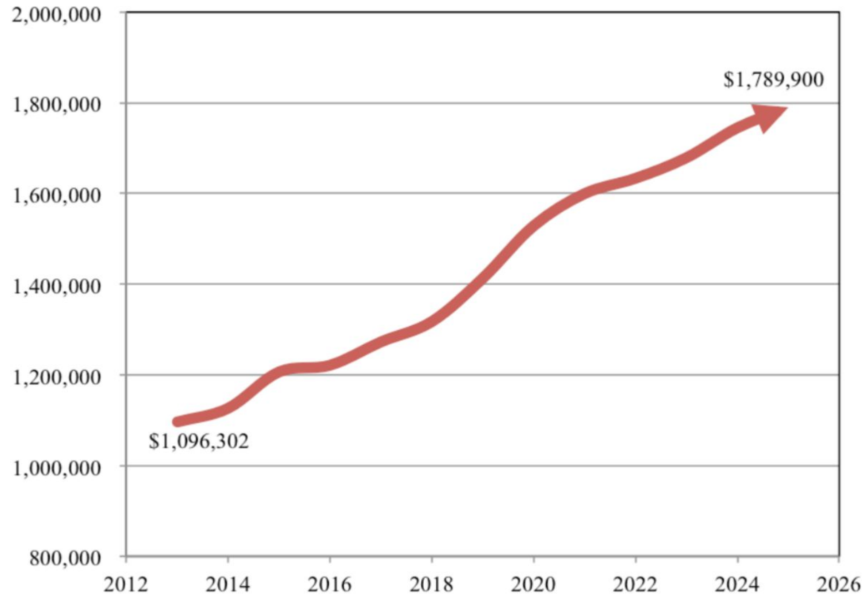

Figure 3. Projected Growth of the 10-Year Average Cost of Fire Suppression (in $1000s) Through 2025[7]9

With such high risks, the federal government is trying to prevent and fight the outbreak of forest fires. Many departments such as the Forest Service, National Park Service and Environmental Protection Agency are devising ways to contain the fire using new methods and technology. They are also turning to the public using competitions such as the one that the Environmental Protection Agency is sponsoring right now. Additionally, NASA also designed and developed a drone that would be able to detect fires through cameras and sensors.[8]

1.1 Problem Statement

With all the recent issues and costs that arise from wildfires, the government is looking for a solution to prevent and mitigate wildfires. Although there has been solutions in the past, so far none come at a reasonable price. The market needs a fire detection device that functions accordingly, not necessarily an extravagant product. Therefore our purpose is to find an inexpensive alternative to fight the ongoing battle against wildfires.

1.2 Market Research

After doing the initial research, we also looked into other products and what seemed to work or not work. According to Professor A. Enis Cetin, chief technology officer of Wildland Detection

[7] (n.d.). The Rising Cost of Fire Operations - Forest Service. Retrieved October 31, 2017, from https://www.fs.fed.us/sites/default/files/2015-Fire-Budget-Report.pdf
[8] (2016, October 10). Climate Change Blamed for Half of Increased Forest Fire Danger - The .... Retrieved October 31, 2017, from https://www.nytimes.com/2016/10/11/science/climate-change-forest-fires.html

Systems, the main automated fire-detecting devices are ground-based visual systems, ground-based non-visual sensors, manned and unmanned aircraft, and satellites.

A popular product now is the Attentis[9] R8 Series which provides an unmanned solar-powered 24-hour surveillance across wide ranges in high risk areas. This product has sensors that provide climate information, air quality, and tracks any occurring fires. Once a fire is detected, the device contacts the responders using wi-fi. They provide an interactive map of the wildlands in real time.

Another product being developed is the Vigilys'[10] FireALERT MK I which is also a self-contained, early warning wildfire detection system. Unlike the other competitor, this one contains a video camera which scans through a full 360 degree horizontal view. Any fire-related signs are analyzed and determined if safe. Their energy is stored into a super capacitor and requires almost no maintenance for around 20 years. However, there is no more information about this product after 2014 despite claims of having a second version to be released in 2015.

Figure 4. Vigilys Product MK1

Ultimately, after much consideration, we decided that our product will be ground-based non-visual sensor as it provides a cheap solution to the issue at hand. The main advantage will be

[9] (n.d.). Attentis Technology - Advanced sensors, Intelligent wireless networks .... Retrieved October 31, 2017, from https://attentistechnology.com/
[10] (n.d.). Wildfire Detection - Vigilys. Retrieved October 31, 2017, from http://vigilys.com/technology/firealert/

its to ability to sense heat and fire despite smoke, rain, and fog which might hinder visual based devices. We definitely wanted to emulate certain characteristics of both products mentioned above. Our device will also be self-contained, fire-proof, tamper-proof, and have a fairly decent range. Later on, we might implement a camera along with the sensors to combine both of best worlds and provide accurate tracking of the wildfire movement.

Furthermore, we know that our product will be in demand as the federal government is looking for ways to reduce costs regarding wildfires. As mentioned earlier, they are holding a competition right now and reaching out to the public to design wildfire alert devices. According to the challenge, the Environmental Protection Agency is looking for a prototype system that "should be accurate, light-weight, easy to operate, and capable of wireless data transmission, so that first responders and nearby communities have access to timely information about local air quality conditions during wildland fire events." They are offering a reward of $60,000 to the winner as an incentive to people worldwide.

The results of our research led to the conclusion that our market will mainly be for the government. However, our product may also benefit residents near wildlands and woodlands. There is enough evidence to prove that our product will do fairly well as a very cheap alternative to some of the more expensive products in the market.

2 Product Requirements

2.1 Customer Requirements

According to the market research, the government and private companies are looking for a device that is both accurate, light-weight, easy to operate, and capable of wireless data transmission. In fact, currently the Environmental Protection Agency is holding a competition to design and develop such a device[11]. As such, the customer requirements are explicitly and implicitly stated below.

According to the market research, the government is looking for the following criteria in a device:

<u>4.1.1 Explicit Requirements</u>

The requirements that need to be met for the customer are as follow:

- ❏ Accurate
- ❏ Easy to operate
- ❏ Easy to attach

[11] (n.d.). Wildland Fire Sensors Challenge | Challenge.gov. Retrieved October 31, 2017, from https://www.challenge.gov/challenge/wildland-fire-sensors-challenge/

- ❏ Cheap & Affordable
- ❏ Light Weight
- ❏ Durable
- ❏ Capable of wireless data transmission
- ❏ Capable of harvesting energy
- ❏ Capable of sensing fire conditions
- ❏ Low Maintenance
- ❏ Weatherproof

4.1.2 Implicit Requirement

Other implicit requirements that are expected are listed below:

- ❏ Tamper Proof
    - ❏ Humans and Animals
- ❏ Long range communication
- ❏ Low power consumption

2.2 Product Specifications

From the customer requirements, we derived the following requirements necessary for our Wildfire sensing product:

- ❏ Durable
    - ❏ Weather/Dust/Tamper/Fire proof
- ❏ Low Maintenance
    - ❏ Harvest Energy
- ❏ Wireless Communication
- ❏ Cheap and Affordable
- ❏ Accurate Sensing in Detecting Fire Conditions
- ❏ Low Power Consumption

A main aspect for the product is that it should be durable. That means that it should be protected from rigorous outdoor conditions and it should have a long life span. Our product will be placed outdoors in areas that are most susceptible to wildfires such as forests or vineyards. Since the product will be placed outdoor, it will be met with harsh conditions that might damage it such as the weather, dust, human/animal tampering and lastly, fires. This requirement connects to the low maintenance requirement because as long as it is durable, it will not need as much maintenance. Ideally, the product should be made to last 5-10 years and it should provide efficient functionality, especially in power consumption.

Another way to lessen the products maintainability is by including means of energy harvesting. By either harvesting non-stray energy or by not harvesting energy at all, a person is required to be near the presence of the product in order to keep it working. This is not ideal considering the various remote outdoor locations the product will be placed in. To maintain each and every unit would be a waste of time, and an inconvenience that can be avoided. Energy will be harvested and collected through environmental means in order to keep the device running autonomously.

To decrease maintenance, a wireless communication feature is ideal because it is more mobile, convenient, and flexible at a lower cost. First, it gives a much greater deal of mobility as radio waves are able to travel freely through the air. It is also convenient and flexible, because it gives the ability for multiple devices to be connected to one another at a certain range in a large mesh network. For our product, this is preferable because a network of sensing devices can provide much better analysis on the environmental conditions in a specific area or zone. Lastly, by eliminating wires and cables, it greatly reduces the cost of the system.[12]

One definite requirement is that the product must be cheap and affordable. Since the product is expected to cover forests (i.e. large areas of land), multiple devices would need to be obtained in order for more accurate information on sensing fire conditions in specific areas. Having to buy multiple units can come at a large cost. However, by keeping costs per unit low and affordable, not only will the product be attractive to government agencies, but will also strive in the market.

To collect information about conditions that may cause fires in forests, accurate sensing in detecting fire conditions will be essential for a good product. However, to keep power consumption and costs low, the number of sensors should be minimized while still providing precise sensing in detecting fire conditions. This type of product should be utilized to help prevent and/or minimize forest fires as it will notify the customer what areas are prone to ignite due to natural occurrences. Therefore, minimally, the customer needs just enough information to understand what is happening in the area the product is placed – which can be done by reducing the amount of sensors while still providing accurate data.

2.3 Competitive Value Analysis

In our research, we chose three competitors with a similar goals. This included Vigily's FireALERT MK 1, Insight Robotics' Wildfire Detection System, and Libelium's Fire Detection System. All of these products aimed to help fight the outbreak of wildfires, but with slightly different approaches. Insight Robotics' product, for example, did not aim for a self-sustaining device and therefore requires a power source to operate. One thing all these products have in common was the pricing; each company has a starting point of over $700 dollars per unit. Thus causing our product to stand out from the competition despite the difference in quality.

[12] (n.d.). What Are The Advantages of Wireless Communication? | It Still Works .... Retrieved October 31, 2017, from https://itstillworks.com/advantages-wireless-communication-1121.html

2.3.1 Customer's Essential Criteria and Weights

To further evaluate our competition, the customer's essential criteria was divided into quality, convenience, and cost. In the quality category, performance and accuracy will be judged. The convenience one will include durability, ease of use, and size. Finally the cost will also be analyzed.

We rated the categories with a scale of 0-3. Attached below are what each scale means in terms of each category and their weight value.

Table 1. Competitive Value Analysis Legend

Category	Description	0	1	2	3	Weight
QUALITY						
Performance	How well does it operate consistently	Not at all	Poorly	Moderately	Well	**90**
Accuracy	How accurate are the measurements of the sensors	Not at all	Poorly	Moderately	Well	**90**
Range	The operating range of the device	<10m	<100m	<1000m	>1000m	**85**
CONVENIENCE						
Durable	Weather-proofness and low-maintenance levels	Not at all	Poorly	Moderately	Well	**85**
Ease of use	User Interface and installing difficulty	Extremely Difficult	Difficult	Moderate	Simple	**80**
Size	Dimensions of the device	$<100m^2$	$<10m^2$	$<0.50\ m^2$	$<0.20m^2$	**55**
Self-sustainable	Ability to sustain itself without outside power source	Requires an outside power source	Self-sustainable for a short time	Self-sustainable but requires charging	Harvests enough energy to sustain itself	**90**
COST						
Price	Cost for product	$1000+	<$500	<$100	<$50	**95**

For the value analysis, each criterion was weighed depending on the importance to the customer. The most important one to us was the price of the device. In order to make the device as marketable as possible, it must be affordable and cheap to appeal to a large customer pool. It is important to note that most of the costs were not available for the competitors' products. Therefore, we estimated the cost using their respective product sheets. The second highest criterias were performance, accuracy and self-sustainability. The device must consistently work as accurately as possible to identify possible fires. With the quick rate of wildfires expanding, any mistake has deadly consequences. Also, to make the product as convenient and maintenance-less as possible, it must be harvest energy in any way to sustain itself.

Next up are durability and range. The product should have a long lifespan and require little to no maintenance. The forests are spread-out and mostly away from civilization meaning that it there will not always be people around to repair. Also, a wider range is ideal to prevent the need for more devices to be installed and used. After these is ease of use. It is important that our device is easy to install and sends understandable and necessary data. If our data is confusing, it wastes critical time.

Finally the last criterion was size. A device that is too big might cause disturbances and invite animals to mess around with it. Furthermore, having a device the size of a handheld would allow for easier transporting and installing.

2.3.2 Vigily's FireALERT MK 1

The Vigilys' FireALERT MK 1 was one of the wildfire detecting devices offered in the market. Considering its purpose is very similar to ours and they have the same target clientele, we consider it a competitor. This particular sensory system has provided over 8 years of operational performance due to its tracking and reporting on a fire's progression in real life. Like ours it is solar powered and self-contained.

Figure 5. Specs of FireALERT MK I

The FireALERT MK I's[13] considerable specifications are nothing to sneeze at. Regarding its quality, it is highly accurate with IR detectors and 360 degree view scan. Furthermore, it

[13] http://vigilys.com/technology/firealert/

analyzes any potential signatures related to fire using a specific spectral pattern within a mile radius. Once detected, it immediately sends an alert via direct satellite communication giving it GPS. Convenience-wise, it is very durable with a 20 year lifespan and weather-proof casing. Additionally, it is fairly easy to use as the device is automated and sends alerts only if a fire has occurred and its location. The size if moderate with a pole-mount design. Finally, cost-wise, this product is most likely very expensive especially with its state of the arts sensors, scanners and solar arrays.

Table 2. Value Analysis of FireALERT MK I

Competitor		**FireALERT MK I**	
QUALITY	**Weight**	**Rating**	**Score**
Performance	90	3	270
Accuracy	90	3	270
Range	90	3	270
CONVENIENCE			
Durable	80	3	240
Ease of Use	85	2	170
Size	55	1	55
Self-sustainable	90	3	270
COST			
Cost	95	1	95
TOTAL		19	**1640**

2.3.3 Insight Robotics' Wildfire Detection System

Figure 6. Insight Robotics' Wildfire Detection System[14]

[14] https://iffmag.mdmpublishing.com/tag/insight-robotics/

Insight Robotics' Wildfire Detection System has received awards from CISIS, Hong Kong ICT Awards, IBM SmartCamp, and Fast Company[15] for this fire detector's accuracy and durability. The features of this product include (but are not limited to):

- ❏ 360° 24/7 scan
- ❏ Thermal imaging sensors
- ❏ Location-based data transmitter
- ❏ Pan-tilt cameras
- ❏ Analyzes historical fire data to identify risk areas and plan fire mitigation strategies.
- ❏ User friendly interface, compatible with pc, tablet, smart phone
- ❏ Ability to spot a fire as small as a single 2m x 1m tree within a 5km radius

These features attracted us, but we were unable to implement all of these into our own design due to having a much lower budget.

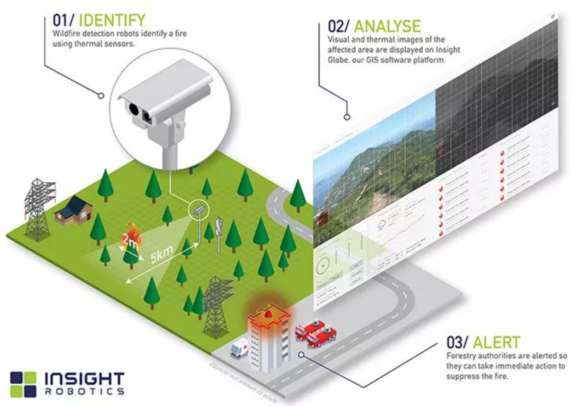

Figure 7. Insight Robotics' Wildfire Detection System Design[16]

Although the features may be superior to our own, their design was headed in a different direction than Nobi's. We aimed for a smaller more cost efficient device, whereas Insight Robotics went big, in all aspects.

Table 3. Value Analysis of Insight Robotics' Wildfire Detection System

Competitor		**Insight Robotics' Wildfire Detection System**	
QUALITY	**Weight**	**Rating**	**Score**

[15] https://www.insightrobotics.com/services/wildfire-detection-system/
[16] https://iffmag.mdmpublishing.com/insight-robotics-wildfire-detection-system/

Performance	90	3	270
Accuracy	90	3	270
Range	90	3	270
CONVENIENCE			
Durable	80	3	240
Ease of Use	85	2	170
Size	55	1	55
Self-Sustainability	90	0	0
COST			
Cost	95	0	0
TOTAL		15	**1275**

2.3.4 Libelium's Fire Detecting System

Libelium, a smart technology company based in Spain, teamed up with the SISVIA Vigilancia y Seguimiento Ambiental to develop a solar-powered fire detection system which consists of three main parts: Wireless Sensor Network, Communications Network, and Reception Center. [17]Their system design is shown in the figure below.

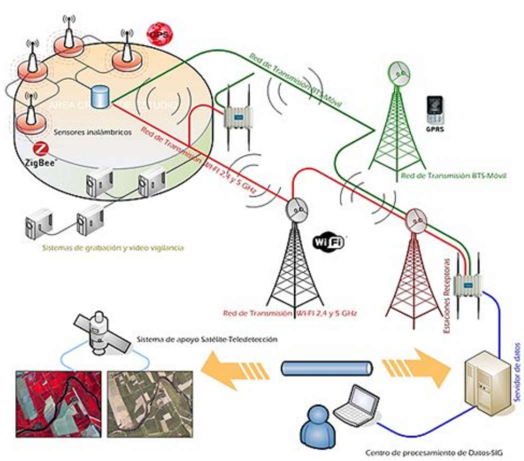

Figure 8. System Design of Libelium Fire Detection

This system works using waspmotes, open source wireless sensor platforms with a focus on low consumption modes, and meshliums, a multiprotocol router. Each of these waspmotes contain sensors that measure temperature, relative humidity, carbon monoxide, and carbon dioxide. According to Libelium, the waspmotes send information every five minutes and when not in use enters a low-power mode. Also, its range is an impressive 1.5 km.

[17] http://www.libelium.com/wireless_sensor_networks_to_detec_forest_fires/

The data can be transmitted via Wi-fi, GPS, Bluetooth, Zigbee, GPRS and Ethernet, making this competitor stand out due to its multiple means of communication devices. The meshlium is capable of receiving data via Zigbee from the sensors and sending it to a control center via Wi-fi. Additionally, the product is very sturdy and durable meaning that it is able provide connection in harsh environments.

Table 4. Value Analysis of Libelium Fire Detection

Competitor		**Libelium Fire Detection**	
QUALITY	**Weight**	**Rating**	**Score**
Performance	90	3	270
Accuracy	90	3	270
Range	90	3	270
CONVENIENCE			
Durable	80	3	240
Ease of Use	85	2	170
Size	55	3	165
Self-Sustainability	90	3	270
COST			
Cost	95	0	0
TOTAL		20	**1,655**

3 Design Approach

The two systems described below make up the Nobi. This can clearly be seen through the block diagram and the schematic below. The battery subsystem is necessary to make the MCU subsystem function; however, the MCU subsystem gives the battery subsystem a purpose. Clearly this is a symbiotic relationship.

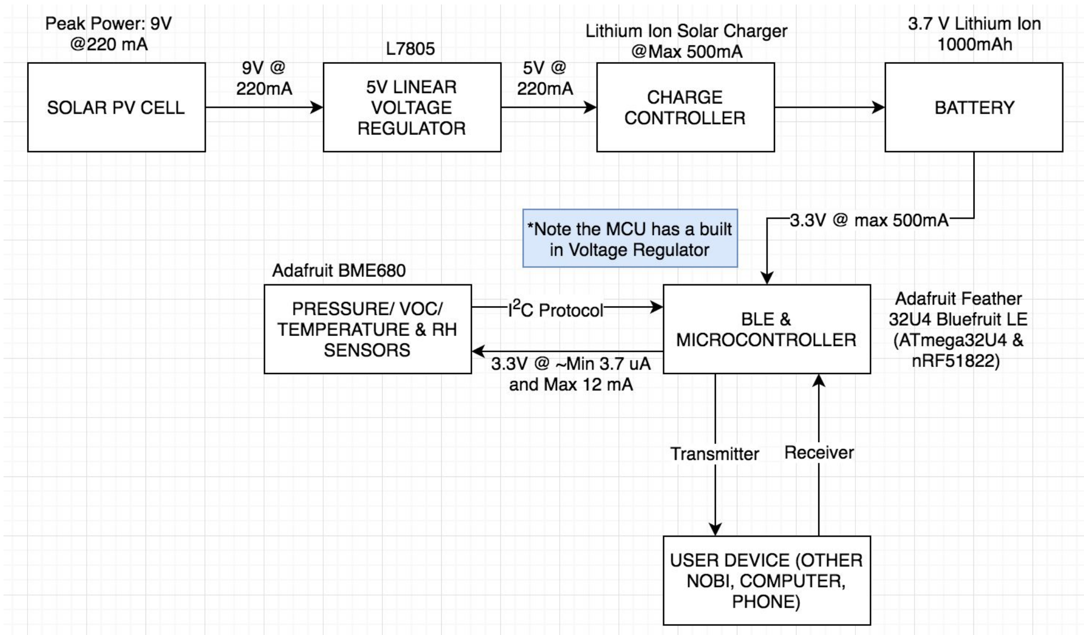

Figure 9. Final Block Diagram of the Nobi

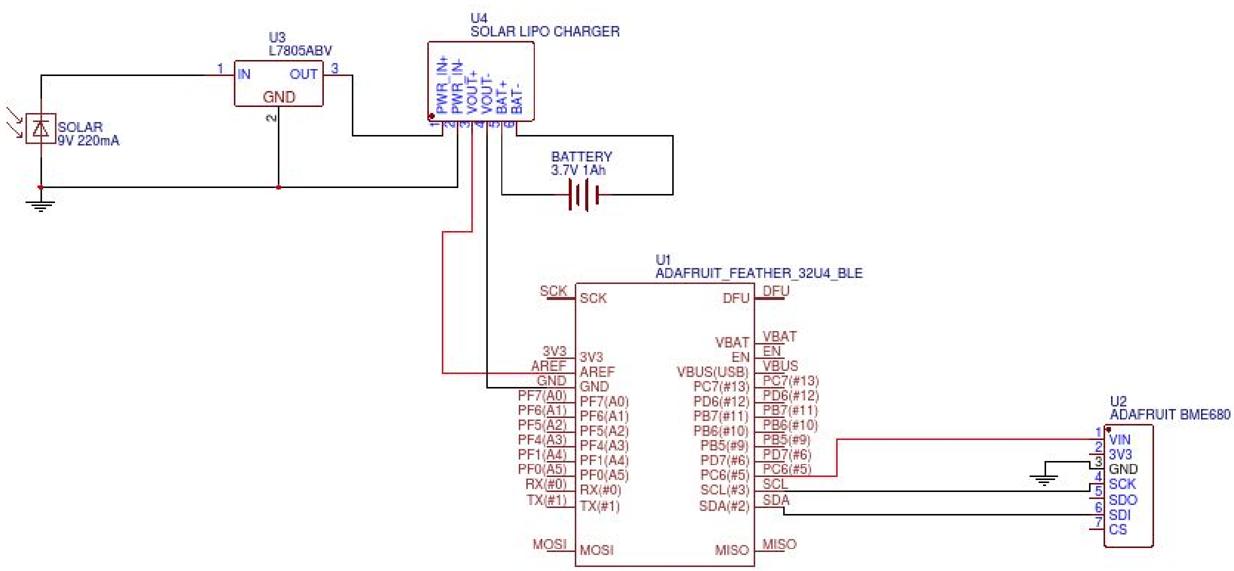

Figure 10. Schematic of the Nobi Prototype

3.1 System Architecture

Our product is made up of several components for the goal of making a fully-functioning wildfire detector device. As such, our components include a PV module, battery, voltage regulator, microcontroller and bluetooth module, and a multi-sensor device that detects temperature, humidity, gases, and pressure.

Our design starts with a PV module which allows the Nobi to harvest energy from its environment. From the solar cell, the voltage is reduced to be appropriate for the charge controller and the battery. The changing current from the solar cell is put through a charge controller to regulate the current. This outputs a smaller current, appropriate for our lithium battery, with no effect on the voltage.. The battery is our energy storage, with a rating of 3.7 volts and 1000 milliamp hours; this will be used to power components such as the bluetooth module and the sensor. We chose a microcontroller module with a built in bluetooth module and voltage regulator that will be connected to the sensors and battery. With the data given from the sensors, our Adafruit Feather 32u4 Bluefruit will then transmit this data to a receiver (cell phone, tablet, computer, etc.), thus completing the system. As a whole, our product will have the ability to send temperature, relative humidity, gas, and pressure data to the user; all powered by solar energy.

3.1.1 Energy Harvesting and Battery Charging System

As mentioned before, our device is divided into two parts. The first one is the energy harvesting and battery charging system which harvests stray energy from the environment and charges a battery. Naturally, this meets the customer requirements of being easy to use, low maintenance, and self-sustaining.

There are several modules included in this system such as the solar panel, voltage regulator, charge controller and battery. Energy is harvested via the solar panel which goes through the voltage regulator and the charge controller to prevent overloading and potential damage to the lithium ion battery.

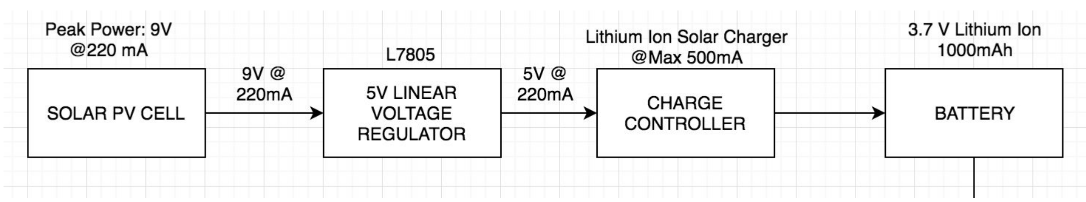

Figure 11. Energy Harvesting and Battery Charging System Block Diagram

This is a hardware system as it mainly deals with getting the right voltages and currents to charge the battery correctly. Unlike a software system, only the components are affecting the behavior of this system. Through the battery, this system is connected to the Bluetooth and Environment Sensing System.

3.1.2 Bluetooth and Environment Sensing System

The Bluetooth and Environment sensing system senses environmental conditions and wirelessly communicates that data to the specified device. The environmental conditions will alert the user if a wildfire is imminent or occurring. Furthermore, this will meet the customer requirements of being low maintenance and have wireless communication and environmental sensing.

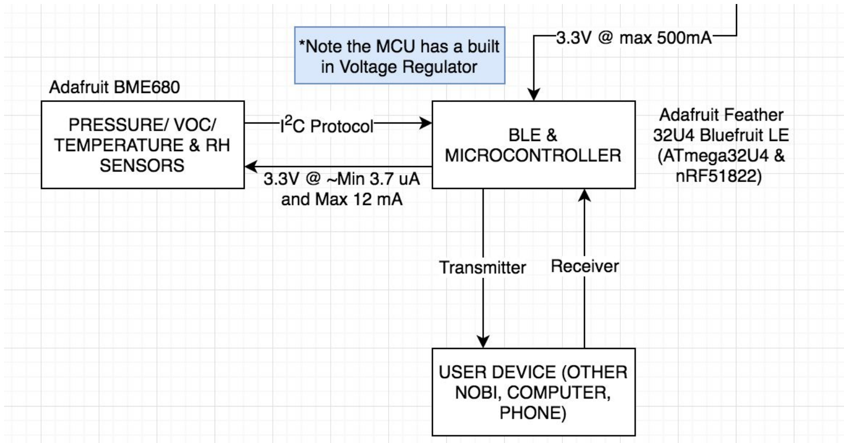

Figure 12. Bluetooth and Environmental Sensing System Block Diagram

As the block diagram above shows, the modules included are the microcontroller, BLE, and the BME680 sensor. The Adafruit Feather 32u4 Bluefruit LE combines both the microcontroller, an ATmega32u4, and the BLE, a nRF51822. This allows for less components to be needed making it more user-friendly and low maintenance. Furthermore, the BME 680 senses temperature, humidity, pressure and gas which is perfect for a wildfire detector device. The preferred implementation is both hardware and software, though mostly the latter. Hardware-wise, all the devices are connected (as shown in Figure 2) to work as efficiently as possible with one another. However, the software aspect will be crucial as it will control what is being sensed, sending or retrieving of information, when things are functioning, and more.

Shown below is the flow diagram of how we expect the software to work. The starting parameters are alert_Iteration which is set to 0 and, in reference to our state machine, the initial state is set to StandardState. The software will put the BME680 in *forced mode*, making it detect the temperature, humidity, pressure, and gases in the environment. If dangerous gas levels are detected, the device will switch to AlertState with a delay of 60 seconds rather than 300 seconds. An alert will be sent to the lookout towers and nearby Nobis, causing the surrounding Nobi devices to go into AlertState as well. Therefore, every minute, data will be sent to lookout towers with the data provided by the sensors. Once the gas levels reach safe levels, the device will return to StandardState; however, the user may also manually reset the device. On the other hand, if dangerous gas levels are not detected, the device will remain in the initial state, StandardState, and data will be sent to the lookout tower every five minutes.

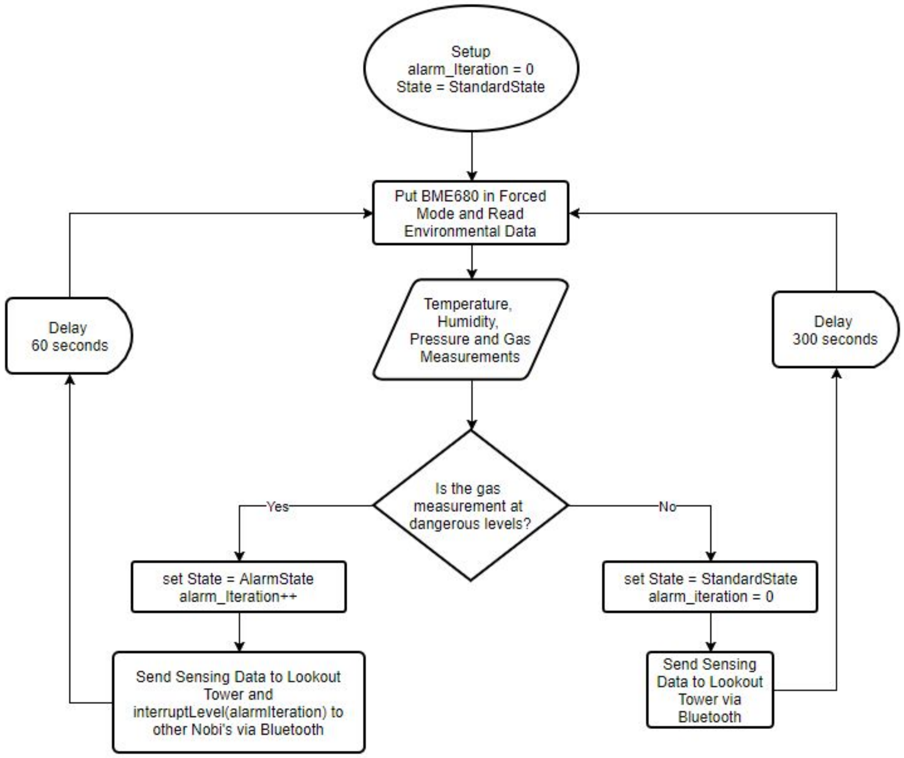

Figure 13. BLE and Sensor System Flow Diagram

3.2 Module Description & Analysis

This section will focus on each separate module of the components that make up the Nobi device. Each part will contain the purpose of the module, relevance to customer requirements, preferred implementation, and the input and output characteristics. Additionally, each piece will be categorized as a hardware or software module and tested.

3.2.1 Solar Photovoltaic Module

A solar cell is necessary to meet both a customer requirement and project requirement. For a device that will be put out in the forest, supplying it with power can be difficult. Our customer will not want to be changing batteries for this device throughout the year and it would be very hard to connect to a power source in the far out wildlands. Therefore, for a low maintenance product, we decided to have the Nobi be powered by solar energy. The second reasoning for this is that our project must harvest its own energy; this component thus meets both specified requirements.

Our solar panel was chosen for its low price and high power output and was purchased off Digikey for $7.90. With ultra-white glass lamination process to provide an efficient and waterproof surface, the solar panel has a slicker and more advanced design as shown in the figure below.

Figure 14. Solar Panel (9V 220mA)[18]

3.2.1.1 Input Signal Characteristics and Response

The input signal for our PV module is given by the sun, therefore can be unpredictable. This led us to research[19] year-long weather averages for a California customer base. California is known for its sunshine, but how much sunshine will this state actually supply? On average, California gets 66% to 88% of sunlight that reaches the ground from sunrise to sunset, yearly. At lowest of 66% sun, this still gives a propitious solar input for our PV module.

To achieve optimal input signal, the placement of our module is highly important. We aim to have this panel parallel to the East/West axis, so all angles of the sun are addressed with highest efficiency. To calculate the best angle from the horizontal, we plan to use the second formula from the figure below to reach out to areas such as California, with a latitude of ~37 degrees[20].

- If your latitude is below 25°, use the latitude times 0.87.
- If your latitude is between 25° and 50°, use the latitude, times 0.76, plus 3.1 degrees.
- If your latitude is above 50°, see Other Situations below.

Figure 15. Tilt Formulas[21]

For California customers, our tilt would be +31 degrees; a positive tilt angle means that the panel faces more towards the equator. In the Northern hemisphere that would mean tilting so it faces towards the South.

[18] https://www.digikey.com/product-detail/en/dfrobot/FIT0330/1738-1072-ND/6588494
[19] https://www.currentresults.com/Weather/California/annual-days-of-sunshine.php
[20] https://www.latlong.net/place/california-usa-2487.html
[21] http://www.solarpaneltilt.com/

3.2.1.2 Functional Description

A solar cell is a device that converts sunlight into direct current (DC) energy. This is done by two layers of semiconductors making up a thin semiconductor wafer. Doping this with Boron on one side and Phosphorus on the other creates a surplus of electrons on one side and a deficiency on the other. When sunlight hits this, the photons from the light knock off some of the excess electrons, thus giving current.[22] These cells are then circuited and sealed in an environmentally protective laminate, completing the building process of our solar module.

3.2.1.3 Output Signal Characteristics

At max power, this panel collects 9V and 220mA; this can range to a short circuit with a voltage of zero and a 500mA current. Below is the general I-V Characteristic of a solar cell, showing that when there is a short or open circuit, at either extreme, power is equal to zero. Our panel gives a max power of 1.8W, when the panel is most efficient.

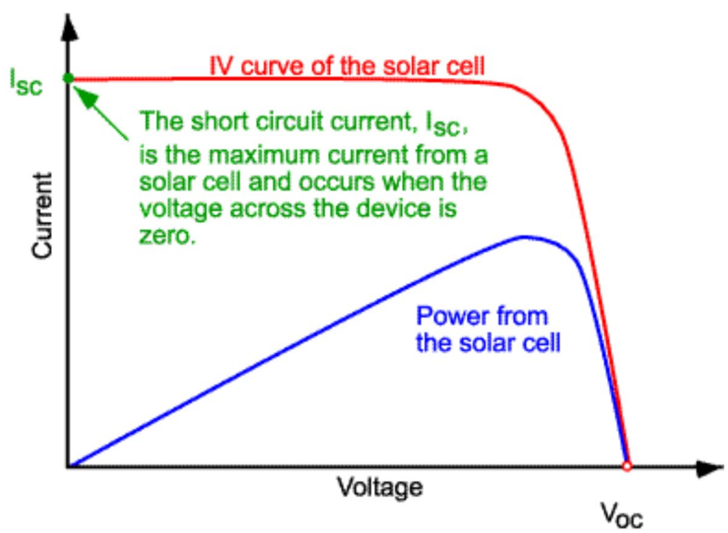

Figure 16. Solar Cell I-V Characteristic[23]

3.2.2 Voltage Regulator

The voltage regulator we are using is the 7805 Line Voltage Regulator in order to make sure that our solar panel does not overload the battery.[24] We believe that adding this is essential to the safety of the device. If the voltage is too high for our battery and charge controller, they may not work requiring the customer to replace the components. Therefore, in order to prevent this and

[22] http://www.solardirect.com/pv/pvlist/pvlist.htm
[23] http://www.pveducation.org/pvcdrom/short-circuit-current
[24] https://cdn-shop.adafruit.com/product-files/2164/L7805CV.pdf

have a low maintenance product, a voltage regulator was added in between the solar panel and the charge controller.

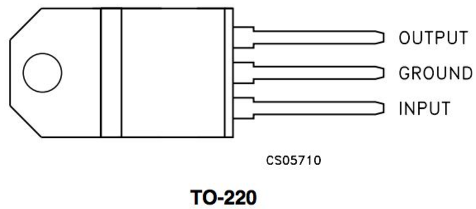

Figure 17. 7805 TO-220 Voltage Regulator Diagram

3.2.2.1 Input Signal Characteristics and Response

The regulator will help bring down our 9V voltage from the solar panel to 5.0V with 2% regulation, making this device perfect for our lithium ion battery. This particular model has a 1.5 current capability with internal current limiting and thermal shutdown making this regulator sturdy, electrically speaking.

According to the data sheet provided, the 7850 model has a ~2V linear drop-out, meaning that at least 7V must be provided to be able to use the regulator and output 5V. Additionally, the dormant mode current draw is 6mA. However, it is important to note that the device still needs proper heat-sinking. The higher the input voltage and output current, the more heat will be generated, creating a potential risk. As a result, we will be able to calculate the wattage:

$$Wattage = (Input\ Voltage - 5V) \times Average\ Current\ in\ Amps$$

3.2.2.2 Output Signal Characteristics

Using the data sheet, we saw that this voltage regulator connects easily with other components. Additionally, we found that the output voltage usually is between 4.9V and 5.1 V. Using the table below we can also see the current when in low power mode and more. Furthermore this voltage regulator will help for proper use of the charge controller and the battery.

Table 5. Electrical Characteristics of 7850

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
V_O	Output voltage	$T_J = 25°C$	4.9	5	5.1	V
V_O	Output voltage	I_O = 5 mA to 1 A, V_I = 7.5 to 18 V	4.8	5	5.2	V
V_O	Output voltage	I_O = 1 A, V_I = 18 to 20 V, $T_J = 25°C$	4.8	5	5.2	V
ΔV_O (1)	Line regulation	V_I = 7.5 to 25 V, I_O = 500 mA		7	50	mV
		V_I = 8 to 12 V		10	50	mV
		V_I = 8 to 12 V, $T_J = 25°C$		2	25	mV
		V_I = 7.3 to 20 V, $T_J = 25°C$		7	50	mV
ΔV_O (1)	Load regulation	I_O = 5 mA to 1 A		25	100	mV
		I_O = 5 mA to 1.5 A, $T_J = 25°C$		30	100	V
		I_O = 250 to 750 mA		8	50	V
I_q	Quiescent current	$T_J = 25°C$		4.3	6	mA
					6	mA
ΔI_q	Quiescent current change	V_I = 8 to 23 V, I_O = 500 mA			0.8	mA
		V_I = 7.5 to 20 V, $T_J = 25°C$			0.8	mA
		I_O = 5 mA to 1 A			0.5	mA

3.2.3 Charge Controller

The charge controller we decided on was the 8-bit CN3083 which is compatible with both a solar powered system and lithium ion rechargeable batteries. The 8-bit ADC can adjust the charging current autonomously based on the output capability of the input power supply. Additionally, the constant current is ideal for lithium ion batteries. This device was considered essential as to not overcharge our battery and cause it to explode. Furthermore, it would prevent the need for an external sense resistor and blocking diode, making the Nobi more reliable and safe.

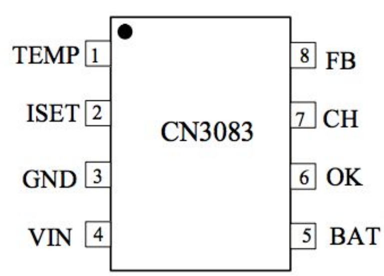

Figure 18. Pin Assignment of CN 3083

3.2.3.1 Input Signal Characteristics and Response

The charge controller requires a voltage of between -0.3V and 6.5V. Any more than that and the device could be permanently damaged causing our customer to have to replace the component. Therefore, we added a voltage regulator before the charge controller to prevent such a situation.

All Terminal Voltage……………−0.3V to 6.5V
BAT Short-Circuit Duration………...Continuous
Storage Temperature…............−65℃ to 150℃
Lead Temperature(Soldering)……............300℃
Maximum Junction Temperature…...150℃
Operating Temperature…...−40℃ to 85℃
Thermal Resistance (SOP8)…………TBD

Additionally, we will be looking carefully at the absolute maximum ratings of the device that are presented above. The regulation voltage of the charge controller is internally fixed at 4.2V with 1% accuracy, but can be modified using a resistor. Furthermore, once the CN3083 reaches low power sleep mode, the battery drain current is less than 3uA.

3.2.3.2 Output Signal Characteristics

The internal thermal regulation circuit prevents the CN3083 from becoming overloaded from excessive temperature which is necessary considering the purpose of our device.[25] Additionally, due to this, the charge current can be set to the average, ambient temperatures from say California and the current will automatically reduce in worst-case scenarios.

Using this device allows us to program the charge current using the following formula:

$$I_{CH} = \frac{1800\ V}{R_{ISET}}$$

The I_{CH} stands for the charge currents in ampere, while the R_{ISET} is the total resistance from the ISET pin to the ground in ohms. So as an example, if we required a 600mA charge current, we would be able to calculate a resistance of:

$$R_{ISET} = \frac{1800\ V}{0.6A} = 3.0\ k\Omega$$

However, if the charger is in constant-temperature or constant voltage mode, the charge current can be measured through the ISET pin voltage. The following equation can be used:

$$I_{CH} = \frac{V_{ISET}}{R_{ISET}} \times 900$$

[25] http://www.robotshop.com/media/files/pdf2/datasheet-dfr-565.pdf

Table 6. Electrical Characteristics of the CN3083

Parameters	Symbol	Test Conditions	Min	Typ	Max	Unit
Input Supply Voltage	VIN		4.4		6	V
Operating Current	I_{VIN}	Charge Termination Mode	400	650	950	uA
Undervoltage Lockout	Vuvlo	VIN falling		3.7	3.9	V
Undervoltage Lockout Hysteresis	Huvlo			0.1		V
Regulation Voltage	V_{REG}	Constant Voltage Mode	4.158	4.2	4.242	V
BAT pin Current	I_{BAT}	R_{ISET}=3.6K, V_{BAT}=3.6V	400	500	600	mA
		R_{ISET}=3.6K, V_{BAT}=2.4V	25	50	75	
		V_{BAT}=V_{REG}, standby mode	1.75	3.5	7	uA
		VIN=0V, sleep mode			3	
Precharge Threshold						
Precharge Threshold	V_{PRE}	Voltage at BAT pin rising	2.9	3.0	3.1	V
Precharge Threshold Hysteresis	H_{PRE}			0.1		V
Charge Termination Threshold						
Charge Termination Threshold	Vterm	Measure voltage at ISET pin	0.18	0.22	0.26	V
Recharge Threshold						
Recharge Threshold	V_{RECH}			$V_{REG}-0.1$		V
Sleep Mode						
Sleep Mode Threshold	V_{SLP}	V_{IN} from high to low, measures the voltage ($V_{IN}-V_{BAT}$)		40		mv
Sleep mode Release Threshold	V_{SLPR}	V_{IN} from low to high, measures the voltage ($V_{IN}-V_{BAT}$)		90		mv

3.2.4 Battery

All the energy harvested from the solar panel must be stored for the Nobi device, this will also be useful in cases such as the winter, when there is minimal sunlight. The device will be powered by a rechargeable lithium ion battery that outputs a nominal 3.7V at 1000mAh.

Figure 19. Lithium Battery 3.7 V, 1 Ah[26]

Ideally, this battery will take approximately 4.5 hours to fully charge which was calculated using the charge of the battery, Ampere hours, and the average input current charging the battery.

$$\frac{1000mA*hours}{220mA} = 4.545\ hours$$

However, charge time will range depending on supplied current from the PV module discussed in section 3.2.1.1.

3.2.4.1 Module Inputs

For safety reasons, this battery must be charged using a lithium polymer charger as shown in section 3.2. Basically, the charger will keep the current, or charge rate, constant until the battery reaches its peak voltage. Then it will maintain that voltage, while reducing the current. This is necessary to prevent our battery from catching fire, something that a wildfire detector should implicitly be preventing.

Worst-case scenario, as told in section 3.1.1, there is 4 hours per day on average when the solar module with be collecting energy. Thus supplying the battery with around 250mA of current. With an estimated max total current draw of 37mA, we presume the Nobi will work year round.

3.2.4.2 Output Signal Characteristics

This battery has a standard 2-pin JST-PH connector with 2mm spacing between pins for easy implementation. We plan to connect these to our microcontroller and bluetooth module to charge it with 3.7 V. In turn, the battery will also charge our sensor from an internal voltage regulator within the feather microcontroller.

3.2.5 RH, Temperature, Gas & Pressure Sensor[27]

[26] https://www.digikey.com/products/en?keywords=1568-1492-ND

[27] https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BME680-DS001-00.pdf

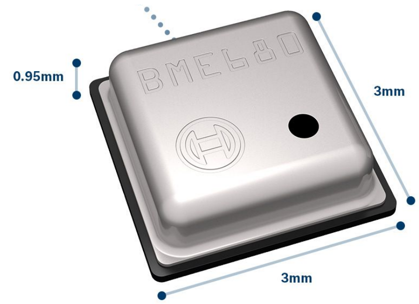

Figure 20. BME680 Integrated Environmental Unit Bosch Sensortec[28]

This module consists of the Adafruit breakout board[29] which uses the BME680 as a low powered gas, pressure, temperature and relative humidity sensor. As implied, this module will be used to sense these environmental conditions in forest or wildland areas that are greatly prone to wildfires. This module is very important for the Nobi system as it not only helps detect areas that are prone to wildfires, but also helps in detecting an ongoing fire and measure the health implications it may have on people within the vicinity. The preferred implementation of this module is to be both a hardware and software system because it will sense environmental conditions whilst sending data to a microcontroller and manipulating that data via algorithms.

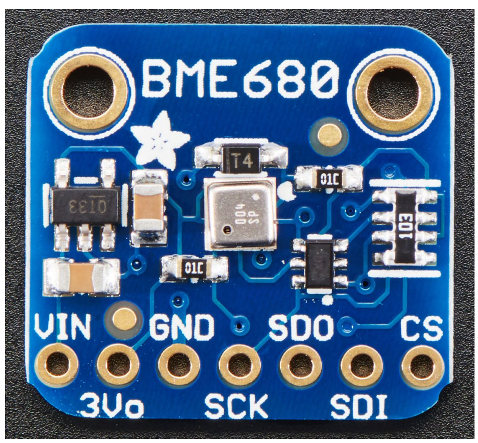

Figure 21. Adafruit BME680 Breakout Board[30]

3.2.5.1 Input Characteristics

This breakout board developed by Adafruit operates using a 3-volt to 5-volt power supply. The board itself has a built in voltage regulator that can cut down the operating voltage to the necessary value needed to optimally run the sensor – which is 1.8-volts. According to the

---

[28] https://www.bosch-sensortec.com/bst/products/all_products/bme680
[29] https://www.adafruit.com/product/3660
[30] https://learn.adafruit.com/assets/48064

datasheet provided by Bosch Sensortec, the BME680 is a digital 4-in-1 sensor with gas, humidity, pressure and temperature measurement "based on proven sensing principles." Its small dimensions and its low power consumption enable the integration of in battery-powered devices, such as the Nobi.

Depending on the mode, and on what it is sensing, it draws different amounts of current ranging from as little as 0.15 uA (sleep current) to as high as 12mA (average supply current in heater continuous mode). However, the amount of current drawn by this device will depend on each type of sensor. The main sensors used are the temperature, relative humidity, and MOX sensors. It is expected, that each sensor will run separately from one another, in order to not affect each other's results, especially since the MOX/gas sensors needs to heat itself in order to collect information.

All sensors in the BME680 will operate using the *forced mode* and *sleep mode* power mode. In the forced mode, the sensor performs a single measurement on request and returns to sleep mode afterwards. This mode is ideal for our application as the Nobi will use the sensor in a timely set basis.

Temperature, Pressure, and Humidity Sensor Specification

In the forced mode, a single TPHG (Temperature, Pressure, Humidity and Gas) cycle is performed sequentially. When the temperature sensor is operating, in forced mode it will draw 1 uA. All sensors can operate between -40 ˚C to 85 ˚C, which is an ideal region since the weather in California ranges between a minimum of 0 ˚C to 40 ˚C.[31]

The pressure sensor is not as important in the Nobi as the rest of the sensors. However, if it is decided that it will be operational, in forced mode it will draw approximately 3.1 uA. Lastly, when the humidity sensor is operating, in forced mode it will draw around 2.1 uA, and output data back with a response time of 8 seconds. Both the pressure and humidity sensor can fully operate accurately in a range of 0 ˚C to 65 ˚C. That's good because weather in California fits within that region. Thus it is expected that the sensor will yield with utmost accuracy.

Gas Sensor Specification

When a metal oxide is heated, it changes its resistance based on the volatile organic compounds (VOC) in the air, thus making the gas/MOX sensor detect gases such as Carbon Monoxide and perform air quality measurements. However, the sensor itself can only output one resistance value with an overall VOC content – it cannot differentiate gasses or alcohols. However, this will

[31] https://www.usclimatedata.com/climate/california/united-states/3174#

not be a problem because of the environment and the altitude that Nobi will be placed in. If the sensor detects a sudden change in gasses, then it will most likely be assumed as a fire hazard.

Table 7. Gas Sensor Specifications

Average Supply Current ($V_{DD} \leq 1.8$ V, 25°C)	$I_{DD,IAQ}$	Ultra-low power mode		0.09		mA
		Low power mode		0.9		mA
		Continuous mode		12		mA
Response time[2] (brand-new sensors)	$\tau_{33\text{-}63\%}$	**Ultra-low power mode**		**92**		**s**
	$\tau_{33\text{-}63\%}$	Low power mode		1.4		s
	$\tau_{33\text{-}63\%}$	Continuous mode		0.75		s

For sensing gas, the sensor has three different types of mode functions: Ultra-low power mode, low-power mode and continuous mode. Each mode is designed for a specific purpose. The continuous mode is used for testing purposes only, especially in fast events or stimulus, and can consume an average of up to 12mA due to an update rate of 1Hz. The low power mode is designed for interactive applications where air quality is tracked and observed at a higher update rate of 3 seconds with an average current consumption of 0.9mA. Finally, the Ultra-low power mode is designed for battery powered over extended periods of time. It features an update rate of 300 seconds and an average current consumption of 0.09mA. However, the less power a mode consumes, the higher the response time. The low power mode has a response time of 1.4 seconds which is roughly 90 times faster than the ultra-low power mode. Considering that a quick response time is ideal, especially in sensing fires, the MOX sensor is expected to work in a low-power state.

Communication

The BME680 sensor is a slave device used by the I^2C digital interface protocol. The I^2C interface supports the standard, fast and high speed modes, and can support the following transactions: singly byte write, multiple byte write, single byte read, and multiple byte read. The 7-bit device address of the BME680 is 0x77 by default (however, placing a jumper between SDO to GND on the breakout board changes the address to 0x76). The microcontroller will write to the sensor in order change the sensor mode from *sleep mode* to *forced mode*.

Total Input Power Consumption

Combining all input requirements for each sensor in the BME680, the estimated max current needed to be drawn by the BME680 is approximately 1mA. That analysis was done using values that were mentioned above which were 900uA, 2.1 mA, 3.1 mA, and 1mA for the gas, humidity, pressure, and temperature sensors respectively. An extra 100 uA was included for error purposes, and extra current drawing that may occur.

3.2.5.2 Functional Description

After the power-up sequence, the sensor automatically starts in *sleep mode*. The microcontroller (master device) can change the sensor mode to *forced mode* in order to start environmental sensing measurements. After *forced mode* is done, the sensor automatically returns to *sleep mode* afterwards. In *sleep mode*, no measurements are performed, and there is minimal power consumption at around 0.15uA.

As soon as the sensor is placed in *forced mode*, as mentioned previously, a single TPHG (Temperature, Pressure, Humidity and Gas) cycle is performed sequentially. The diagram below shows how the sensors will work one after the other. According to the BME680 datasheet "the BME680 measurement period consists of temperature, pressure and humidity measurement with selectable oversampling. Moreover, it contains a heating phase for the gas sensor hot plate as well as a measurement of the gas sensor resistance." After the measurement period is complete, the temperature and pressure data can be passed through an IIR filter, which removes short-term fluctuations from outside disturbances.

Figure 22. Forced Mode operation in the BME680 with TPHG cycle

Unlike the rest of the sensors, the gas sensing part of the BME680 involves two steps: (1) Heating the gas sensor hot plate to a target temperature (typically between 200 ˚C and 400 ˚C) and keeping that temperature for a certain duration of time. And (2) Measuring the resistance of the gas sensitive layer.

3.2.5.3 Output Signal Characteristics

The BME680 output consists of the ADC output values. Each sensing element behaves differently. The relative humidity, pressure and temperature are calculated using a set of calibration parameters, whilst the gas sensor retrieves its ADC value from the resistance of the gas sensitive layer. In general, for the humidity, pressure and temperature sensors, the output data has a resolution of 16 bits. For the temperature and pressure data as mentioned previously, if the IIR filter is enabled, their output data has a resolution of 20 bits. To suppress disturbances (i.e. wind blowing into the sensor) in the output data, the sensor features an internal IIR filter as it effectively reduces the bandwidth of the temperature and pressure output signals and increases the resolution of the output data to 20 bit. The humidity and the gas values inside the sensor don't fluctuate rapidly, and thus don't need low pass filtering. With the I^2C interface, the converted data will be read by the microcontroller.

3.2.6 Microcontroller and Bluetooth Module

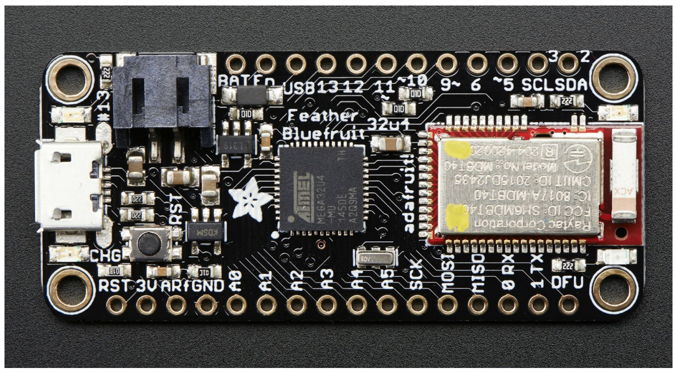

Figure 23. Adafruit Feather 32u4 Bluefruit LE[32]

The Adafruit feather 32u4 Bluefruit LE[33] is an all-in-one Arduino compatible microcontroller with a BTLE module. This is a perfect development board because it uses both a mesh-compatible Bluetooth Low Energy device programmed with multi-function code, but it also has a programmable microcontroller that is compatible with the Arduino IDE. The MCU is the ATmega32u4, whilst the BLE module is the nRF51822 chipset from Nordic. This module is the brain of the Nobi system as it controls the data collected from the environment via the BME680, and sends that data to other Nobi devices or lookout-towers via the BLE module. The preferred implementation of this module is to be both a hardware and software system because it controls other sensors and modules, and because it utilizes the Arduino IDE to perform tasks that will make the Nobi a wildfire detection sensor, and possibly a wildfire prevention sensor. However the latter is preferred for implementation of the system.

3.2.6.1 Input and Output Characteristics

The Adafruit Feather 32u4 Bluefruit LE operates with a 3-volt to 5-volt power supply, however it is expected to be used with a 3.7-volt lithium ion battery or a 5 volt USB connection. This development board has a 3.3-volt regulator with a 500mA peak current output in order to cut down the input voltage to a safe and optimal value due to the power supply range of each

[32] https://learn.adafruit.com/assets/28634
[33] https://learn.adafruit.com/adafruit-feather-32u4-bluefruit-le

component. It comes with a battery connector and with a built in 100mA lipoly charger which can come in handy in debugging the device away from the computer.

The BLEs outputs includes both transmitting and receiving data to and/or from other devices with bluetooth capabilities. On the other hand, the output for the microcontroller is different. As mentioned in the section of the BME 680 module, the microcontroller can output analog or digital information in order to make a device function a certain way. The sections below, describe both the input and output power consumptions of the ATmega32u4 and the nRF51822 bluetooth module.

Microcontroller: ATmega32u4[34]

The ATmega32u4 operates between a voltage range of 2.7-volts to 5.5-volts and consumes around 2mA to 27 mA in idle mode and active mode. During power-down mode, it uses less than 10uA. With a total of 44 pins, as seen in the schematics, only 5 programmable I/O pins are used by the sensor, and 3 pins are internally used by the nRF51 Bluetooth Module. Each pin can supply or take an input current of 10uA. Thus it can be estimated, that 10 pins will use approximately 100uA. However, depending on the output of that programmable I/O pin, it may need to draw more current than each single wire.

Table 8. Power supply current of the ATmega32u4 MCU

Symbol	Parameter	Condition	Min.[5]	Typ.	Max.[5]	Units
I_{IH}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin high (absolute value)			1	µA
R_{RST}	Reset Pull-up Resistor		30		60	kΩ
R_{PU}	I/O Pin Pull-up Resistor		20		50	
I_{CC}	Power Supply Current[6]	Active 4MHz, V_{CC} = 3V (ATmega16U4/ATmega32U4)			5	mA
		Active 8MHz, V_{CC} = 5V (ATmega16U4/ATmega32U4)		10	15	
		Active 16MHz, V_{CC} = 5V (ATmega16U4/ATmega32U4)			27	
		Idle 4MHz, V_{CC} = 3V (ATmega16U4/ATmega32U4)			2	
		Idle 8MHz, V_{CC} = 5V (ATmega16U4/ATmega32U4)			6	

The current consumption is a function of several factors such as: operating voltage, operating frequency, loading of I/O pins, switching rate of I/O pins, code executed and ambient temperature – however, the dominating factors are operating voltage and frequency. Because the

[34] http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf

Feather has the ATmega32u4 already clocked at 8MHz at a 3.3-volt logic, as seen on the graphs below on Figure 15 and Figure 16 respectively, at 25 ˚C the current consumption is roughly around 4.5 mA in Active Mode, while in idle mode, it has a current consumption of 1.5 mA.

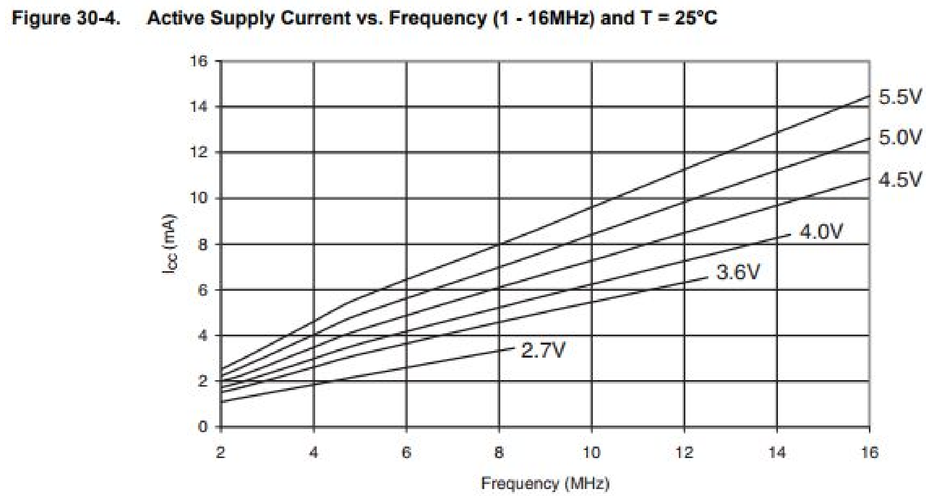

Figure 24. Active Supply Current vs Frequency Graph

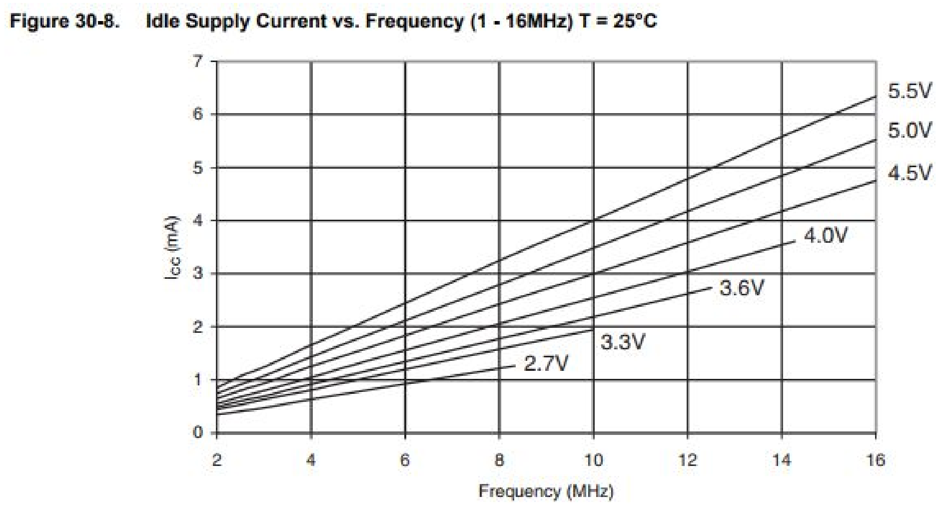

Figure 25. Idle Supply Current vs Frequency Graph

Bluetooth Module: nRF15822[35]

This module has an input supply range of 1.8-volts to 3.6-volts. Looking at the modes in the nRF15, the most viable option is the system ON mode because "here the system is fully operational and the CPU and selected peripherals can be brought into a state where they are

[35] http://infocenter.nordicsemi.com/pdf/nRF51822_PS_v3.1.pdf

functional and more or less responsive depending on the sub-power mode selected." According to the data sheet, with all blocks idle, the system ON mode draws as much as 2.6 uA.

Since the bluetooth module is connected to the microcontroller using the I^2C protocol, it takes about 380 uA in transferring information. The UART specifications state that around 200 uA will be drawn in receiving and transmitting data.

Table 9. Input Voltage of the Bluetooth Module

Symbol	Parameter	Notes	Min.	Typ.	Max.	Units
VDD	Supply voltage, normal mode		1.8	3.0	3.6	V
VDD	Supply voltage, normal mode, DC/DC converter output voltage 1.9 V		2.1	3.0	3.6	V
VDD	Supply voltage, low voltage mode	1	1.75	1.8	1.95	V
t_{R_VDD}	Supply rise time (0 V to 1.8 V)	2			60	ms
T_A	Operating temperature		-40	25	85	°C

Total Power Consumption

Looking at all the values of Adafruit Feather 32u4 Bluefruit LE module, it is expected that with an input voltage of 3.3 volts, the maximum amount of current that will be drawn will be at a value of 6 mA. This is only an estimation. The actual amount of current drawn may be much smaller than that, at around 4 mA.

3.2.6.3 Functional Description

The microcontroller is the brain of the system as it will be instructing the sensors and the BLE in the things it needs to do. As mentioned previously, the microcontroller, BLE, and the sensors are all connected by hardware. However, they are integrated more thoroughly using software. The microcontroller will be programmed using the Arduino IDE. The way it will be programmed will determine the amount of power consumed for that subsystem. The Figure below is similar to the figure shown in section 2 - it is a flowchart of how the microcontroller will be programmed in order for the Nobi to function properly. The setup mode is expected to run once, the moment after the MCU is powered. After that, the device will loop through similar commands to get the specified results.

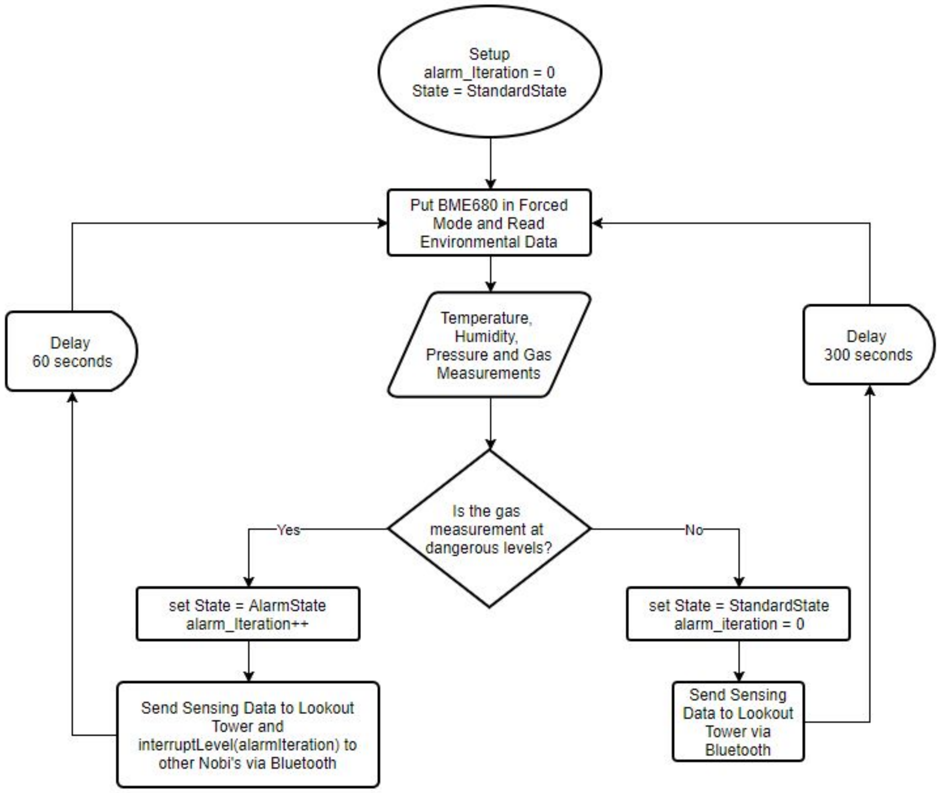

Figure 26. Flow Diagram of Software Modules

The Nobi is expected to communicate with other Nobi devices in their proximity in order to transfer the information to the lookout tower (that is assumed to be far from some sensors). Ideally, once one device senses danger, it should notify other devices nearby it of the possible danger. Another flowchart, as shown in Figure 19, shows an Alarm Interrupt that should theoretically change the delay time (make sensing more frequent) of the sensor during its waiting periods. The detection of fire, can be better controlled if some sensors knew the fire was coming.

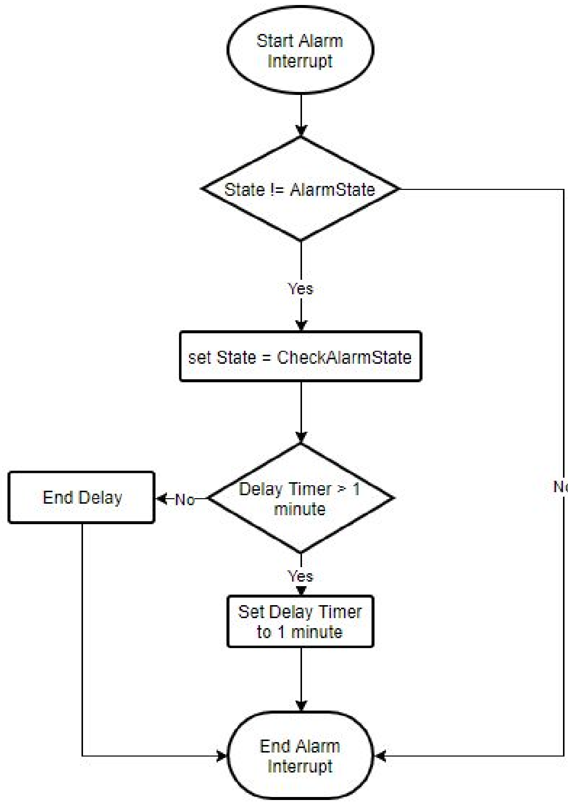

Figure 27. Alarm Interrupt of the Nobi Flow Chart

The flowcharts shown are general representations of what should happen in the system. In reality, if the Nobi knows that there is an ongoing fire, then it will continuously run without a waiting period. This will consume a lot of energy, but because it is in the danger of being consumed by the flames, it won't matter.

3.3 Defining Product Changes and Additions

3.3.1 Software Modifications

Since the software was tested in a specific area, it was not accounted for that different places would cause different ambient readings for the gas measurement. After the third design review, although the prototype was working as expected, it wasn't able to get the correct readings due to the new environment. As explained previously, the gas sensor in the BME680 heats up and measures a metal oxide resistance which collects the overall volatile organic compounds (VOCs) in the air. A way this can be solved, is by automatically calibrating the device when its initially setup. The flowchart below shows the steps taken for obtaining calibration data, and for setting the gas values that will change the state of the sensor to danger state, alert state or standard state.

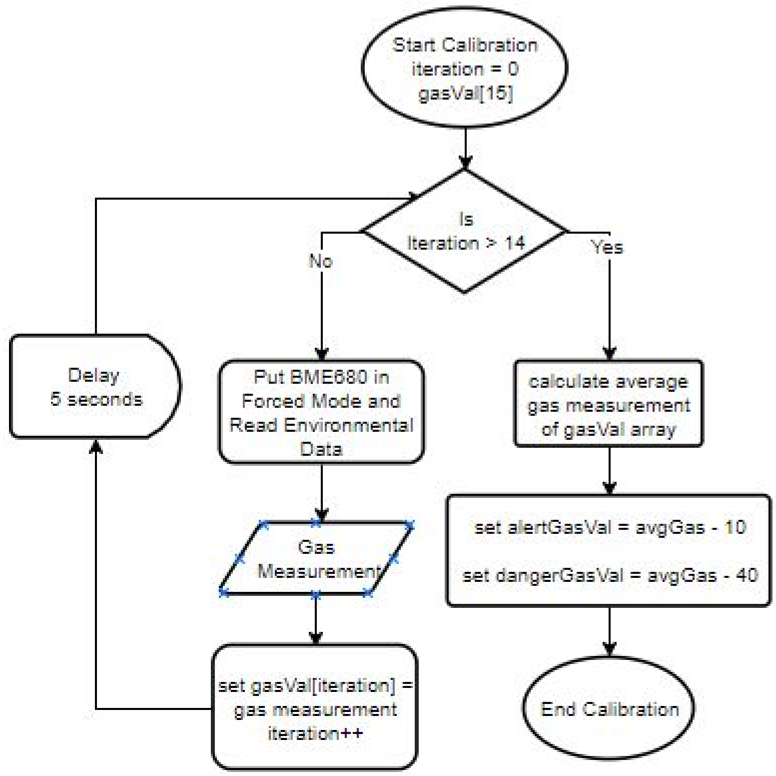

Figure 28. Calibration Flow Chart

As seen, every 5 seconds for 15 iterations, that metal oxide resistance will be measured and stored in an array. After the last iteration, the average of all measurements is calculated which then determines that gas levels necessary for the Nobi to be in danger state or alert state. However, these values were chosen for prototyping means and are susceptible for change in future prototypes. Nonetheless, if placed in an environment whose ambient "normal" gas measurements are above 50 kΩ, then the program could function as expected.

A change in the overall flow chart was the addition to a danger state. The danger state would be initiated after very high levels of gases have been detected by the Nobi (which would be set in the calibration stage). From there, it would collect data every 2 seconds as the more information on the gas levels could help notify occupants on the danger of the spread of the fire and of bad gases. At this state it is assumed that there is a fire. Power consumption isn't of importance as the Nobi will most likely catch on fire – therefore more information on what is happening is better than none.

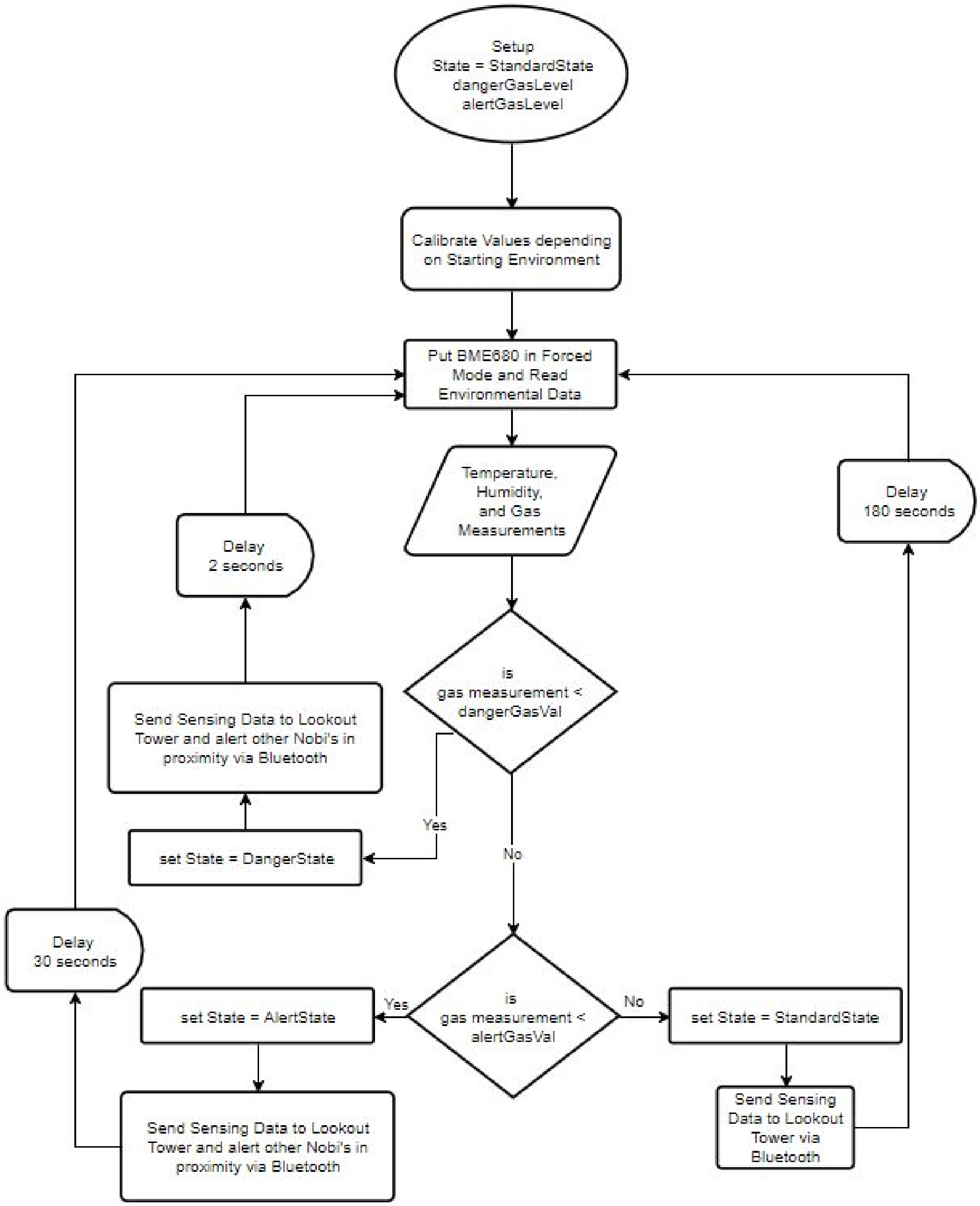

Figure 29. Updated Overall Nobi Flow Chart

Another change, were the delays of standard and alert state which were moved from 5 minutes and 1 minute respectively to 3 minutes and ½ minute. This was done as research showed that fires can greatly spread in a span of 30 seconds[36][37][38]. However, delays can most likely be shortened if future research shows otherwise.

---

36 https://www.youtube.com/watch?v=piofZLySsNc
37 https://www.gtmetrofire.org/public-education/safety-information/fire-spread-and-fire-drills/
38 https://skysaver.com/blog/how-fast-fire-spread/

In the previous paper on Product Interfaces, it was briefly mentioned that the future of the Nobi system would heavily rely on a Bluetooth mesh network. The future Nobi will most likely include an interrupt implementation. If a Nobi is set to alert state, then any other devices in its proximity will check whether they are in danger of a fire. Since fires spread quickly, for devices in standard state, this interrupt will come in handy as it will help alert the device sooner than later. It is best to not have the device engulfed in flames when it begins to do its measurement readings. This implementation can be seen in the flowchart below.

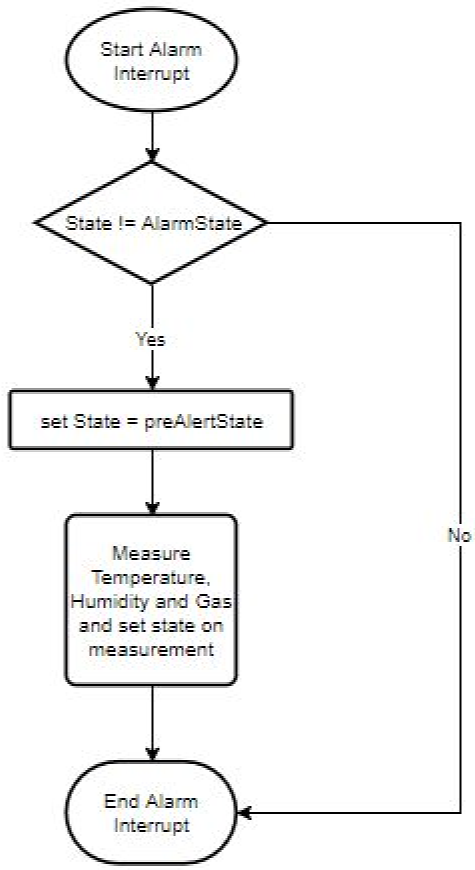

Figure 30. Updated Bluetooth Mesh Network Interrupt Flowchart

For the prototype, and for debugging purposes

3.3.2 Installation

In order to achieve the best placement for the Nobi, we considered mounting it at the top of the tree.This would involve screwing into the heartwood[39] of the tree to assure upmost security for the Nobi device. The Nobi must be placed near the top to avoid obstruction of sunlight needed

[39] https://www.arborday.org/trees/treeguide/anatomy.cfm

for the panel. With our market consisting of large affiliations such as the government and EPA, we believe there will be no problem with a more intricate installation process.

Research as to where to mount this device was completed based on the types of trees located in California. Fire-resistant pines, including the Ponderosa Pine[40], shed their lower, more vulnerable branches as they mature. This tree species suffers virtually no crown damage under a naturally mild fire regime, thus making it the perfect home for a Nobi device.

Upon completing this research, we were then able to consider arrangements for mounting our device. In our first design, the Nobi would be screwed into a metal platform welded to a heavy duty custom bolt.

Figure 31. Heavy Duty Custom Bolt[41]

This strong custom bolt will provide support with only one puncture point into the tree. The bolt is capable of lasting 5 years while supporting heavy objects, so with our light design we expect this to hold the Nobi for at least 5 to 10 years. With the customizability of the bolt, we will be able to advertise various sized bolts to our customers. Bolt sizes were decided based on the diameter range for the Ponderosa Pines. Ponderosa Pines range from around 30 inches to as wide as 2 meters, therefore our bolts are expected to have a varying diameter of 5 to 15 inches with a length of 15 inches to a meter. This is with expectation that the user will choose a suitable tree for the sizes we supply, although a general instruction manual will be given for users planning to buy their own equipment tailored to their specified tree. In the case that tree branches are obstructing light for the panel, we planned to advise our customers to prepare for possible trimming of the surrounding branches.

This was our ideal design due to the durability of this method, however to maintain an inexpensive product we were forced to reconsider. Instead of selling our product with a

40 http://anrcatalog.ucanr.edu/pdf/8386.pdf
41 https://www.familyhandyman.com/garden-structures/tree-house-building-tips/view-all/

predetermined installation method, we now have a recommended installation strategy, allowing the organization to do with our product what they please. This deemed to be a more attractive solution due to the capability of the government and EPA as well as keeping the focus for our product to be functionality and affordability.

A strategy we recommend would be to make one to two meters in diameter clearings around the forest where “Nobi Towers” can be installed. As well as the Nobi Towers, some Nobi’s can be placed within the firewalls of the California wildlands. These firewalls were created in attempt to stop the progression of a wildfire by having strips of cleared land. Placement of a Nobi would be made easy by these areas, however this would not be the only locations for this device. The problem with having the Nobis primarily within the firewalls, is the device would likely not catch the initial fire. Thus why clearings within the forest at key points would be necessary for optimal detection.

3.3.3 Data Visualization

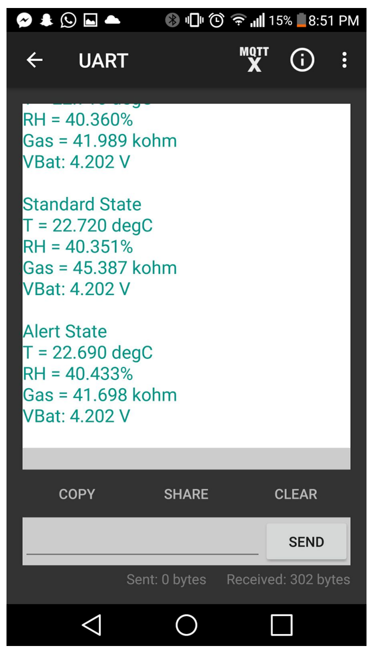

Figure 32. Screenshot of UART Bluefruit LE App Receiving Data from the Nobi

As of now the collection of data is done through a universal asynchronous receiver transmitter (UART) phone application by Adafruit. It is called *Bluefruit LE* and it can be used by both android and iOS users. In UART mode, the user receives the data transmitted by the BLE module in the Nobi. In the third design review, this was clearly shown in the prototype. However, as seen in the figure above, there was a lot of information being spewed really quickly to the user. Even though the final device will be sending data in longer time intervals, that amount of information can be hard to handle and understand. A solution to this is data visualization.

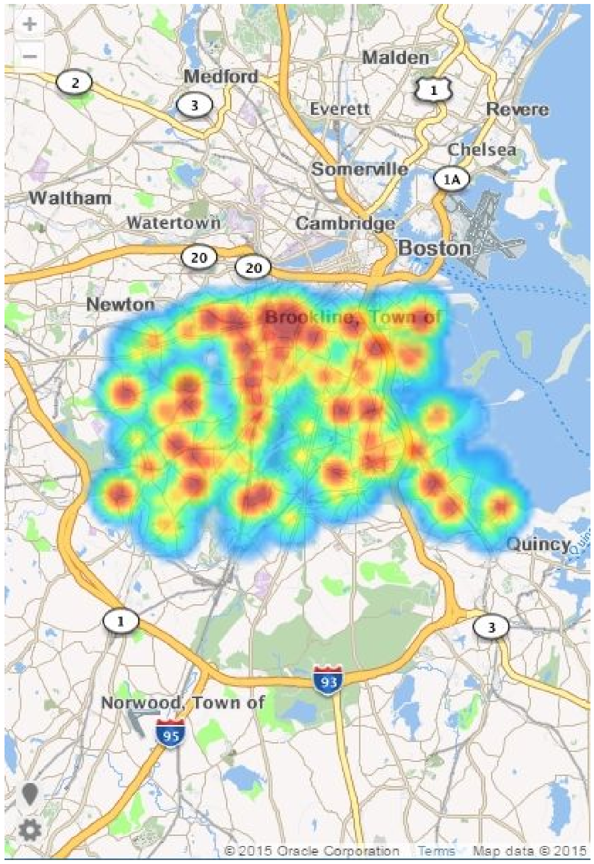

Figure 33. Oracle Data Visualization[42][43]

The final Nobi system will be expected to use data visualization when portraying the data. Figure 8 is an example by Oracles Data Visualization of what the information from the Nobi mesh system should portray. Doing this allows the users to easily visualize areas of concern in the forest. Visualizing the data not only makes it easier to understand the data at a larger scale, but it can also allow for quicker steps to be taken to prevent wildfires.

Through GPS, knowing the location of each device not only will it be able to place the retrieved information at a certain place on a map, but with certain algorithms, the retrieved data will not be limited to what the Nobi can sense. By collecting the measurement data from individual Nobi devices at specified intervals, and by knowing their location, it is expected that with state-of-the-art algorithms the speed and direction of the wildfire can also be found. This type of data will help in quickly alerting people of danger and giving them a chance to evacuate safely.

[42] https://www.oracle.com/solutions/business-analytics/data-visualization.html
[43] https://blogs.oracle.com/analyticscloud/heat-up-your-maps-with-oracle-data-visualization

4 Product Results

This section discusses whether the prototype functioned as expected, and what the system testing and results were of the device.

4.1 Product Functionality

For the components that we had, the prototype functioned as expected. Depending on the sensor readings of temperature, humidity and gas levels in the environment, the Nobi successfully displayed the correct information as well as the current state of the device. Since the prototype utilizes development boards, the size of the first product is relatively large compared to what was envisioned. The size of the device could have been limited with the use of 3D printing, however due to lack of skills in that area of expertise and lack of time, we had to keep the components contained in a large container. Although this was a downside, on the good side this model was a scaled up version of the expected final product result.

As mentioned, the casing of the initial prototype was not optimal. However, it was designed in such a way to look as similar to what the final product was visualized to be. One concern with the product is that it needed to be durable and weatherproof. As of now, the components of the prototype are encased within a solid tupperware container, with the solar panel as the roof of the device. Together the container and the solar panel seal the components of the device in order to protect the circuitry. However, although it is slightly shielded by the solar panel, the BME680 sensor is in direct contact with the environment. For prototyping means this is good enough, however it is expected the sensor will be protected from the environment whilst having the ability to access information from it.

4.2 System Testing and Results

In our prototype, two types of tests were conducted to verify if it was working as expected. It included testing for the sensors, and testing for certain algorithms (for calibration and choosing states). Make note that without access to a professional controlled environment, testing on the accuracy of the device was complicated as we weren't sure what we could compare it to.

4.2.1 Sensor Testing

Sensors were tested to not only verify their functionality, but to also understand how they would react to changes in the environment. For the temperature sensor, heat and cold sources were applied close to the sensor. For the humidity sensor, steam was used to see the rise. And finally, for the gas sensor, smoke from a fire was created to simulate the smoke of a wildfire. The fire was lit inside of a controlled, secure and fireproof container, then turned off by removing the oxygen within the container. That is how the smoke was made.

The next section will talk about state testing. Each state has a specific gas level reading. By testing out the gas sensor, we were able to understand the effect of little smoke to a lot of smoke in the vicinity of the device. With little smoke around, the metal oxide resistance would drop by a fifth of its ambient/normal gas level. With a lot of smoke around, the metal oxide resistance would drop by four fifths of its ambient/normal gas level.

4.2.2 Calibration and State Testing

In order to test the states of each device, the device needed to re-calibrate itself every time it was in a new environment. In the first prototype of the Nobi this feature was not added, and while showcasing the product, the gas threshold values for alert and danger state were much lower than the ambient/normal gas level.

The testing for the calibration and state changes were conducted simultaneously. The device was placed in three different areas all with very different gas readings of the ambient environment. As soon as the sensor got used to its environment (readings of the metal oxide resistance were minimally changing), then calibration of the device would begin. In all three areas, the calibration algorithm worked as expected. It had set the alert gas level to a fifth of the normal gas level reading, and the danger gas level to four fifths of the normal gas reading. When simulating for a wildfire by creating smoke in the proximity of the Nobi, the device in all three different locations acted accordingly to what was expected. Therefore not only was the calibration working, but the state changes were working under certain circumstances.

5 Business Analysis

For the business analysis, we focused on recording all the fixed costs and unit costs. With these, a break-even point analysis could be made and later on a return on investment calculation. After some research, we determined that fixed costs include lease, development costs, and salary as demonstrated in Table 10. Everything totaled up to be $61,000.

Table 10. Fixed Cost Break-down

Expense	Cost ($)
Rent	$3,000
Utilities	$1,000
Operating Expenses	$2,000
Development Costs	$35,000
Salary Expense	$20,000

To determine the unit cost, we researched the price of each component of the Nobi assuming 10,000 of each were bought. We also added the cost of a GPS tracker which we plan to implement in the future. This resulted in a total unit cost of $39.76 which is well below the $50 constraint. All the components' prices are shown below in the following table.

Table 11. Unit Cost Break-down

Components	Price/10,000 units
Solar Panel	$7.90
ATmega238p	$1.46
nRF51822 BLE	$2.05
BME 680	$7.40
Li-Ion Battery	$9.95
Voltage Regulator	$0.50
Charge Controller	$0.50
GPS Tracker	$10.00

Furthermore, we also calculated the amount of revenue we expect to get from each Nobi. With our unit price, we assumed a 65% profit for us, the retailers and the distributors. This means that we would sell our product to the retailers for $66 and the retailers would sell their product for $110. This results in a final retail price of $180.

With both the fixed cost, unit cost, and revenue a break-even analysis can be conducted. In order to do this, the following formula is followed:

$$Fixed\ Cost\ +\ Unit\ Cost\ *\ N\ =\ Revenue\ *N$$
$$\$61,000\ +\ \$40\ *\ N\ =\ \$180 * N$$
$$N\ =\ 436$$

Using the formula, we calculated that our break-even point is 436 units. This would mean that our total cost would be $78,440 and a revenue of $78,480. After that, the rest of the money earned would be profit.

5.1 Initial Investment

As mentioned earlier, the fixed costs result in $61,000 per month. In order to prevent any issues that may arise for development, the initial investment would be $180,000 which is about three times our cost. This would provide three months to develop, manufacture, and distribute our product for a profit.

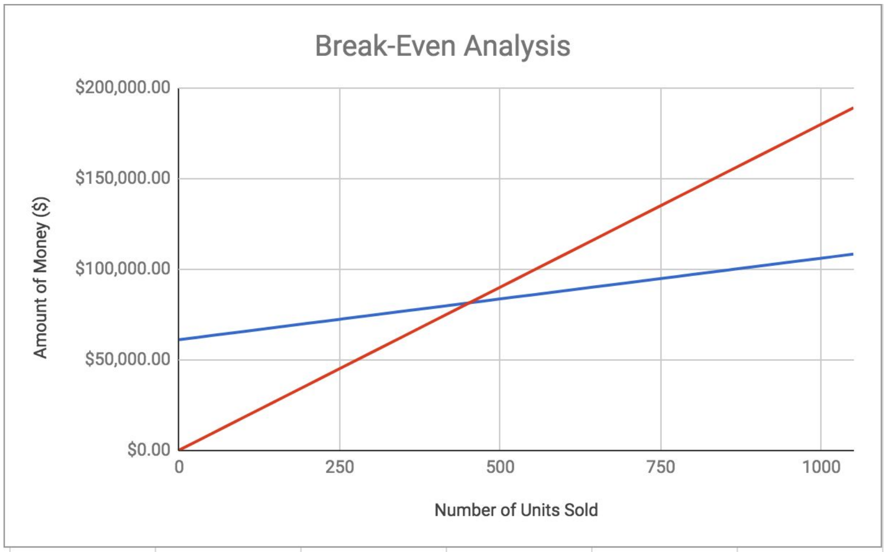

Figure 34. Break-even Analysis Graph

Our break-even point is at 436 Nobi units at a total cost of $78,440 and a revenue of $78,480. Therefore, with the initial investment, we would have enough money to develop that many units within the given time constraint. Assuming that we sold 300 - 500 units per month, we would be able to pay back our investment within three to five months.

5.2 Return on Investment

Return on investment (ROI) is often used to evaluate the efficiency of an investment. This means that it compares how likely it is to get the investment back. Naturally a high ROI is desirable for investors while a low one may ward off potential investors.

ROI is calculated using the following formula:

$$ROI = \frac{Revenue\ of\ Investment - Cost\ of\ Investment}{Cost\ of\ Investment}$$

Assuming we sold 10,000 units, our ROI would result in 290.46% which means that the investor would get their money back and more.

$$ROI = \frac{1,800,000 - 461,000}{461,000} = 290.46\%$$

6 Failure and Hazard Analysis

Any product in the market, no matter how well designed and researched, contains a possibility of hazard and failure.The Nobi device has both internal and external failures and hazards that will be mentioned.

6.1 Internal Failures and Hazards

Internally, the components for the prototype are temporarily encased in a tupperware to help against weather damages. However, it is not fully weatherproof. For the actual device, we intend to make it fully weatherproof and fireproof using a sealant on the casing to prevent any accidents that may occur and potentially harm the user. The components that make up the Nobi will also be heavily protected in case the device falls to the ground. In this case, the Nobi will have the ability to be reused if still meeting working standards.

Additionally, we will be making sure of the circuit's internal functionings are at the appropriate duty cycles and limits. This is to prevent any damage from heat; though our device is low powered so should have few heat issues.

Another failure that may occur is related to the solar panel as cloudy days may prevent it from charging the battery. If the cloudy days continue, there may be a risk of Nobi not having enough power to sustain itself. In order to mitigate this issue, we plan to include the charge and voltage values of the battery to be sent to watchtowers to keep track of potential Nobi failures.

6.2 External Failures and Hazards

One way that the Nobi can experience external risks and failures is through the mounting of the device. For the Nobi to function well, it needs to be durable and for it to be durable, and needs to be able to protect itself from harsh environmental conditions that might make it lose its support. The mounting of the Nobi is exceptionally important. If the mount isn't strong enough to support the weight, or can be easily affected by animal, human or weather tampering, then it can become a hazard to people or animals that are traversing the forest. Considering that the Nobi will be mounted on the top of the trees (or on the Nobi Towers), a falling Nobi can have a serious and dangerous impact on those that are below it. A solution to this issue would be to limit the size of the device, and to also research, design and develop a mount that is suitable for the placement of the Nobi and durable in any condition. However, in case that it falls, it should be made in such a way that it does not fatally wound anything in its way.

7 Recommendations

The design process is a long one and requires many iterations of a product until considered market-worthy. Likewise, we will be working on future prototypes of the Nobi. Some improvements we are thinking of is reducing the size of the Nobi. As of now it is around the size of a fist; however, we plan to design one to fit roughly inside an Altoids box (roughly a volume of 2”□2”□2”). Furthermore, shiny materials often attract animals which may be troublesome in the future. To prevent this, Nobi's future encasing should blend with the environment.

We also plan to make the hardware more compact. Instead of using the breakout boards, we will be using the actual components themselves, making the Nobi much smaller. PCBs will also be looked into and designed to make the Nobi easier to put together and overall expediting the process.

As we mentioned in earlier homeworks, we expected the customer to mount the Nobi themselves instead of providing the service. However, once development starts, we will focus on how to minimize the cost of the mounts so that we can mount the devices ourselves.

8 Conclusion

The Nobi now has a fully functioning prototype which we plan to improve upon in the future. As of now, we will be focusing on improving the components and versatility of the Nobi and making multiple prototypes in order to fully test the mesh network. Furthermore, we hope to implement a GPS tracker in the upcoming prototype and start on the data visualization that we mentioned earlier in the paper. Through these improvements, we hope to have a final product that will help prevent wildfires and mitigate the destruction caused by them.

Once the final product is developed, we we will consult a business mentor in order to launch this product into the market and simultaneously learn first-hand how to manage a business. Moreover, we plan to submit our product and relevant papers to the IEEE after meeting up with our advisors. We hope this product has the ability to spread world-wide and improve the lives of people all over.

Appendix A: Computer Code

Nobi Code

```
//////////////////////////BME680 SENSOR//////////////////////////////
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"

#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10

#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME680 bme; // I2C
//Adafruit_BME680 bme(BME_CS); // hardware SPI
//Adafruit_BME680 bme(BME_CS, BME_MOSI, BME_MISO,  BME_SCK);
//////////////////////////BME680 SENSOR//////////////////////////////

//////////////////////////BLUETOOTH DEVICE///////////////////////////////
#include <Arduino.h>
//#include <SPI.h>
#include "Adafruit_BLE.h"
#include "Adafruit_BluefruitLE_SPI.h"
#include "Adafruit_BluefruitLE_UART.h"

#include "BluefruitConfig.h"

#if SOFTWARE_SERIAL_AVAILABLE
  #include <SoftwareSerial.h>
#endif

#define FACTORYRESET_ENABLE         1
#define MINIMUM_FIRMWARE_VERSION    "0.6.6"
#define MODE_LED_BEHAVIOUR          "MODE"
```

```
/* ...hardware SPI, using SCK/MOSI/MISO hardware SPI pins and then user selected
CS/IRQ/RST */
Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_CS, BLUEFRUIT_SPI_IRQ,
BLUEFRUIT_SPI_RST);

// A small helper
void error(const __FlashStringHelper*err) {
  Serial.println(err);
  while (1);
}
/////////////////////////BLUETOOTH DEVICE//////////////////////////////
#include "floatToString.h"

//State delays
#define STANDARD_DELAY 300000
#define ALERT_DELAY 60000
#define DANGER_DELAY 2000

//States within the Nobi
enum nobiState {
  standard,
  alert,
  danger
};

nobiState state; // State of the Nobi Device
double alertGasVal = 40000;
double dangerGasVal = 10000;

//Finding average ambient gas level in environment,
//then calculating for the threshold values of alert
//and danger state
void callibration() {
  ble.print("------------\n");
  ble.print("Callibrating values\n");
  int calLength = 20;
  int iteration = 0;
  double gasCallibration[calLength];
```

```
  while (iteration < calLength){

    if (! bme.performReading()) {
    Serial.println("Failed to perform reading :(");
    return;
    }

    float sensorGasResistance = (bme.gas_resistance);
    gasCallibration[iteration] = sensorGasResistance;
//    Serial.print(iteration);
//    Serial.print(": ");
//    Serial.println(sensorGasResistance);

    iteration++;
    delay(3000);
    ble.waitForOK();
  }

  double sum = 0;
  for (int n = 0; n < calLength; n++){
    sum += gasCallibration[n];
//    Serial.println(sum);
  }

//  Serial.println();
  double avgGasVal = sum/calLength;
//  Serial.println(avgGasVal);
  alertGasVal = avgGasVal - avgGasVal*(3.0/10.0);
  dangerGasVal = avgGasVal - avgGasVal*(8.0/10.0);
  printGasValues();
  ble.print("Callibration Complete\n");
  ble.print("----------\n");
  delay(1000);
}


void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
```

```
  //while (!Serial);
  Serial.println(F("BME680 and Bluetooth Test"));
  Serial.println(F("---------------------------------------"));

  if (!bme.begin()) {
    Serial.println("Could not find a valid BME680 sensor, check wiring!");
    while (1);
  }
  nobiState state = standard;
  setBME680Sensor();
  setBluetoothDevice();

}

//Setting up the BLE
void setBluetoothDevice(){
  /* Initialise the module */
  Serial.print(F("Initialising the Bluefruit LE module: "));

  if ( !ble.begin(VERBOSE_MODE) )
  {
    error(F("Couldn't find Bluefruit, make sure it's in CoMmanD mode & check wiring?"));
  }
  Serial.println( F("OK!") );

  if ( FACTORYRESET_ENABLE )
  {
    /* Perform a factory reset to make sure everything is in a known state */
    Serial.println(F("Performing a factory reset: "));
    if ( ! ble.factoryReset() ){
      error(F("Couldn't factory reset"));
    }
  }

  /* Disable command echo from Bluefruit */
  ble.echo(false);

  Serial.println("Requesting Bluefruit info:");
  /* Print Bluefruit information */
```

```
  ble.info();

  Serial.println(F("Please use Adafruit Bluefruit LE app to connect in UART mode"));
  Serial.println(F("Then Enter characters to send to Bluefruit"));
  Serial.println();

  ble.verbose(false);  // debug info is a little annoying after this point!

  /* Wait for connection */
  while (! ble.isConnected()) {
      delay(500);
  }

  Serial.println(F("******************************"));

  // LED Activity command is only supported from 0.6.6
  if ( ble.isVersionAtLeast(MINIMUM_FIRMWARE_VERSION) )
  {
    // Change Mode LED Activity
    Serial.println(F("Change LED activity to " MODE_LED_BEHAVIOUR));
    ble.sendCommandCheckOK("AT+HWModeLED=" MODE_LED_BEHAVIOUR);
  }

  // Set module to DATA mode
  Serial.println( F("Switching to DATA mode!") );
  ble.setMode(BLUEFRUIT_MODE_DATA);

  Serial.println(F("******************************"));

}

//Setting up the BME680 sensor
void setBME680Sensor(){
  // Set up oversampling and filter initialization
  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setIIRFilterSize(BME680_FILTER_SIZE_3);
  bme.setGasHeater(320, 150); // 320*C for 150 ms
```

```
}

void loop() {
  //Get user input
  readBLEInput(); //Checking for user input. Used for debugging and manual calibration
  //~ = Calibration
  //+ = increased thresold state values
  //- = decrease threshold state values
  //1 = show state threshold values

  //Get BME reading
  getBME680Reading();
}

void readBLEInput(){
  ble.readline();
  if (strcmp(ble.buffer, "~") == 0){
    Serial.println("Calibrating!");
    ble.print("\nCalibrating\n");
    delay(20);
    callibration();
  }

  if (strcmp(ble.buffer, "-") == 0){
    Serial.println("Decreasing State Values");
    alertGasVal-=3000;
    dangerGasVal-=3000;
    printGasValues();
  }

  if (strcmp(ble.buffer, "+") == 0){
    Serial.println("Increasing State Values");
    alertGasVal+=3000;
    dangerGasVal+=3000;
    printGasValues();
  }

  if (strcmp(ble.buffer, "1") == 0){
    printGasValues();
```

```
  }

}

//Prints the threshold gas values for the alert
//state and the danger state
void printGasValues(){
    ble.print("---------\n");
    ble.print("AlertGasVal: ");
    ble.print(alertGasVal);
    ble.print(" kohm\n");
    delay(20);
    ble.print("DangerGasVal: ");
    ble.print(dangerGasVal);
    ble.print(" kohm\n---------\n");
    delay(20);
}

char buffer[10];

void printState(){
  if (state == danger){
    ble.print("Danger!\n");
  } else if (state == alert) {
    ble.print("Alert State\n");
  } else {
    ble.print("Standard State\n");
  }
}

//Sends sensor readings to the Bluetooth Device
void sendInfoViaBLE(float temp, float pres, float humid, float gasRes){

  printState();
  delay(20);
  ble.print("T = ");
  ble.print(floatToString(buffer, temp, 3));
  ble.print(" degC\n");
  delay(20);
```

```
  ble.print("RH = ");
  ble.print(floatToString(buffer, humid, 3));
  ble.print("%\n");
  delay(20);
  ble.print("Gas = ");
  ble.print(floatToString(buffer, gasRes/1000.0, 3));
  ble.print(" kohm\n");
  delay(20);

  // check response stastus
  if (! ble.waitForOK() ) {
    Serial.println(F("Failed to send?"));
  }
}

#define VBATPIN A9

String getBatteryVoltage(){
  float measuredvbat = analogRead(VBATPIN);
  measuredvbat *= 2;    // we divided by 2, so multiply back
  measuredvbat *= 3.3;  // Multiply by 3.3V, our reference voltage
  measuredvbat /= 1024; // convert to voltage
  String batVal = floatToString(buffer, measuredvbat, 3);
  return "VBat: "+batVal+" V";
}

void getBME680Reading(){
  if (! bme.performReading()) {
    Serial.println("Failed to perform reading :(");
    return;
  }

  float sensorTemperature = bme.temperature;
  float sensorPressure = (bme.pressure / 100.0);
  float sensorHumidity = bme.humidity;
  float sensorGasResistance = (bme.gas_resistance);

  Serial.println();
```

```
  //If it gas resistance reaches a dangerous ppm, then goes to alert delay
  if (sensorGasResistance <=dangerGasVal){
    state = danger;
    Serial.println("Danger!");
  }
  else if (sensorGasResistance <= alertGasVal){
    state = alert;
    Serial.println("Alert State");
  } else {
    state = standard;
    Serial.println("Standard State");
  }

  sendInfoViaBLE(sensorTemperature,sensorPressure,sensorHumidity,sensorGasResistance);
  ble.print(getBatteryVoltage());
  ble.print("\n\n");
  if (state == danger) delay(DANGER_DELAY/60);
  else if (state == alert) delay(ALERT_DELAY/60);
  else delay(STANDARD_DELAY/60);
}
```

BluefruitConfig.h

```
// COMMON SETTINGS
// ----------------------------------------------------------------------------------------------
// These settings are used in both SW UART, HW UART and SPI mode
// ----------------------------------------------------------------------------------------------
#define BUFSIZE                        128   // Size of the read buffer for incoming data
#define VERBOSE_MODE                   true  // If set to 'true' enables debug output


// SOFTWARE UART SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the pins that will be used for 'SW' serial.
// You should use this option if you are connecting the UART Friend to an UNO
// ----------------------------------------------------------------------------------------------
#define BLUEFRUIT_SWUART_RXD_PIN       9    // Required for software serial!
#define BLUEFRUIT_SWUART_TXD_PIN       10   // Required for software serial!
#define BLUEFRUIT_UART_CTS_PIN         11   // Required for software serial!
#define BLUEFRUIT_UART_RTS_PIN         -1   // Optional, set to -1 if unused
```

```
// HARDWARE UART SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the HW serial port you are using. Uncomment
// this line if you are connecting the BLE to Leonardo/Micro or Flora
// ----------------------------------------------------------------------------------------------
#ifdef Serial1    // this makes it not complain on compilation if there's no Serial1
  #define BLUEFRUIT_HWSERIAL_NAME      Serial1
#endif


// SHARED UART SETTINGS
// ----------------------------------------------------------------------------------------------
// The following sets the optional Mode pin, its recommended but not required
// ----------------------------------------------------------------------------------------------
#define BLUEFRUIT_UART_MODE_PIN        12    // Set to -1 if unused


// SHARED SPI SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the pins to use for HW and SW SPI communication.
// SCK, MISO and MOSI should be connected to the HW SPI pins on the Uno when
// using HW SPI.  This should be used with nRF51822 based Bluefruit LE modules
// that use SPI (Bluefruit LE SPI Friend).
// ----------------------------------------------------------------------------------------------
#define BLUEFRUIT_SPI_CS               8
#define BLUEFRUIT_SPI_IRQ              7
#define BLUEFRUIT_SPI_RST              4    // Optional but recommended, set to -1 if unused

// SOFTWARE SPI SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the pins to use for SW SPI communication.
// This should be used with nRF51822 based Bluefruit LE modules that use SPI
// (Bluefruit LE SPI Friend).
// ----------------------------------------------------------------------------------------------
#define BLUEFRUIT_SPI_SCK              13
#define BLUEFRUIT_SPI_MISO             12
#define BLUEFRUIT_SPI_MOSI             11
```

floatToString.h

```
// floatToString.h
//
// Tim Hirzel
// tim@growdown.com
// March 2008
// float to string
//
// If you don't save this as a .h, you will want to remove the default arguments
//     uncomment this first line, and swap it for the next.  I don't think keyword arguments compile
in .pde files

//char * floatToString(char * outstr, float value, int places, int minwidth=, bool rightjustify) {
char * floatToString(char * outstr, float value, int places, int minwidth=0, bool rightjustify=false)
{
   // this is used to write a float value to string, outstr.  oustr is also the return value.
   int digit;
   float tens = 0.1;
   int tenscount = 0;
   int i;
   float tempfloat = value;
   int c = 0;
   int charcount = 1;
   int extra = 0;
   // make sure we round properly. this could use pow from <math.h>, but doesn't seem worth the
import
   // if this rounding step isn't here, the value  54.321 prints as 54.3209

   // calculate rounding term d:   0.5/pow(10,places)
   float d = 0.5;
   if (value < 0)
      d *= -1.0;
   // divide by ten for each decimal place
   for (i = 0; i < places; i++)
      d/= 10.0;
   // this small addition, combined with truncation will round our values properly
   tempfloat +=  d;

   // first get value tens to be the large power of ten less than value
```

```
if (value < 0)
   tempfloat *= -1.0;
while ((tens * 10.0) <= tempfloat) {
   tens *= 10.0;
   tenscount += 1;
}

if (tenscount > 0)
   charcount += tenscount;
else
   charcount += 1;

if (value < 0)
   charcount += 1;
charcount += 1 + places;

minwidth += 1; // both count the null final character
if (minwidth > charcount){
   extra = minwidth - charcount;
   charcount = minwidth;
}

if (extra > 0 and rightjustify) {
   for (int i = 0; i< extra; i++) {
      outstr[c++] = ' ';
   }
}

// write out the negative if needed
if (value < 0)
   outstr[c++] = '-';

if (tenscount == 0)
   outstr[c++] = '0';

for (i=0; i< tenscount; i++) {
   digit = (int) (tempfloat/tens);
   itoa(digit, &outstr[c++], 10);
   tempfloat = tempfloat - ((float)digit * tens);
```

```
        tens /= 10.0;
    }

    // if no places after decimal, stop now and return

    // otherwise, write the point and continue on
    if (places > 0)
    outstr[c++] = '.';
    // now write out each decimal place by shifting digits one by one into the ones place and
writing the truncated value
    for (i = 0; i < places; i++) {
        tempfloat *= 10.0;
        digit = (int) tempfloat;
        itoa(digit, &outstr[c++], 10);
        // once written, subtract off that digit
        tempfloat = tempfloat - (float) digit;
    }
    if (extra > 0 and not rightjustify) {
        for (int i = 0; i< extra; i++) {
            outstr[c++] = ' ';
        }
    }

    outstr[c++] = '\0';
    return outstr;
}
```

References

1. (n.d.). The Rising Cost of Fire Operations - Forest Service. Retrieved December 15, 2017, from https://www.fs.fed.us/sites/default/files/2015-Fire-Budget-Report.pdf
2. (2016, October 10). Climate Change Blamed for Half of Increased Forest Fire Danger - The .... Retrieved December 15, 2017, from https://www.nytimes.com/2016/10/11/science/climate-change-forest-fires.html
3. (n.d.). Back to Basics with the Fire Triangle - Elite Fire Protection. Retrieved December 15, 2017, from http://www.elitefire.co.uk/basics-fire-triangle/
4. (2017, October 11). Wildfire property damage could reach $65 billion in ... - CNN Money. Retrieved December 15, 2017, from http://money.cnn.com/2017/10/11/news/economy/california-wildfire-cost-estimate/index.html
5. (2017, October 11). California's Wildfires - The New York Times. Retrieved December 15, 2017, from https://www.nytimes.com/2017/10/11/us/california-fires-questions.html
6. (2017, October 13). California fires: 36 dead as 'ashes and bones' found in ruins - CNN. Retrieved December 15, 2017, from http://www.cnn.com/2017/10/13/us/california-fires-updates/index.html
7. (n.d.). The Rising Cost of Fire Operations - Forest Service. Retrieved December 15, 2017, from https://www.fs.fed.us/sites/default/files/2015-Fire-Budget-Report.pdf
8. (2016, October 10). Climate Change Blamed for Half of Increased Forest Fire Danger - The .... Retrieved December 15, 2017, from https://www.nytimes.com/2016/10/11/science/climate-change-forest-fires.html
9. (n.d.). Attentis Technology - Advanced sensors, Intelligent wireless networks .... Retrieved December 15, 2017, from https://attentistechnology.com/
10. (n.d.). Wildfire Detection - Vigilys. Retrieved December 15, 2017, from http://vigilys.com/technology/firealert/
11. (n.d.). Wildland Fire Sensors Challenge | Challenge.gov. Retrieved October 31, 2017, from https://www.challenge.gov/challenge/wildland-fire-sensors-challenge/
12. (n.d.). What Are The Advantages of Wireless Communication? | It Still Works .... Retrieved October 31, 2017, from https://itstillworks.com/advantages-wireless-communication-1121.html
13. (n.d.). Wildfire Detection - Vigilys. Retrieved December 15, 2017, from http://vigilys.com/technology/firealert/
14. (n.d.). Insight Robotics - International Fire Fighter - MDM Publishing Ltd. Retrieved December 15, 2017, from https://iffmag.mdmpublishing.com/tag/insight-robotics/
15. (n.d.). Insight Robotics Wildfire Detection System | Insight Robotics. Retrieved December 15, 2017, from https://www.insightrobotics.com/services/wildfire-detection-system/
16. (2015, March 17). Insight Robotics Wildfire Detection System – International Fire Fighter. Retrieved December 15, 2017, from https://iffmag.mdmpublishing.com/insight-robotics-wildfire-detection-system/
17. (2010, September 4). Detecting Forest Fires using Wireless Sensor Networks with ... - Libelium. Retrieved December 15, 2017, from http://www.libelium.com/wireless_sensor_networks_to_detec_forest_fires/

18. (n.d.). FIT0330 DFRobot | Sensors, Transducers | DigiKey. Retrieved December 15, 2017, from https://www.digikey.com/product-detail/en/dfrobot/FIT0330/1738-1072-ND/6588494
19. (n.d.). Annual Days of Sunshine in California - Current Results. Retrieved December 15, 2017, from https://www.currentresults.com/Weather/California/annual-days-of-sunshine.php
20. (n.d.). California, USA Map Lat Long Coordinates. Retrieved December 15, 2017, from https://www.latlong.net/place/california-usa-2487.html
21. (n.d.). Optimum Tilt of Solar Panels. Retrieved December 15, 2017, from http://www.solarpaneltilt.com/
22. (n.d.). PV Modules - Solar Direct. Retrieved December 15, 2017, from http://www.solardirect.com/pv/pvlist/pvlist.htm
23. (n.d.). Short-Circuit Current | PVEducation. Retrieved December 15, 2017, from http://www.pveducation.org/pvcdrom/short-circuit-current
24. (2010, September 8). L7805CV. Retrieved December 15, 2017, from https://cdn-shop.adafruit.com/product-files/2164/L7805CV.pdf
25. (n.d.). USB-Compatible Lithium-Ion Battery Charger with ... - RobotShop. Retrieved December 15, 2017, from http://www.robotshop.com/media/files/pdf2/datasheet-dfr-565.pdf
26. (n.d.). Electronic Components and Parts Search .... Retrieved December 15, 2017, from https://www.digikey.com/products/en
27. (n.d.). BME680 Datasheet - Adafruit Industries. Retrieved December 15, 2017, from https://cdn-shop.adafruit.com/product-files/3660/BME680.pdf
28. (n.d.). BME680 - Bosch Sensortec. Retrieved December 15, 2017, from https://www.bosch-sensortec.com/bst/products/all_products/bme680
29. (n.d.). Adafruit BME680 - Temperature, Humidity, Pressure and Gas Sensor .... Retrieved December 15, 2017, from https://www.adafruit.com/product/3660
30. (n.d.). Adafruit Learning System. Retrieved December 15, 2017, from https://learn.adafruit.com/assets/48064
31. (n.d.). Climate California - temperature, rainfall and average - US Climate Data. Retrieved December 15, 2017, from https://www.usclimatedata.com/climate/california/united-states/3174
32. (n.d.). Adafruit Learning System. Retrieved December 15, 2017, from https://learn.adafruit.com/assets/28634
33. (2015, November 18). Overview | Adafruit Feather 32u4 Bluefruit LE | Adafruit Learning System. Retrieved December 15, 2017, from https://learn.adafruit.com/adafruit-feather-32u4-bluefruit-le/overview
34. (n.d.). ATmega16U4/32U4 Datasheet. Retrieved December 15, 2017, from http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf
35. (n.d.). nRF51x22 PS.book - Nordic Infocenter - Nordic Semiconductor. Retrieved December 15, 2017, from http://infocenter.nordicsemi.com/pdf/nRF51822_PS_v3.1.pdf
36. (2009, August 9). How fast does fire spread - YouTube. Retrieved December 15, 2017, from https://www.youtube.com/watch?v=fsaLCdC3iWw

37. (n.d.). Fire Spread and Fire Drills | Grand Traverse Metro Emergency .... Retrieved December 15, 2017, from https://www.gtmetrofire.org/public-education/safety-information/fire-spread-and-fire-drills/
38. (2016, February 17). How Fast Does Fire Spread? | SkySaver Rescue Backpacks. Retrieved December 15, 2017, from https://skysaver.com/blog/how-fast-fire-spread/
39. (n.d.). Anatomy of a Tree - Arbor Day Foundation. Retrieved December 15, 2017, from https://www.arborday.org/trees/treeguide/anatomy.cfm
40. (n.d.). Recovering from Wildfire: A Guide for California's ... - ANR Catalog. Retrieved December 15, 2017, from http://anrcatalog.ucanr.edu/pdf/8386.pdf
41. (n.d.). How to Build a Tree House — Building Tips — The Family Handyman. Retrieved December 15, 2017, from https://www.familyhandyman.com/garden-structures/tree-house-building-tips/view-all/
42. (n.d.). Data Visualization - Visual Analytics | What is Data ... - Oracle. Retrieved December 15, 2017, from https://www.oracle.com/solutions/business-analytics/data-visualization.html
43. (2017, May 10). Heat Up Your Maps with Oracle Data Visualization - Oracle Blogs. Retrieved December 15, 2017, from https://blogs.oracle.com/analyticscloud/heat-up-your-maps-with-oracle-data-visualization