

WHERE2PARK: AN INTERNET OF THINGS APPROACH TO MANAGING CARBON EMISSIONS

A Research Paper submitted to the Department of Electrical and Computer Engineering
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Engineering

By
Cameron Davis

December 10, 2020

Technical Project Team Members
Gunther Abbot, Sean Reihani, Nawar Wali

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

ADVISOR
Harry C. Powell, Department of Electrical and Computer Engineering

Where2Park: A Modern Urbanization Solution

Gunther Abbot, Cameron Davis, Sean Reihani, Nawar Wali

December 10, 2020

Capstone Design ECE 4440 / ECE4991

Signatures

Gunther Abbot



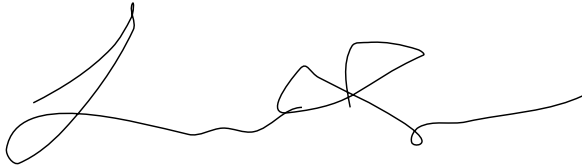
12/10/2020

Cameron Davis



12/10/2020

Sean Reihani



12/10/2020

Nawar Wali



12/10/2020

Statement of work:

Gunther Abbot: I took the lead on the board layout and design of the phase-locked loop subsystem of our sensor node. The goal of the phase-locked loop is to convert frequency information into voltage information, which can then be interpreted by the ADC. This frequency comes from the oscillator circuit and allows our sensor node to detect the presence or absence of a vehicle. We used the industry-classic CD4046 phase-locked loop, which came with plenty of resources for designing frequency-to-voltage translators. Our first major roadblock came when we discovered we had not properly conditioned the input signal to the phase-locked loop to be within CMOS logic swing levels, as the datasheet required. This required a small board hack consisting of a voltage divider and a blocking capacitor to bias the oscillator signal up to the proper threshold. This was successful, and for a time we were observing proper voltage scaling on the phase-locked loop output. Upon performing some clean-up and trim-back of the components on our board, we experienced an excessive current draw in the oscillator stage that dragged down the source battery voltage. This could possibly be due to a poor solder joint in the floating components on our board in the area of the board modification. The current draw drew the source voltage down to such an extent that the oscillator signal was no longer within the CMOS logic threshold of the phase-locked loop, and the phase-locked loop no longer produced meaningful output.

Cameron Davis: For the project, I was responsible for creating the software architecture and working on the desktop application will serve as the interface to visualize the data. The desktop application presented a challenge in trying to read the data from the Zigbee nodes and the coordinating updates that were shown in the GUI. The application was designed to show the structure where the system was implemented and the available parking spots will be displayed. Originally, I started building the application using the Nana C++ library but I was unable to succeed in integrating UART communication. After running into this roadblock, I switched to using Java to try to implement the functionality. There were few resources online to work with UART using Java and they were contained in a deprecated API. This required implementing an interface for which I had to write each of the API functions based on the Java Digital I/O library since no existing library was available for use.

Sean Reihani: My responsibilities were the firmware and network development. The bulk of my work came down to studying TI's Zigbee Stack, Z-Stack. The first major hurdle was that we purchased a deprecated set of LaunchPads that could not handle our application's needs, a fact that was not stated anywhere on the project page. This resulted in a notable delay, but not a catastrophic one by any means. The majority of my time went to figuring out Z-Stack. The provided sample code was large and verbose, and contained almost no comments. The online labs helped me start, but did a poor job explaining fundamental concepts. Indeed, I learned through online internet forums that this software was intended for professionals. Fortunately, online resources such as TI's E2E forum helped me figure out the stack. A notable issue that the sent payload did not equal the parsed payload on the receiving end. Our only means of debugging this was through UART prints; I could not view the incoming message struct's member variables in the debugger, and worse, the TI packet sniffer would not work, preventing us from sniffing the over-the-air data. Fortunately this was not an issue, as we only needed to

“ping” the coordinator, and packets were successfully transmitted and received. Overall, the embedded code performed the core functions correctly and reliably (however, network joining did require several attempts). However, the unpredictable packet behavior would have impeded battery level transmission were we still supporting that feature.

Nawar Wali: My responsibilities include designing and testing the metal detector schematic as well as planning the detector coil. The circuit has a few different parts and has to take into account the output/feeding into the microcontroller. The first part of the schematic is the Colpitts Oscillator where the main feature is a tank circuit. A tank circuit is a theoretical circuit where an inductor and capacitor keep each other running. The Colpitts oscillator uses a tank circuit to provide a frequency at a chosen level. The reason why a metal detector uses a Colpitts oscillator is that Colpitts also provides a distortion signal that can be measured. When the metal comes in range of the inductor, the inductance changes which in turn changes the frequency of the system. The inductor coil is chosen to be 5.5mH. This can be created by 143 turns on a 5” diameter former using 0.56mm enamelled wire, which I had originally created. While I was testing the system, the coil response was unusable so I decided to adapt the wire to the current parts and circuitry. I kept the resistors and capacitors the same and changed the wire to 76 turns on a 4.03” diameter. The new wire has an inductance of about 2.4mH. I kept the resistors and capacitors the same because another hardware component that we used/designed had an acceptable range of frequencies that the coil was able to display.

Table of Contents

Capstone Design ECE 4440 / ECE4991	1
Signatures	1
Statement of work:	2
Table of Contents	3
Table of Figures	5
Abstract	6
Background	6
Design Constraints	7
External Standards	7
Tools Employed	8
Ethical, Social, and Economic Concerns	8
Intellectual Property Issues	9
Detailed Technical Description of Project	10
Project Timeline	18
Test Plan	19
Final Results	20
Costs	22
Future Work	22
References	22
Appendix	25

Table of Figures

Figure 1: Diagram of Zigbee Mesh Network	11
Figure 2: Overall Schematic	12
Figure 3: Oscillator Subsection	12
Figure 4: MCU Voltage Regulator Subcircuit	!Error! Bookmark not defined.
Figure 5: Low Pass Subcircuit	!Error! Bookmark not defined.
Figure 6: Transient Analysis of PLL Oscillator	!Error! Bookmark not defined.
Figure 7: Diagram of the Sensor Node	15
Figure 8: Flowchart for Desktop Application Operation	16
Figure 9: Graphical User Interface	17
Figure 10: Gantt Chart - Original Projected Timeline	18
Figure 11: Hardware Test Plan	19
Figure 12: Software Test Plan	20
Figure 13: Base Oscillating Frequency	21
Figure 14: Oscillating Frequency in the Presence of Metal	21
Appendix A: Total Project Cost	25
Appendix B: Estimated Cost of Producing 1000 Units	26

Abstract

Where2Park is a battery-powered Internet of Things (IoT) project aimed for deployment in parking garages. Where2Park relies on a series of sensor nodes embedded in parking spots. The nodes will feature metal detecting sensors and interfaces with a microcontroller. The nodes are designed to be part of a Zigbee mesh network, and each node will be capable of being configured as either an end device, router, or a coordinator. The network was designed to interface with a centralized application that features a map of the parking spaces that indicates their availability. The graphical user interface (GUI) will update in real time. This project is being developed in the context of a stark increase of IoT nodes and the rise of the “smart city.” In addition to the desired integration into smart cities, the end goal of the project is to aid users in efficiently finding parking spaces in the face of growing urbanization.

Background

Parking in highly populated urban centers is widely regarded as a nuisance and inefficient. With the US population expected to reach 400 million by 2060[1], whereas the global population is expected to reach 10.2 billion by that same year[2] (i.e., a 70 million and a 2.6 billion increase, respectively). Urbanization is also vastly increasing. From 1950 to 2018, the urban population increased from 751 million to 4.2 billion. The urban population is expected to increase by 2.5 billion people by 2050, during which 68% of the world’s population will inhabit urban areas[3]. Additionally, IoT is on track to become a central pillar of urban development, as a 2015 projection expected cities to spend \$41 trillion on IoT across the following two decades[4]. Thus, with growing urban populations in the United States and around the world, it would be a worthwhile venture to expedite the parking process in such a manner that can be integrated into the rapidly expanding IoT urban infrastructure.

One prior implementation of a smart parking lot sensor is the Bosch Parking Lot Sensor. This device has many features similar to our own design, but one notable difference is the use of LoRaWAN protocol as opposed to Zigbee[5]. One deficiency of LoRaWAN is that it only allows packet transmission at a maximum rate of once every couple of minutes, while Zigbee will allow for real time updates. Zigbee also maintains superior reliability. The Zigbee protocol includes built-in packet acknowledgement, meaning that any failed packets are automatically resent. LoRaWAN does not include this feature by default, and can have relatively high packet loss rates. Additionally, LoRaWAN is a Star Network, whereas Zigbee is a Mesh Network that possesses the capability to self-heal, i.e. if a node on the network fails, other nodes will automatically reroute[6]. LoRaWAN is typically used for long-range applications. Zigbee is sufficient for this case, as traffic is routed between sensor nodes, and sensors would be deployed in close proximity to one another and in high density. A centralized computer system (for the purposes of this project, the laptop running the user interface) could connect to the internet via WiFi or cellular and upload parking data to the cloud. Peripherals may also be used to extend the nodes’ range. In addition, the mesh network capabilities of our solution allow for easy and inexpensive scaling.

The skills used to complete the project include:

- Analog circuit design and analysis
- PCB layout
- Embedded software development
- Understanding of networking concepts and protocols
- Software engineering principles

All group members have experience (through classes, projects, research, internships, etc.) in all of these domains, thus allowing each group member to be acutely aware of the others' progress and requirements, and help debug as needed. Sean has completed coursework involving networking and embedded development (Introduction to Embedded Systems, Real Time Embedded Systems, Operating Systems, Computer Networks) and successfully implemented an embedded solution as part of a research project, he has primarily focused on the firmware aspect of the project. In particular, he was involved in the networking and integration with the desktop application. Cameron has taken several classes that provide him a foundation in the principles of software development, software architecture and understanding how these devices work together (Advanced Software Development, Computer Networks, Computer Architecture) and is responsible for the development of the desktop application interface. Gunther, similar to Sean, has class experience in low level software development (Real Time Embedded Systems, Operating Systems) along with internship experience in embedded systems. He enjoys the heat of the soldering iron, which led him to take the lead in PCB layout and phase-locked loop design. Nawar has taken several courses (Solid State Devices, Computer Architecture, Advanced Software Development and Hardware and Software Security) that provide a portfolio of hardware principles and is responsible for the metal detector coil, the metal detector and PCB design along with Zigbee implementations.

Design Constraints

Our project design accounted for the University's funding of \$500 towards our project. The majority of our budget went towards our printed circuit boards (PCB) and the TI SimpleLink boards. Other materials we accounted for include the materials to make our wire metal detector coils, wood, resistors and other electrical components. The limited board sendouts and time between them were also a constraint. The itemized budget can be found in the cost section.

External Standards

When creating a product that is going to be designed for public use, the product should be up to safety standards. The safety standards that apply to our project is the voltage levels of the oscillator to the launchpad. The NEMA safety standard uses various grades of electrical enclosures typically in industrial applications. Each has ratings on hazardous parts and additional type-dependent designated environmental conditions [8].

The IEEE standards have been developed to have green and clean technologies to protect and improve the quality of the environment and the earth. The application to our project is the soldering aspect of the circuit and the battery life of the application. The battery life is monitored

by a separate circuit and will be replaced properly. The batteries have alkaline and need to be disposed of properly[9].

The code of ethics is another standard that needs to be followed as it allows the access to consumers in a safe manner. As engineers, we should be expected to show the highest standards of honesty and integrity. Engineering has an impact on society and can change based on the criticism of people. This impacts our project since we have to make sure that the application must adhere to the highest principles of ethical conduct[10].

Another code that we had to follow is the Electronic Code of Federal Regulations. The Electronic Code of Federal Regulations has specific details regarding the bandwidth of certain circuitry. “ Frequency hopping systems shall have hopping channel carrier frequencies separated by a minimum of 25 kHz or the 20 dB bandwidth of the hopping channel, whichever is greater. Alternatively, frequency hopping systems operating in the 2400-2483.5 MHz band may have hopping channel carrier frequencies that are separated by 25 kHz or two-thirds of the 20 dB bandwidth of the hopping channel, whichever is greater, provided the systems operate with an output power no greater than 125 mW”[12].

Tools Employed

- Eclipse is a Java integrated development environment that was used to create the GUI.
- Multisim is a circuit simulator that helped us to test the theoretical circuit relationships. The connectivity, shorts and etc were helpful in the testing process. While testing on multisim, we determined the parts that we used.
- Ultiboard is a tool that helped us to develop from the base circuit level to be able to transform to a PCB.
- VirtualBench is a tool that helped us to simulate the components physically rather than virtually.
- Tektronix TDS 2014 is an oscilloscope with more channels than the VirtualBench, which allowed us to observe every stage of the sensor node PCB at once.
- Code Composer Studio is a C++ IDE used for compiling and debugging embedded software
- RealTerm was the terminal that allowed us to view UART data in both ASCII and Hex

Ethical, Social, and Economic Concerns

Environmental Impact

The sensor, were it to go into production, would have to be embedded into the concrete and parallel to the parking spot, or alternatively encased in a hard resin at the center of the parking spot. The system may require a significant amount of power. Our original intention was to power the nodes with batteries. Batteries need to be appropriately thrown away as they may contain alkaline which may be harmful to the environment.

Sustainability and Scalability

Each sensor module is powered with an alkaline or lithium-ion battery. This means a replacement battery will be needed after a certain period of time to keep the system working long term negatively affecting the sustainability of our project[6]. An economic constraint might be the GUI access. Many people may not have roaming data or even regular access to the internet. Given the common hardware shared by the nodes, the network size can easily be scaled. Zigbee mesh networks can contain hundreds of nodes at a time.

Manufacturability

Our project was impacted by the cost and availability of our selected materials and sensors. We had several cases of low availability of the capacitors and inductor backups. In the long run, the inductor backups weren't needed so we didn't have a setback regarding that.

Ethical Issues

The user will be interacting with the software based GUI so there are not significant health risks associated with our end product. Ethical issues associated with our product include protecting end user privacy issues since the system tracks specific parking spots [8]. Remote access to parking data could possibly facilitate stalking or robbery, but the ambiguity of the data provided by the system negligible, especially given that the same dangers can come from visual investigation on the part of nefarious parties.

Intellectual Property Issues

Looking through the requirements for patents and going through existing technology helped us to determine that we can make the argument to patent but it might be unapproved. We were able to get in touch with a patent examiner in the electrical department to talk through the patentability and he said that we could make an argument but that it might be hard to show a new solution.

We had actually found this patent in our searches for inspiration, this is the *Method and device for assisting the driver of a motor vehicle*[13]. The basis is that it assists the driver during the parking process and then it detects if the spot is open and if there are any non-moving objects in the vicinity. It claims to record the surrounding data to determine whether or not to park in that spot. In regards to our project, we made sure to be cognizant of the potential privacy issues and security of our system and have no user data collection methods.

One patent that is similar in practise to our project is *Navigating a Customer to a Parking Space*[14]. The basis of the project is that the user sends a parking request from a mobile device to a specific store computer. The store computer then uses camera systems, sensor systems, location of customer mobile devices and point-of-sale activity to determine a spot for the user and sends directions to the spot via customer mobile device. It claims that the computer system will use system memory and processors to navigate the user. This patent however, was abandoned. Our project is similar in idea but in practise is a bit different. We use more hardware aspects rather than software comparatively. We also make notice of the user's private data and do not use anything more than a car over a detector.

Another patent is *Cooperative Parking*[15]. The project involves a car that has a transmitter to signal to a parking spot when the proximity is near. The sensor is a position sensor mounted on the parked vehicle. It then activates the engine or auxiliary motor, disengages any locking mechanism such as brakes or transmission and "parks" the car. It also claims that while in parking mode, depending on the car, the parking itself is manually done. This patent goes beyond our project in that it modifies the car itself to park. The general idea is there but it does go beyond.

Detailed Technical Description of Project

Where2Park is an IoT system that tracks the status of parking spots in real time, and conveys this information to a central application that in turn conveys this information to end users. The system employs a fleet of vehicle-detecting sensor nodes, one per parking spot, which form a mesh network that relays information about each spot (i.e., the presence/absence of a car, node battery level) back to a central computer system. This central computer provides a front-end user interface that, combined with pre-set metadata such as location, allows the user to see exactly which parking spots are vacant in real time via a graphical user interface.

Networking and Firmware

Nodes communicate with one another using Zigbee technology, an 802.15.4 based specification for low data rate, wireless, personal area networks. The sensor nodes can form a mesh network using Zigbee, which allows for greater practical range than a star network topology.

Coordinator nodes create and manage a network in a process called commissioning, in which network information is flooded into the network. Once the network is created, a process called network steering commences, in which new devices are found that can join the network. Next "finding and binding" occurs; devices with matching Zigbee clusters are found, and addressing information is found and stored in binding tables.

Z-Stack builds upon TI's RTOS. The application has a task function that contains initialization code. After initialization, an infinite process loop is triggered. The RTOS timer is used to periodically trigger an event that polls the ADC. If the voltage difference is more than 0.1 V (the previous value being first captured during init and then stored in a global variable), a raw data request is initiated (i.e. a series of chars is sent to the coordinator, whose short address is always 0x0000). The data is then parsed on the Zigbee coordinator in the function `zclGenericApp_processAflncomingMsgInd`. The current parking state is stored in the coordinator. If there is a change, this is stored and the appropriate message is outputted through UART.

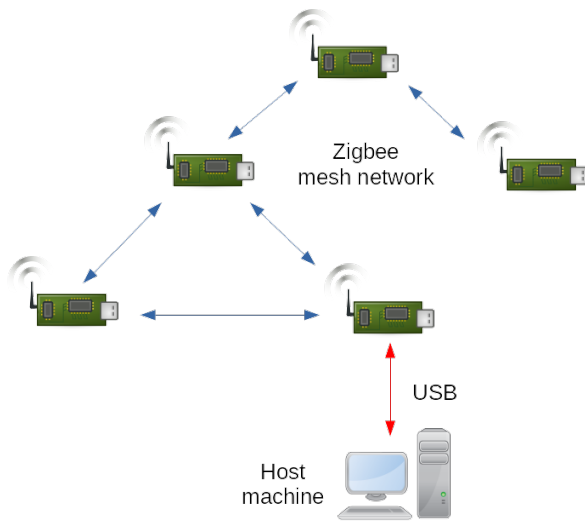


Figure 1: Diagram of Zigbee Mesh Network

Metal detector

To detect the car in the respective spot, a specific type of detector must be used. If an ultrasonic sensor is used, soiling could result in the misidentification of a parked car. To prevent

common deviations, we decided to use a metal detector. The Colpitts oscillator can be described as a tank circuit with a BJT inverting amplifier. The oscillator can provide frequencies up to the 400kHz range which can support detecting cars. The circuit has a few different parts and has to take into account the output/feeding into the microcontroller. A series of a capacitor and inductor forms a tank circuit. A tank circuit is where energy is transferred continuously between an inductor and capacitor, which results in oscillations. The current flow provides a change in polarity that will allow magnetic fields to form and collapse. Ideally, the inductor would be the metal detecting coil which would oscillate near metals without a power source. To offset this, a BJT will provide continuous gain. The inductor coil is chosen to be 5.5mH. This can be created by 143 turns on a 5" diameter former using 0.56mm enamelled wire. The baseline frequency was chosen to be at 100kHz roughly. Given that L is 5.5mH,

$C_T = \frac{1}{fL} = 445\text{pF}$. Given that capacitors sometimes operate on different harmonics rather than ideally, $C_2 = 2C_1$ and $C_1 = 4C_T/3$. $C_2 = 2.2\text{nF}$ and $C_1 = 560\text{pF}$.

The next part of the metal detector is the connection to the microcontroller. The issue is that the microcontroller has a sample rate of 200kS/s. The oscillator in the face of metals can have a frequency range of 200KHz to 400KHz. The plan to solve this problem is to feed it into a low pass filter and then into a comparator. The cutoff frequency for the low pass filter is 150kHz and thus can tell us that if it goes through the filter, that the circuit has come into contact with the coil.

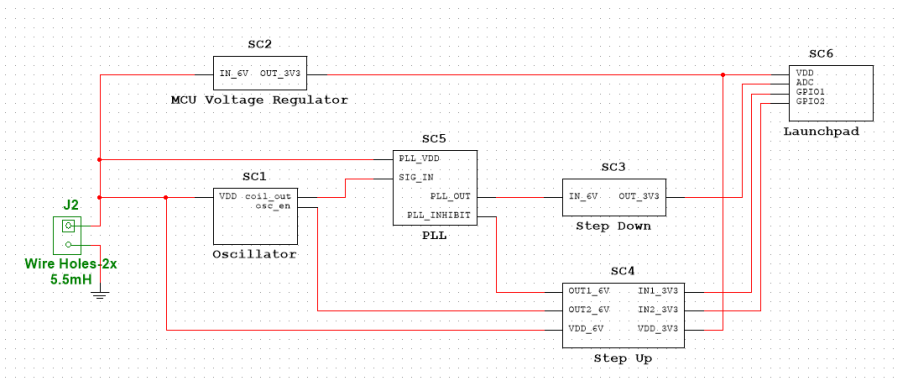


Figure 2: Overall Schematic

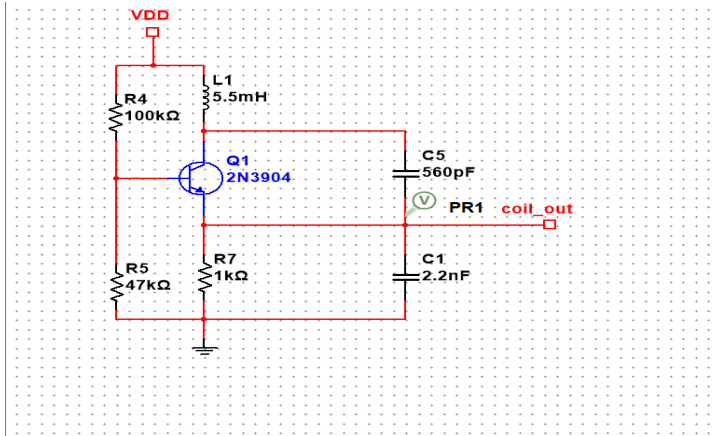


Figure 3: Oscillator Subsection

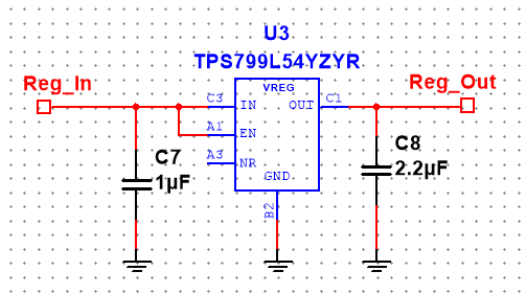


Figure 4: MCU Voltage Regulator Subcircuit

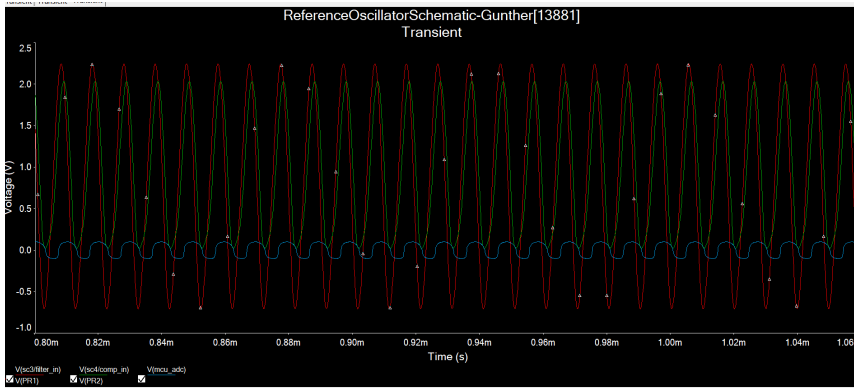


Figure 6: Transient Analysis of PLL Oscillator

While testing the system, it was shown that the coil did not show the response that was needed. Therefore, the coil was then switched with another one which was 76 turns on a 4.03” diameter. The new wire has an inductance of about 2.4mH. The PLL will still be able to handle the frequency difference as we had actually overestimated with the previous coil. It will be explained more in depth below.

The formula for the coil are as follows:

$$L = \frac{\text{Radius}^2 \times \text{Turns}^2}{9\text{Radius} + 10 \text{Length}}$$

Phase-Locked Loop

The phase-locked loop used was the CD4046 using the wideband Phase Comparator 2. The low-pass filter section of the loop was designed with a center frequency of 230kHz, which is roughly between the frequency of the coil with and without metal in proximity. The PLL is designed to lock on any transient frequency from the metal detector and translate it to a voltage ranging from 0V to 6V. Output is taken from the low pass filter stage and fed through a voltage divider which steps the range down to range from 0V to 3V. This is an appropriate input to the microcontroller’s ADC, where this DC voltage is read and interpreted to mean that either metal or no metal is present.

Sensor Node

The metal detector has a power supply of 4.5V in the form of 3 AA batteries with bypass capacitors. To make sure the current flow to the microcontroller stays at approximately 40mA, a sensitivity potentiometer is set up as a voltage divider and to read the division, an analog pin will be used.

The purpose of the sensor node is to interpret the vacancy of its parking spot. It consists of the metal detecting coil, battery, Zigbee antenna, and microcontroller that manages the node's power supply and interfaces with the mesh network.

The microcontroller used on the sensor node is a TI SimpleLink Cortex-M device with built-in Zigbee functionality and multiple analog to digital converters. This part was chosen due to its integration with Zigbee and low-power architecture which are characteristics well suited for battery-powered wireless sensor nodes. Two analog to digital converters will be used on the microcontroller, one for monitoring the voltage of the battery and another for reading from the metal.

Each node will be identical from a hardware standpoint, i.e. any node can serve as a router, coordinator, or end-device. This serves two purposes: scalability and reliability. In regards to the former, if new parking spaces are added to a lot, then installing new sensor nodes is a trivial matter. In regards to the latter, if one device fails, then another can take over. This means that if a coordinator/end-device fail, then a sensor node in router configuration can take its place, and if a router fails, traffic can be rerouted through other nodes.

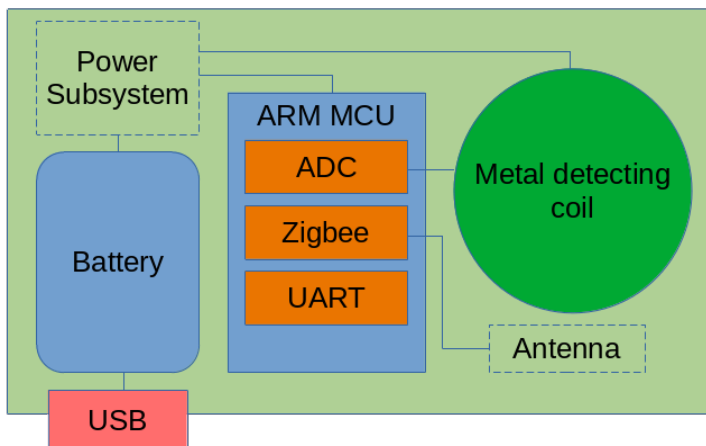


Figure 7: Diagram of the Sensor Node

Desktop Application

The desktop application was the interface to visualize the data. The application was designed to connect to a Zigbee node configured as a coordinator. The other Zigbee nodes, fashioned as coordinators, will send signals so that the system can manage and interact with the number of available parking spots. Figure 8 shows the planned implementation to connect the GUI to the receiver node. The application was designed to show where the system is located and the available parking spots would be displayed. The network coordinator will connect to our computer using a USB and functioning on UART communication protocol. In order to use UART, we needed to develop an API specific for this project. The application was designed using the Java Swing library, a cross platform library for GUI programming. [11] This platform was intended to allow us to seamlessly integrate with our chosen microcontroller and produce an easily usable interface available for the user shown in Figure 9.

Flowchart Outlining the Operation of the Desktop Application

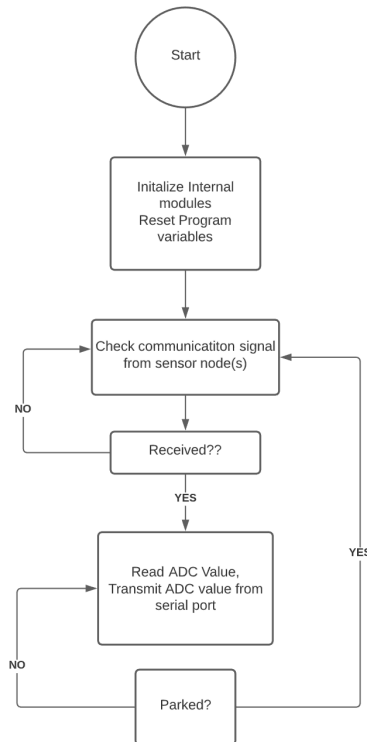


Figure 8: Flowchart for Desktop Application Operation

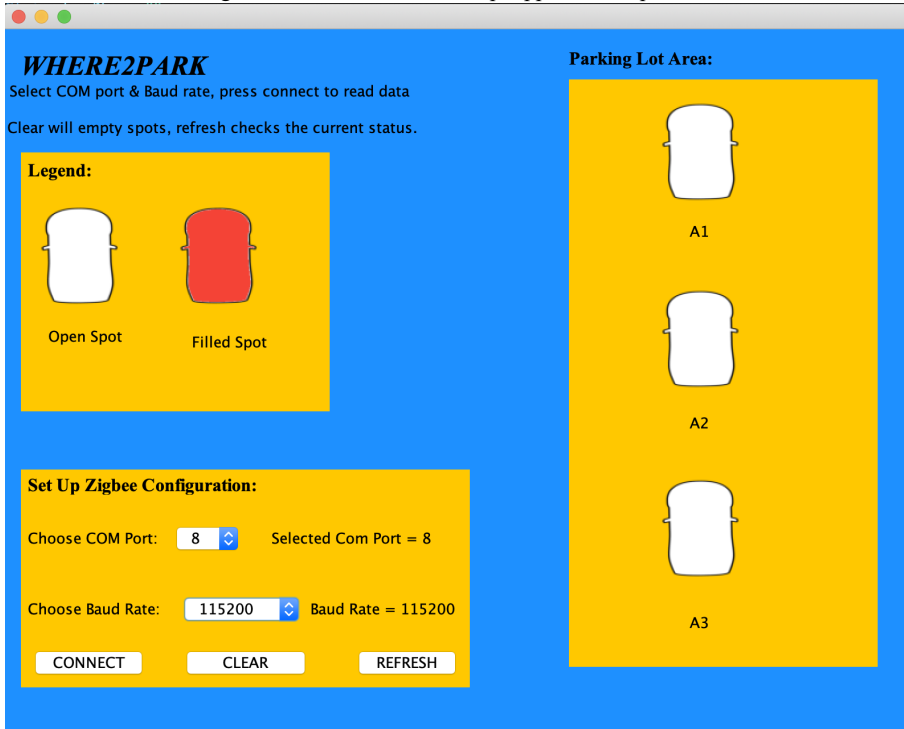


Figure 9: Graphical User Interface

Project Timeline

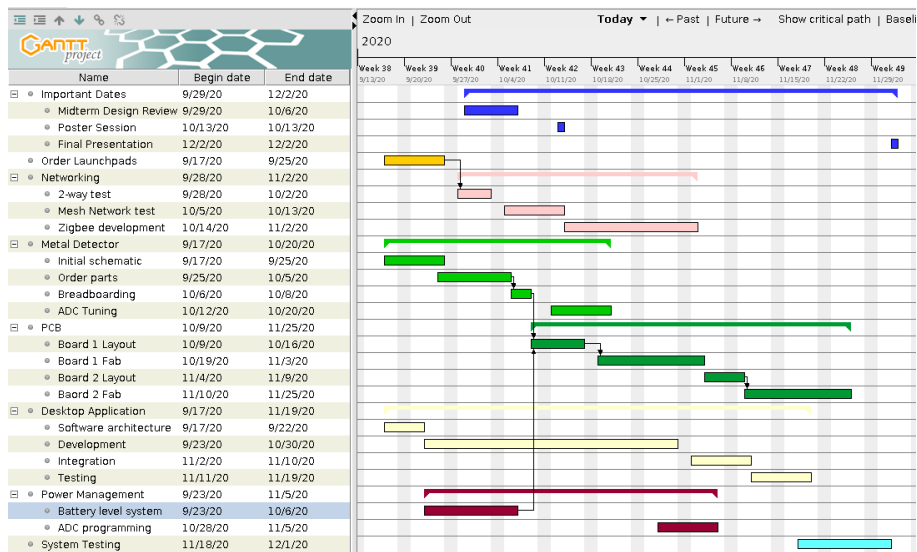


Figure 10: Gantt Chart - Original Projected Timeline

The Gantt chart above represents the estimated project timeline at the beginning of our project. The main components of the project include the Desktop Application, Metal Detector, Networking and Power Management. The software components were developed in parallel to the hardware. The initial steps of the project include the hardware aspects such as Metal Detector and the PCB fabrication.

The group approach to the project was dividing up the tasks into subsections where one person takes the lead. However, over-compartmentalization caused in part by the extenuating circumstances of the COVID-19 pandemic led to the team performing inadequate technical integration, contributing to many issues.

The team did not always follow the Gantt chart timeline. For example, a misunderstanding resulted in the purchase of the incorrect processors, which caused a delay in firmware development. An unexpected lack of documentation produced further delays, and the network was not functioning properly until December.

Test Plan

To test the hardware aspects, we would need to check if the metal detector actually detects metals. Since the inductor was created from a coil, we can simply wave a piece of metal over the coil and try to detect the frequency changes. While waving a large metal screw-like product, we were able to determine that the coil and oscillator work. The next step is to test the coil in accordance with the other hardware components that take the output to the launchpad, such as the PLL. When testing the PLL and other components with the oscillator, the output determined that the hardware system works.

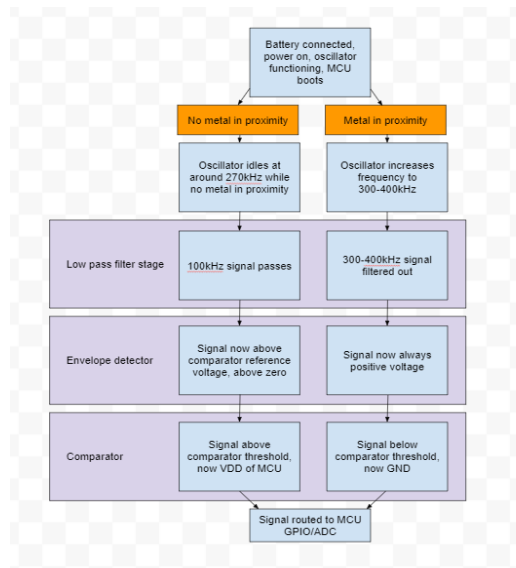


Figure 11: Hardware Test Plan

To test the embedded and software, we had to determine if the launchpad GUI can properly communicate over UART. We then had to test if the complete system works together. The expected outcomes of the project include a successful interface between sensor nodes and host machine as well as proper calibration of ADC and successful detection of a vehicle.

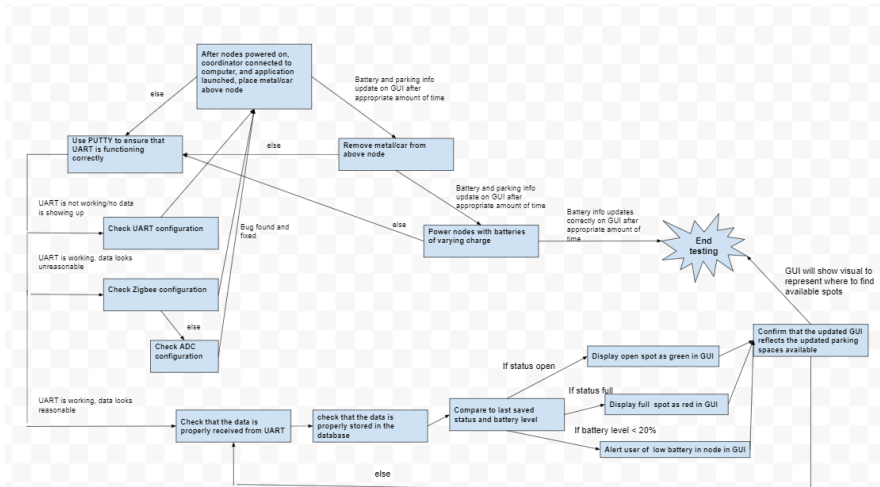


Figure 12: Software Test Plan

Final Results

Our final product achieved the basic functionality of our proposal in that we were able to detect a car using our metal detector circuit and send that signal from the Zigbee transmitter node to the Zigbee coordinator node but unfortunately, we were unable to integrate the GUI into the final project. The metal detector portion of the system has displayed a predicted response. The base level frequency is around 270-280kHz and the response of the oscillator with metal over it reached 290-300kHz. The oscillation frequency with other metallic objects showed a consistent change from the original base frequency. Based on the changes, we can accept that the oscillator with the homemade inductor coil reached functionality.

Commented [1]: @sr9gv@virginia.edu
Assigned to Sean Reihani_

Commented [2]: just some screenshots with basic info

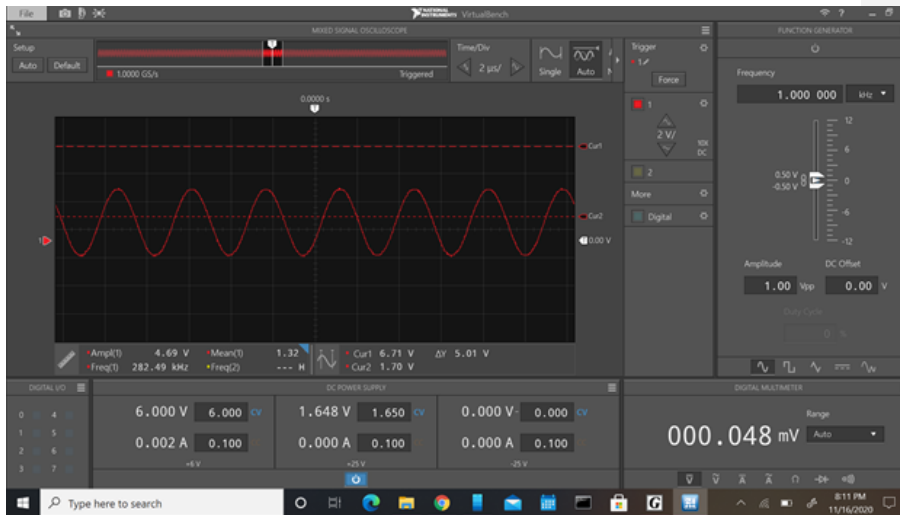


Figure 13: Base Oscillating Frequency

When we wave a metal over the inductor coil, we can see a change in the frequency.

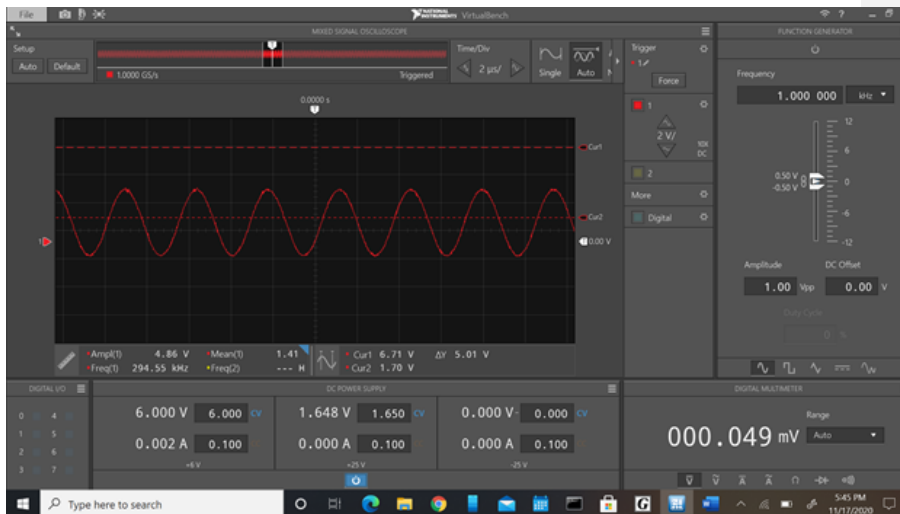


Figure 14: Oscillating Frequency in the Presence of Metal

Costs

The total cost of our project is broken down in the Appendix A. Due to time constraints, some parts that we ordered for the project ended up being unused which slightly misrepresents the total cost of our project if we were to scale it. In order to use this product in industry, we considered the cost of manufacturing 1000 units. The individual unit price would decrease for our as ordering from DigiKey becomes cheaper per unit as the total quantity increases. The time we took to design and put together each node The estimated total cost for scaling our project, broken down in Appendix B, comes out to approximately \$127,000. We were allotted \$500 for the project and scaling that by 1000 would put our budget at \$500,000. Relatively speaking under these circumstances, the actual cost of materials seems reasonable. Another factor that would be taken into account would be how we produced our project. Manufacturing each node by hand would not be efficient at this scale so we would consider using the extra funds towards creating an automated process to increase production time.

Future Work

In future iterations, the first major change would be to successfully integrate the GUI into the program. Distance and time constraints combined with a lack of documentation for integrating Java and UART communication proved more difficult than anticipated to render the project fully functional. Another suggested improvement to our project would be to implement a sustainable battery source. This would have incorporated one of the hallmarks of IoT devices into our project but it became a secondary concern as we continued to work on the project. If a future group wanted to reproduce this product, we would highly recommend choosing your communication protocol carefully. We implemented our project using Zigbee, however, another option would have been to use Bluetooth. We were unaware of the degree of difficulty of trying to integrate Zigbee into our project and it would have been beneficial to do more research into what was necessary to actually complete the project.

References

[1] J. Vespa, L. Medina and D. Armstrong, "Demographic Turning Points Population Projections for the United States: 2020 to 2060", US Census Bureau, p. 2, 2020 [Online]. Available: Vespa, J., Medina, L. and Armstrong, D., 2018. Demographic Turning Points Population Projections for the United States: 2020 to 2060. US Census Bureau, [online] Available at: <<https://www.census.gov/content/dam/Census/newsroom/press-kits/2018/jsm/jsm-presentation-pop-projections.pdf>>

[2]A. Cilluffo and N. Ruiz, "World population growth is expected to nearly stop by 2100", Pew Research Center, 2020. [Online]. Available: <https://www.pewresearch.org/fact-tank/2019/06/17/worlds-population-is-projected-to-nearly-stop-growing-by-the-end-of-the-century/>.

[3]"68% of the world population projected to live in urban areas by 2050, says UN | UN DESA | United Nations Department of Economic and Social Affairs", UN DESA | United Nations Department of Economic and Social Affairs, 2020. [Online]. Available: <https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html>.

[4]L. Adler, "The Urban Internet of Things", Data-Smart City Solutions, 2020. [Online]. Available: <https://datasmart.ash.harvard.edu/news/article/the-urban-internet-of-things-727>.

[5]"The wireless smart parking sensor for detecting parking space occupancy", Bosch Connected Devices and Solutions, 2020. [Online]. Available: [https://www.bosch-connectivity.com/products/connected-mobility/parking-lot-sensor/#:~:text=The%20Parking%20Lot%20Sensor%20\(PLS,parking%20is%20installed%20in%20minutes](https://www.bosch-connectivity.com/products/connected-mobility/parking-lot-sensor/#:~:text=The%20Parking%20Lot%20Sensor%20(PLS,parking%20is%20installed%20in%20minutes).

[6]"LoRaWAN vs Zigbee – Which Wireless IoT Network is the best for me?", Daytech.io. [Online]. Available: <https://www.daytech.io/copy-of-knowledgebase>.

[7] "Metal Detector Circuit Diagram and Working."
<https://www.electronicshub.org/metal-detector-circuit/>

[8]"NEMA ICS 7.1 : Safety Standards for Construction and Guide for Selection, Installation, and Operation of Adjustable-Speed Drive Systems."
https://global.ihs.com/doc_detail.cfm?document_name=NEMA%20ICS%207%2E1&item_s_key=00265241

[6]"Standards - IEEE Environmental Engineering."
<https://environmental.ieee.org/standards>

[7]"KiCad EDA." /

[8]"Code of Ethics | National Society of Professional Engineers."
<https://www.nspe.org/resources/ethics/code-ethic>

[9] Mistri, Raj Kumar & Asheer, Sarah & Kumari, Puja & Toppo, Manisha & Singh, Amit. (2017). Cheap And Efficient Metal Detector. 3. 666-669.

[10] "Parking Lot Safety." <https://www.nsc.org/road-safety/safety-topics/distracted-driving/parking-lot-safety>[11] "Nana C++ Gui Programming Library Documentation"

<http://nanapro.org/en-us/documentation/>

[12]"Electronic Code of Federal Regulations (eCFR)", Electronic Code of Federal Regulations (eCFR), 2020. [Online]. Available: https://www.ecfr.gov/cgi-bin/text-idx?node=pt47.1.15&rgn=div5#se47.1.15_1247.

[13] Christian Pampus, "Method and device for assisting the driver of a motor vehicle." U.S. Patent 9045160B2 issued May 2, 2015.

[14] David High, David Eugene Ferrell, Micheal Dean Atchley, "Navigating a customer to a parking space." U.S. Patent 201701483241 abandoned March 30, 2018.

[15]Gregory Jenson Boss, Rick Allen Hamilton, Andrew R. Jones, Kevin C. McConnell, "Cooperative Parking." U.S. Patent 20070282489A1 issued March 3, 2009.

Appendix

Appendix A: Total Cost of the Project

Product Name/Number	Date Ordered	Qty	Unit Price	Total Cost
LAUNCHXL-CC2650	9/18/20	4	\$34.80	\$139.20
Enammelled Wire	10/13/20	5	\$0.48	\$2.40
560pF	10/13/20	5	\$0.56	\$2.80
2200pF/2.2nF	10/13/20	5	\$0.95	\$4.75
2.2uF	10/13/20	5	\$0.32	\$1.60
10pF	10/13/20	5	\$0.35	\$1.75
Launchpad CC26X2R1	10/13/20	5	\$0.65	\$3.25
C330C225M5U5TA7303	10/13/20	5	\$0.39	\$1.95
LAUNCHXL-CC26X2R1-ND	10/13/20	1	\$47.99	\$47.99
LAUNCHXL-CC26X2R1-ND	10/27/20	2	\$47.99	\$95.98
HM3385-ND	11/9/20	5	\$19.84	\$99.20
36-2463-ND	11/10/20	5	\$1.22	\$6.10
Metal Coild	11/10/20	2	\$7.00	\$14.00
Total Project Budget:		\$500.00		
Total Project Cost:		\$420.97		
Remaining Funds:		\$79.03		

Appendix B: Estimated cost of producing 1000 units

Product Name/Number	Qty	Unit Price	Total Cost
Enammelled Wire	1000	\$0.48	\$480.00
560pF	1000	\$0.56	\$560.00
2200pF/2.2nF	1000	\$0.95	\$950.00
2.2uF	1000	\$0.32	\$320.00
10pF	1000	\$0.35	\$350.00
Launchpad CC26X2R1	1000	\$0.65	\$650.00
C330C225M5U5TA7303	1000	\$0.39	\$390.00
LAUNCHXL-CC26X2R1-ND	1000	\$47.99	\$47,990.00
LAUNCHXL-CC26X2R1-ND	1000	\$47.99	\$47,990.00
HM3385-ND	1000	\$19.84	\$19,840.00
36-2463-ND	1000	\$1.22	\$1,220.00
Metal Coild	1000	\$7.00	\$7,000.00
Estimated Project Cost for 1000 units:		<u>\$127,740.00</u>	