4.2 Design Exploration

4.2.1 Design Decisions

ESP32 C6:
The ESP32 has built-in Wi-Fi capabilities, low power consumption, and energy efficiency. All these functions are critical for our project since we will deploy the pole in the cornfield, and batteries will supply it. The connection between the nodes will be relatively stable, allowing us to transmit and receive the resistance data. ESP32 C6 supports a higher standard Wi-Fi with a stable connection and faster speed. This will be suitable to our case due to the high counts of ESP32 we placed in the field, this will not interfere with other nodes. ESP32 has a built-in security feature which is WPA3, enhancing the network's data protection. Besides, using ESP32 to build out wireless mesh networks can be easily scaled to different ranges, and it can cover more area by adding the pole in the field.

Placement of Interdigitated Electrode(IDE):
We will place the sensor on three different levels, each of which has three sensors. This allows us to get a more comprehensive analysis of the pesticide spray distribution. The sensor at different heights will capture different data during the spray. The sensor's placement is critical because the data we get will affect the spray techniques.

Base Station:
We will use an additional ESP32 C6 as our base station to store the data from multiple sensor nodes. A central node will allow us to manage and analyze data from the whole mesh network. The ESP32 has the capability to store the data from 6-12 nodes from the mesh network in a .txt file. The ESP32 has an built-in long range wi-fi protocol which will make it far easier to transmit data between nodes and the base station.

Wheatstone Bridge:
The Wheatstone bridge design in our circuit aims to get a precise resistance voltage from the IDEs. Due to our needed accuracy resistance measurement, a Wheatstone bridge will be better than a normal voltage divider. Wheatstone bridge circuit is more sensitive to small value changes; in our case, the sensor will measure different resistances ranging between 100k to 200k ohms. The ratio-based circuit will reduce the error created by the supply voltage fluctuations. We selected an op-amp with low noise and high input impedance, LMC660, which allowed us to output the range between 0.1V to 1.0V.

4.2.2 Ideation

Raspberry Pi • Built-in Wi-Fi • Locally stored/ external storage • Access remotely • Low power consumption	**Arduino** • Low power consumption • Support basic data storage • Lack of real-time processing	**Industrial Standard** • Might require specific soft for data access • High cost • High reliability • Weather proves
ESP32-C6 • Limited data storage • Low cost • Low power consumption	**Data Collection For Wireless Mesh Network**	**Cloud-Based** • Real-time data collection • High cost • Depends on cellular coverage
Laptop with Data Software • Large data storage • High power consumption • Less durable • Complex data analysis	**Custom Build(SD Card)** • Low power consumption • Customizability • Limited Capability • Data management challenge	**Lo-Ra** • Long-range data collection • Low power consumption • Depends on the range

4.2.3 Decision-Making and Trade-Off

During our decision-making, we focus on five aspects based on the project requirements:

Criteria	Weightage
Data Storage Capacity	20%
Power Efficiency	15%
Real-time Data Processing	25%
Measurement Accuracy	20%
Cost	20%

Data Storage Capacity
Based on these criteria, we have chosen the ESP32-C6 as our central device for the data collection in our mesh network system. The ESP32-C6 has built-in flash memory, which is enough for moderate data logging. Pairing this with external storage such as an SD card enhances its capacity for larger data sets, especially for long-term data collection. In our case, we need our system to collect data for 3 hours. This decision allows us to ensure this happening.

Power Efficiency
The ESP32-C6 is designed for low-power applications. It supports various power-saving modes, making it an energy efficient choice. This microcontroller is especially useful in remote installations, where battery life is crucial. This also reinforced our decision.

Real-time Data Processing
With a relatively high processing power and Wi-Fi 6 capability, this microcontroller is suited for low-latency data processing applications. The Wi-Fi feature allows faster data rates and reduced latency, which is essential for real-time processing in a distributed network.

Measurement Accuracy
Accuracy largely depends on our sensors provided by Claussen Labs. However, the ESP32-C6's reliable 12-bit ADC (analog to digital converter) and other input interfaces support accurate sensor data capture, making it a solid choice for environmental sensing applications such as pesticide spray monitoring.

Cost
The ESP32-C6 is known for its affordability compared to other microcontrollers with similar capabilities. Priced at nine dollars, it allows for scaling of our project to cover multiple nodes without exceeding budget constraints.

4.3 Proposed Design

4.3.1 Overview

Our design will collect and conglomerate pesticide data. Essentially, we will scatter sensors throughout a corn field at different crop canopy levels that record pesticide saturation at that point. Each sensor will have a microcontroller to which it will feed data. In basic terms, the microcontroller acts as a digital log to collect the sensor measurements. Data can then be sent between these microcontrollers and eventually to a central node (see Figure 1). The central node acts as a base station where all sensor data ends up. It allows researchers to remotely pull all pesticide saturation data from one station. The measurements will be organized into a user-friendly text file.

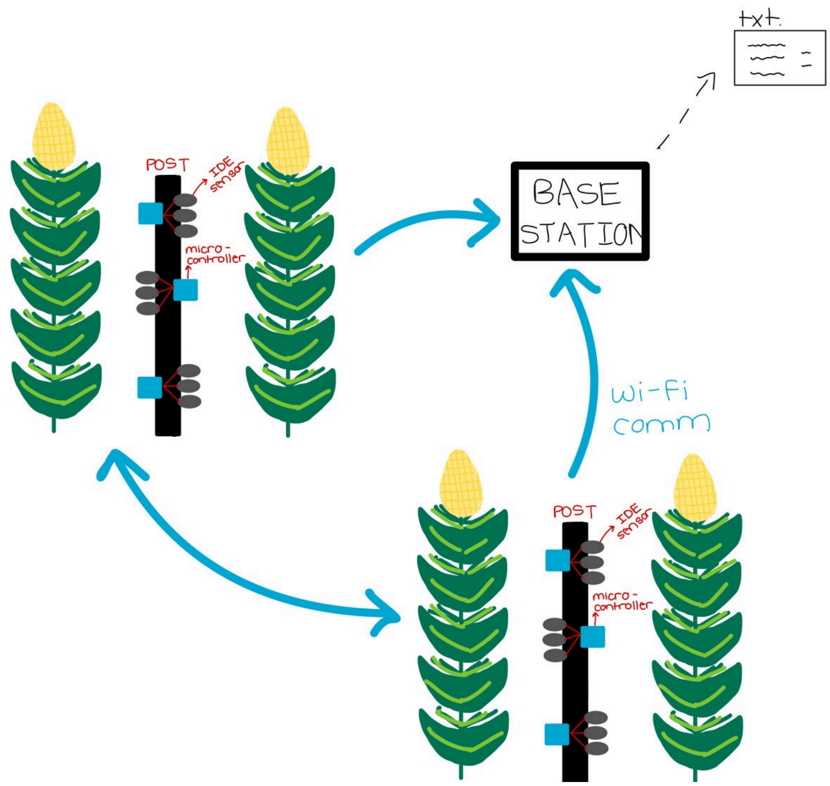

Figure 1: Simplified overview of the communication system

4.3.2 Detailed Design and Visual(s)

The goal of our project is to develop a wireless mesh network that assists in the monitoring and mapping of pesticide spray. Using interdigitated electrodes (or IDEs) developed by Claussens Labs, we can collect resistance values at different levels of the crop canopy. These values will correlate to the pesticide saturation at that level of the post.

Each post will have nine different sensors (IDEs) connected to three different microcontrollers or nodes. Acting as a gate between the sensors and microcontrollers will be a PCB. The PCB is known as a Wheatstone bridge circuit. We have designed it to take in a resistance value and convert it to a corresponding voltage. It is an imperative step since the microcontroller takes in data via voltages rather than resistance values. It does this via its built-in ADC (analog-to-digital converter). The digital voltage values will be converted back into resistance measurements through programming. Next the output will be written to a physical SD card in addition to being transmitted (see Figure 2)

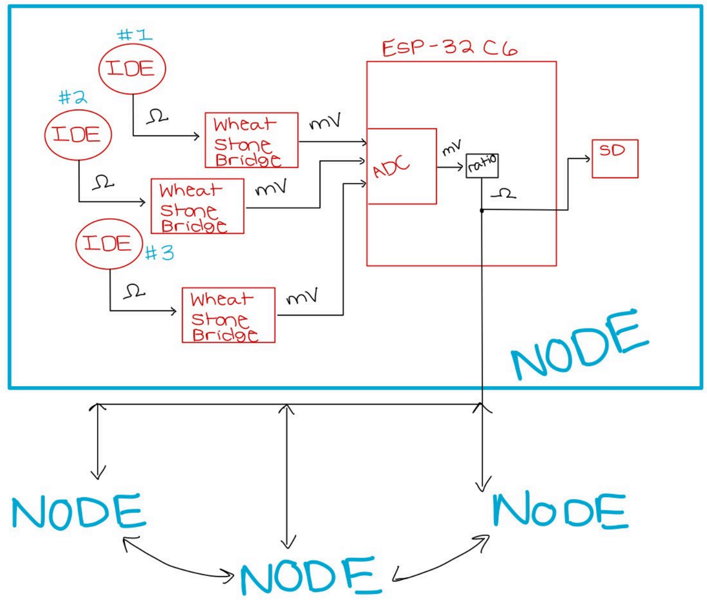

Figure 2: Block diagram of sensor data path

The microcontrollers will transmit the data between nodes (i.e. other microcontrollers scattered throughout the field) and to the base station by a mesh network. A mesh network is useful as it allows data to be transmitted between poles, reducing the need for all microcontrollers to be within the base station's connectivity range. We will utilize the built-in long-range wifi on the ESP32 microcontroller to conglomerate the data at the central node into a user-friendly text document that researchers can pull and analyze. Researchers will thus be able to gather data remotely and automatically, reducing time in the field and increasing productivity.

Our project will propel Claussen Labs forward in their expedition to determine which pesticide distribution methods are most efficient. With an ever-growing world population, determining the best means of pesticide application is necessary for creating dependable and high-yield food sources.

4.3.3 Functionality

In this design, researchers or farmers can efficiently monitor pesticide distribution across a crop field. To begin, the user places sensor-equipped poles in different field areas, each with ESP32 microcontrollers that form part of a mesh network. After powering on the devices, the user goes to a central node—accessible via a laptop with an SSH connection over Wi-Fi—and sends a wake signal that activates all sensor nodes. The central node displays the status of each node, ensuring they are ready for data collection.

As the user applies pesticides, each sensor node records data on pesticide levels at various canopy heights. The user can monitor network health and confirm that all sensors are operational via terminal commands on their laptop. When the application is complete, the user returns to the central node, issues a command to stop data recording, consolidate data at the central node, and put all nodes into sleep mode to conserve power.

Next, the user retrieves the data by downloading a .txt file from the central node, which contains all recorded readings for analysis. Additional terminal commands allow the user to monitor network status and troubleshoot as needed, providing feedback on each node's connectivity. Overall, this design streamlines field monitoring, allowing the user to control the mesh network and data collection from a single interface, reducing manual handling of each sensor and making the system easy and efficient for field operations.

Timeline			
Step	Action	System Response	User Benefit
1	Pole Setup	Nodes initialized in sleep mode	System conserves energy until needed
2	Wake-Up Signal	Nodes activated and confirm presence	User confirms all nodes are active
3	Pesticide Application	Nodes continuously collect data	Insights into pesticide application over time
4	Application Completion	Nodes sleep; data compiled to .txt file	Full record of pesticide distribution
5	Data Retrieval	User obtains .txt file for analysis	Enables post-analysis and optimization

Table 1: Timeline of system use/functionality

4.3.4 Areas of Concern and Development

In this design, several key concerns need to be addressed to ensure it meets user requirements effectively in a real-world agricultural setting. One primary challenge is crop height interfering with network connectivity. Dense or tall crops may block signals between sensor nodes, potentially leading to data gaps in the mesh network. To counter this, we’ll experiment with network topologies and node placements, and possibly strengthen signals with additional hardware. Another concern is data accuracy, as environmental conditions like humidity or interference from other devices can distort sensor readings. Developing calibration protocols that account for these factors will be essential, and cross-verifying data with multiple sensors may help filter out inaccuracies.

Sensor calibration itself is a crucial factor, as inconsistencies among nodes could lead to unreliable measurements. Regular calibration, potentially with an automated feature at startup, will help maintain uniformity in data collection across the field. Additionally, power management is a challenge due to the energy needs of the distributed sensor nodes. By implementing low-power sleep modes, optimizing data collection intervals, and facilitating rechargeability, we aim to ensure battery life without compromising data coverage. Scalability also presents a challenge as field sizes and monitoring needs increase. Larger networks might require more sophisticated data routing and adaptive node configurations to handle increased loads without compromising performance.

Overall, the current design largely meets user needs by providing a flexible, SSH-based interface that allows users to control data collection and monitor network status centrally. However, ensuring consistent performance across various field conditions and scaling the system for larger fields remain concerns. To address these, we plan to conduct testing to evaluate connectivity, implement sensor calibration protocols, test energy efficiency features, and simulate larger network topologies.

For further development, we'll seek client and advisor feedback on acceptable data accuracy and frequency requirements to help us fine-tune data collection protocols. Additionally, they can provide insights on power management with ESP32 microcontrollers in remote setups, and guidance on calibration techniques for environmental variability, which will be invaluable in refining our approach to better meet user expectations and real-world conditions.

4.4 Technology Considerations

For our project we are using both internal and external technologies. Both forms of technology will still be used together and are ultimately going to be connected together one way or another. First, the ESP32 C6 microcontroller, as explained before in our design decisions the ESP32 C6 has various strengths such as: built-in Wi-Fi capabilities, low power consumption, and energy efficiency. Next, our circuit design includes a wheatstone bridge which allows us to precisely measure the resistance of an unknown resistance, which in our case will be coming from our pesticide sensor. Although the Wheatstone bridge is more sensitive to voltages compared to a simple voltage divider it also allows for more accurate resistance measurements, which is exactly what we need for our project. Finally, the pesticide sensor which has been provided to us by Claussen Labs and is a part of our wheatstone bridge. When the time comes to spray pesticide on the crops, this sensor will be doused by a certain amount of pesticide depending on the location of the sensor. The amount of pesticide on our sensor will give us a certain resistance measurement which will be stored in an ESP32 C6.

When it comes to technology available that has similar functions to our project, there is a good amount of products to look into. First, the Libelium, which allows the monitoring of multiple environmental parameters involving a wide range of applications, from plant growing analysis to weather observation. The strengths of this technology are; it supports 30 different sensors covering critical environmental parameters such as soil moisture (can also do temperature, humidity, solar radiation, wind speed, rainfall), easy to deploy, functional wireless mesh network, and energy efficient. The weaknesses of the Libelium are; high cost ($5,000 - $20,000) for the whole system, complex maintenance, connectivity issues in remote areas, and low data security. Another technology that is available is the iMETOS, which has the strengths of real-time data access from the platform, alerts for critical weather events, durable to harsh weather, and integration with other sensors. The weaknesses of this technology are Doesn't directly monitor pesticide spray, poor connection in some areas, high initial cost ($1,500 - $3,000), and a complex setup.

For our project, we looked at these available technologies and found possible solutions and designs that could be implemented into our own project. Including the solutions and resources provided to us by our client, we were able to narrow down and start designing our own circuits and mesh networks. For our circuit, that will be connected to the sensors and microcontroller, we initially started with a simple voltage divider. This worked but had to have such a high resistance that ultimately caused a lot of room for unexpected error. We then found the solution of the wheatstone bridge, which can be seen in Figure 3. This new circuit allowed us to lower the resistances and lower the room for error, it did not erase it but it allowed us to be able to control it easier. Additionally, the new feature of the wheatstone bridge resulted in more accurate voltages and it also made it easier to find certain voltages due to being able to easily change resistance values on either side of the bridge.

4.5 Design Analysis

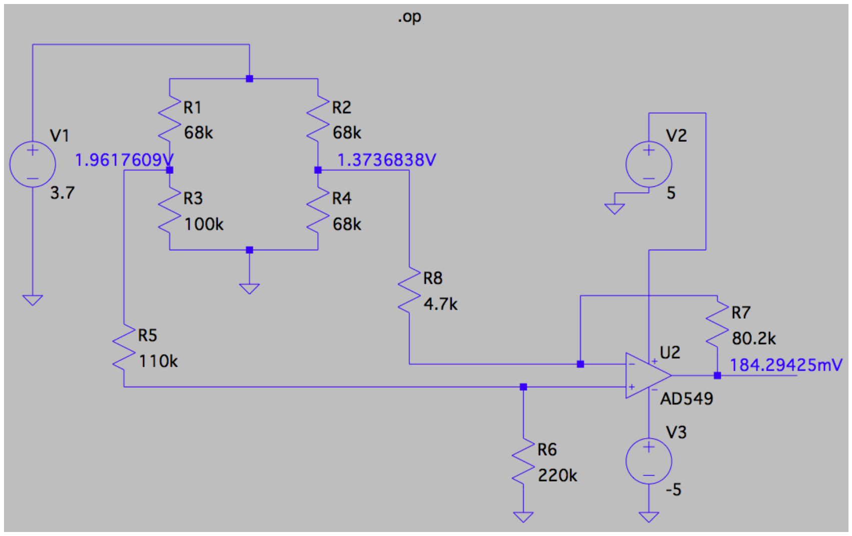

Figure 3: Circuit Design

On the hardware side of our project we have built a circuit shown in Figure 3. The circuit will be implemented with the pesticide sensors provided to us by Claussen Lab. The circuit consists of two big parts, the Wheatstone bridge and a Differential Amplifier. The Wheatstone bridge is the left side of the circuit diagram and its function is to measure an unknown resistance, in our case it would be our sensor, by balancing two branches of a bridge circuit. This bridge allows us to get accurate voltage readings to then send through the differential amplifier. The differential amplifier, which is the right side of our circuit diagram, takes two input voltages and outputs the difference between the two. After the differential amplifier, its output will be sent to the ESP32 microcontroller, specifically the ADC of the microcontroller, where the voltage will be converted to the resistance value of our sensor.

When we are not able to get on campus to use the voltage sources and other instruments to test our circuits, we simulate them on LTSpice. From our experience both forms of testing work well and even give us very similar results. Currently we are working on redesigning the circuit due to some unexpected errors occurring. Through LTSpice, we ran a worst case scenario on the resistor components which led to an output voltage that was out of our range, to fix this we are currently working on reducing resistor values as well as changing our reference voltage to give us a bit of room for slight error that is out of our control.

For the future we are planning on adding a power source and voltage regulators. This will allow us to power our circuit with batteries and regulate the voltage that is coming from the batteries and entering our circuit in order to consistently have an input of 5 volts. Additionally we plan on adding diodes after our differential amplifier to prevent too much voltage or too little voltage to enter the ADC of the microcontroller.

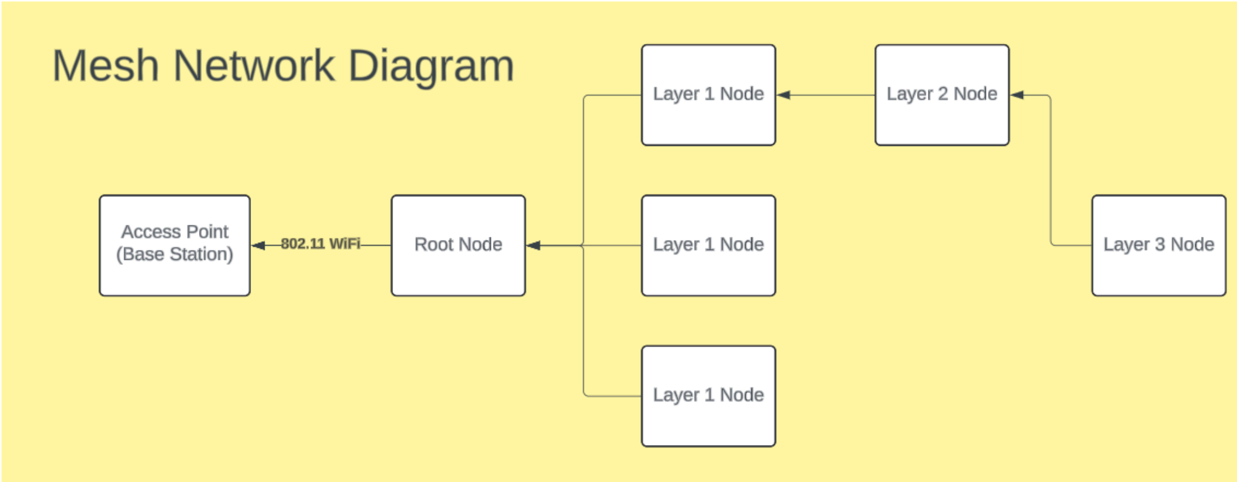

Figure 4: Mesh Network Design

For the networking side of our project, which will be used to transfer sensor data to a central node, we have successfully set up a mesh network framework created by Espressif. It works by programming a single ESP32 (the base station) as an access point. It is to this ESP32 that all the rest will form a network in order to connect to. The rest are flashed as mesh nodes. They are given the SSID and password for the access point. Upon being turned on they search for both the access point and other mesh nodes. The nodes automatically organize themselves into a network where any ESP32 can send a wifi packet to any other ESP32 in the network. Theoretically, the network is self-healing meaning that if a node were to be disconnected, the remaining nodes would reorganize in order to reconnect any other nodes that were connected to the disconnected node.

This current implementation is functional, however there are several areas to be improved. Firstly, the whole network uses conventional 802.11 wifi. In order to meet the requirements of our project we will likely need to use 802.15.4, a special long-range wifi protocol developed by Espressif. Another area that could be improved is that by default only one node connects to the access point (base station). This means that all traffic will need to be routed through one board which could prove to be an issue with the 802.15.4 protocol's low bandwidth of ~250kbps. It would be beneficial if multiple nodes could be connected to the access point so as to avoid bottlenecks in the network.