EE/CprE/SE 491 WEEKLY REPORT 6

10/18/2024 – 10/24/2024

Group number: sdmay25-04

Project title: Wireless Mesh Network for Pesticide Spray Monitoring and Mapping

Client: Claussen Lab- Iowa State University

Advisor: Nathan Niehart

Team Members/Role:

Software Side -

Ashley Falcon: IDEs and Microcontrollers, Group Communicator

Drew Scheidler: Mesh Networking; Note Taker

Henry Hingst: Mesh Networking; Group Leader

Hardware Side -

Hector Perez Prieto: Microcontroller; Circuit Design and Testing

Yok Quan Ong: Circuit Design and testing; Microcontroller

Wesley Smith: Circuit Design/Simulation; Microcontrollers; Note Taker

---

- **Weekly Summary**
  - Connected and tested software and hardware components with favorable readings/results
  - We had an advisor meeting as well, where we determined the teams' next steps
    - Recounted software and hardware teams' progress
    - Received advice on current progress
  - Hardware Team
    - Designed, tested, simulated, and integrated the Wheatstone bridge
    - Successfully communicated with the ESP32 with the circuit
    - Got a solid reading from the circuit, sent voltage values to the ESP32, and got a close reading from the measured and transferred values.
  - Software Team
    - Successfully connected an ESP32 to an existing wifi network
    - Read voltage values from the ADC
    - Displayed an updated reading of voltage values from the ADC to the

monitor
- Powered and ran code on the MC independent of connection to a computer

- ***o*** **<u>Past week accomplishments</u>**
  - ***Ashley Falcon***:
    - Primary for writing code used to operate ADC on microcontroller
      - Initialized the ADC
      - Set up so that raw data from the ADC could be output to the terminal
      - Flashed the code to the microcontroller successfully
      - Initialized GPIO output pin
        - Set code so when below a certain threshold, the GPIO would go high and turn on an LED
        - This allowed us to determine if the code was still running independently of the computer
    - Worked with the HW team to utilize their circuit
      - Allowed us to observe the interaction between the circuit and the microcontroller code
      - Operated successfully
        - Read output in mV
    - Connected with Clients to determine how they had been flashing code in the past
  - ***Drew Scheidler***:
    - Documented steps and code to install and set up required environments and software
    - Aided with code to use ADC on MC
      - Troubleshooting errors
      - Initialization code for ADC and One Shot mode
    - Set up circuit to test ADC
      - Researched ADC physical capabilities and limits
    - Set up circuit to be powered with power supply
    - Helped setup connection between MC and hardware circuit
    - Researched setup of environment and software on Windows machines
  - ***Hector Perez Prieto***:
    - Looked into adding a Wheatstone Bridge and differential amplifier to our already-built circuit
      - Looked into the benefits that the Wheatstone Bridge introduced

        - Relearned how a differential amplifier worked and applied it to our circuit
    - Made changes to the previously built circuit; these changes improved the range of voltages the circuit would produce depending on the resistance value it received from our sensors
        - Tested in the Coover labs and simulated in LTSpice
        - Calculated theoretical voltage values with specific resistances that matched up with the values of voltages received in the lab
- ***Henry Hingst:***
    - Began implementing and testing wifi connectivity on the ESP32
        - Successfully connected to a wifi network using an ESP32 and had it display information about the network to ensure the connection was successful
    - Continued looking into using 802.11 LR mode on the ESP32
        - Found that Espressif has published a GitHub repository of examples of how to create and configure the wifi connectivity on the ESP32
- ***Yok Quan Ong:***
    - Worked on simulating and testing the sensor value using a Wheatstone bridge and op-amp
        - Integrated into the original circuit, we have
        - Simulated on LTspice and observed the output
        - tested on the actual circuit and observed the reading
    - Work with the software team to get the reading from Wheatstone bridge
        - Observed the measured range and the data received from the MCU side
- ***Wesley Smith:***
    - Worked on simulating and testing a better way to convert sensor values to voltages using a Wheatstone bridge and differential op-amp, as recommended by our advisor.
        - Looked into how the Wheatstone Bridge works
        - Integrated it into the circuit we had
        - Simulated and adjusted the circuit resistances to output the goal voltage range
        - Tested it in the lab
    - Worked with the software team to get a feel for how the circuit will integrate with the ADC of the microcontroller

- ***o*** **Individual contributions**

NAME	Individual Contributions	Hours this week	HOURS cumulative
Ashley Falcon	ADC coding, ADC circuit simulation	**7**	**32**
Drew Scheidler	ADC testing, Environment setup, and research	**7**	**35**
Hector Perez Prieto	Made changes and corrections to the circuit, we had to expand our voltage ranges further	**8**	**34**
Henry Hingst	ESP32 wifi testing and implementation	**6**	**34**
Yok Quan Ong	Circuit simulating and testing	**6**	**32**
Wesley Smith	Circuit testing and simulation, integration to ADC	**7**	**36**

- ***o*** **Plans for the upcoming week**
  - Hardware
    - Add a voltage regulator after the power supply to stabilize the voltage input in the Wheatstone bridge circuit
    - Change the input voltage to 5V (Boost Converter)
    - Adjust resistances of the current circuit to output the voltage range goal with changes
    - Run worst-case analysis on LTspice
    - Check the common mode rejection
    - Set up a back-to-back reverse bias after the output of the op-amp to prevent too much voltage from entering the microcontroller
    - Email client to get a concrete range of sensor value (resistance)
    - Calculate the voltage/resistance correlation s
  - Software
    - Implement a wifi network between 2 ESP32s
    - Continue determining how to use 802.11LR on the ESP32
    - Calibrate ADC to output more accurate voltage readings
    - Research and document Windows environment setup

- ***o*** **Summary of weekly advisor meeting**
  - Professor Neihart provided feedback on both the hardware and software teams.
  - Hardware:
    - Provided advice on how to improve our design of the Wheatstone bridge
      - Change the voltage input to 5V
      - Add a voltage regulator to the circuit
      - Balance out the Wheatstone bridge with -5V

    - Run worst-case analysis on LTSpice on our circuit to verify that our circuit outputs voltages in the desired range when the input voltage is low or too much resistance is introduced
    - Clarification on how to set up a back-to-back diode to make sure the ADC of our microcontroller does not receive too much or too little voltage
    - Email the client and ask for the actual minimum and maximum resistance value that the sensor introduces to the circuit
    - Figure out the voltage/resistance correlation of our circuit
- Software:
    - ADC Accuracy: The ADC readings are slightly off by about 10mV, but this can be offset. A 1-5% random error margin is acceptable, with a circuit output range of 0.184-1.067V.
    - Power Supply: Ensure the microcontroller's 5V supply provides enough current, and use a step-down regulator to derive 3.3V from the 5V source.
    - Sampling Rate: Determine the appropriate sampling rate and how to set it.
    - Communication Protocols: Discussed using TCP/IP or UDP for data transmission with link-level protocol, CRC for resiliency, acknowledgment mechanisms, and handling timeouts.
    - Data Packaging: Ensure data is properly arranged with metadata for sender and receiver verification, including checks for dropped bytes. Implement start/stop bits and send multiple numbers in a single package, verifying accuracy at the receiver end.
    - Unique IDs: Each system or sensor should have a unique identifier, possibly using MAC addresses or addresses created with switches.