EE/CprE/SE 491 WEEKLY REPORT 4

10/4/2024 – 10/10/2024

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:

Ashley Falcon: IDEs and Microcontrollers, Group Communicator
Drew Scheidler: Mesh Networking; Note Taker
Hector Perez Prieto: Microcontroller and IDEs
Henry Hingst: Mesh Networking; Group Leader
Yok Quan Ong: Circuit Design, IDE
Wesley Smith: Circuit Design; Microcontrollers; Note Taker

- **Weekly Summary**
  - This week, we continued plowing ahead in our project
  - We each completed our assigned tasks from the week prior
    - Found more documentation for the user manual and hardware abstraction layers
    - Looked into the functionality of the microcontroller, including built-in functions, registers, and physical hardware (i.e. ADC)
    - Dissected the schematic created by the client and concluded that it needed to be completely reworked
    - Researched different Raspberry Pi's
    - Documented different communication protocols in terms of range, specs, and feasibility
      - Wifi
      - LoRa
      - Bluetooth
      - Zigbee
  - We met with our advisor and discussed:
    - Base station
      - Discussed using a Raspberry Pi vs. Microcontroller

        - More research needed
    - Communication protocols
        - The microcontroller's built-in long-range protocol could be best
    - Discussed how to rework the circuit to determine resistance of pesticides on electrodes
        - Will use a voltage divider
        - Preliminary circuit built
    - Created more concrete roles
        - Ashley, Drew, and Henry will be more on the software side
        - Yok, Hector, and Wesley more on the hardware side
    - Updated task list

- **<u>Past week accomplishments</u>**
    - ***Ashley Falcon***:
        - Scheduled advisor meeting
        - Researched Hardware Abstraction Layers (HAL) and APIs
            - Determined that the ESP32-c6 consists of:
                - Low Level (LL) layer
                - Hardware Abstraction Layer
                    - One that I was looking into
                - Driver layer
            - HAL refers to the operation of hardware peripherals in a series of steps
                - Register information is "hidden" or abstracted
        - Found HAL files specific to ESP32-c6
            - Looked into manuals and the manufacturer's website
            - Found .c and .h files for each peripheral
                - Touch sensor, JPEG, LCD, LEDC, etc.
            - Determined coding language to be C
                - Possible to also use C++
        - Visited ETG with Wesley Smith to determine how to order parts
        - Established meeting date with the microcontroller team
    - ***Drew Scheidler***:
        - Researched Raspberry Pi models
            - Researched various models (Raspberry Pi 1, 2, 3, 4, Zero, Compute Module, Pico.)
            - Criteria for evaluation included CPU performance, RAM options, power consumption, cost, wireless capabilities

(Wi-Fi/Bluetooth)
        - Found Raspberry Pi 4 to be best fit
            - Quad-core CPU and up to 8GB RAM for high processing power
            - Dual-band Wi-Fi for strong network capabilities
            - Best option to manage ESP32 nodes and data collection
    - Researched wireless communication protocols
        - Wi-Fi
            - Favored for mesh network due to high data rates and ease of implementation
            - Ideal for transmitting data from ESP32 nodes to the central node
        - LoRa
            - Long-range capability (up to 10-15 km)
            - Low power consumption
            - Favored for the wireless mesh network due to energy efficiency and coverage for field applications
        - Bluetooth
            - Dismissed due to limited range, not feasible for larger fields
- ***Hector Perez Prieto***:
    - Looked into redesigning the schematic that the client provided
        - Includes finding new values for certain components, the addition of a new component, and removing other components
        - Client had trouble with math regarding the values of resistors, Wesley, Yok and I are going to look into this
        - Received advice from our advisor on where to start to solve this problem
    - Found components and bread boards from previous classes that we can use to test our circuits
- ***Henry Hingst:***
    - Continued looking into different wireless protocols included on the ESP32
        - ESP-Now seems to be more applicable than traditional wifi but seems to emphasize response time over range, which isn't ideal for our project
        - 802.11 LR is an ESP32 specific protocol that can support data speeds of ¼ Mbps over a range of 1km which seems perfect for

our project
    - Began learning how to program the ESP32
        - We will be able to use C for our project
        - We will use Free-RTOS to compile our code and flash it onto the ESP32
- ***Yok Quan Ong:***
    - Researched the ESP32 user manual
    - Looked into the GPIO
        - How many pins we are able to use
        - What is the function of each GPIO
    - Looked into the Register
        - Look for each register function
    - Wireless connection technologies
        - Support WiFi, Bluetooth, and ZigBee
    - Circuit troubleshooting
        - Work with Wesley and hector this weekend to troubleshoot the circuit
        - Decide which part we don't need and already have it in the microcontroller
- ***Wesley Smith:***
    - Learned how to order parts with Ashley from ETG
    - Successfully ordered parts
    - Investigated design options for the future
        - What pieces of hardware are necessary from the schematic presented by the client
        - Some pieces of the hardware weren't needed and are built into the microcontroller
        - May have been why the client was having trouble with the circuit
    - Found an improved way to measure the resistance value produced by the sensors provided by the client
        - Planning on testing the circuit in the lab Sunday with Hector and Yok
        - Made sure pieces needed for testing were in past lab kits from other EE classes
        - Made sure the advisor liked the circuit

- ***o*** **<u>Individual contributions</u>**

NAME	Individual Contributions	Hours this week	HOURS cumulative
Ashley Falcon	Meeting Setup, HAL research, Parts	**6**	**18**
Drew Scheidler	RaspberryPi & Wireless Network Protocol Research	**6**	**20**
Hector Perez Prieto	Brainstorming redesign of circuit provided to us by our client	**6**	**18**
Henry Hingst	Wireless protocol & ESP32 programming research	**7**	**20**
Yok Quan Ong	ESP32 User Manual/ GPIO/ Register	**6**	**18**
Wesley Smith	Ordered Parts & Circuit Designing	**6**	**21**

- **Plans for the upcoming week**
  - Hardware and software teams will meet separately
    - HW meeting on Sunday
    - SW meeting on Monday
  - SW team will download IDE (Eclipse? Or other) and build files
    - Will begin breaking down the SW tasks
    - Henry will primarily be on the base station while Drew and Ashley will be doing microcontrollers
  - HW team will begin building new circuit
    - Simulating Circuit
    - Drawing circuit diagram if needed
    - Testing Circuit
    - Planning implementation

- **Summary of weekly advisor meeting**
  - Our meeting with Professor Neihart was exceedingly helpful this week
  - We found guidance in establishing
    - Communication protocols and what options we should explore
      - Found the native long-range protocol on the ESP32 to be the best starting place
    - Options to be used for base station
      - Microcontroller
      - Raspberry Pi
    - C will be the language we code in
    - Feasibility of current circuit diagram
      - It will be completely reworked

        - Neihart expressed that we should use the ADC internal to the microcontroller rather than an external one
- What steps do we need to take to begin downloading microcontroller files
    - Will set up on SW side before getting a physical microcontroller
    - Advised it best to get the software “working” before flashing onto physical hardware