EE/CprE/SE 491 WEEKLY REPORT 9

11/8/2024 – 11/14/2024
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 Team

- **Ashley Falcon:** IDEs and Microcontrollers, Group Communicator
- **Drew Scheidler:** Mesh Networking; Note Taker
- **Henry Hingst:** Mesh Networking; Group Leader

Hardware Team

- **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

- **Hardware Team**
    - Redesigned the circuit as per requested by Dr. Neihart
        - Our old design resulted in big ranges of error in voltages due to having a huge variety of resistor values
    - Do the calculations and simulations for the new circuit
        - Have handwritten calculations that back up the design of the circuit
    - Plot the output of the simulation results
        - MATLAB script that shows relationships between voltage and resistance
    - Run the worst case simulation with Rx is fixed value
        - New circuit showed better results when running the worst case simulations on LTSpice
- **Software Team**
    - Continued reading about the wifi mesh network provided by Espressif
        - Started a visualization of what the code actually does since it’s positively gargantuan
        - Still unsure how to transmit data using the network
    - Brushed up on TCP vs UDP and how they’re implemented on the ESP32

- We will want to use TCP since we need to guarantee the reliable transfer of sensor data to the base station

- Looked into an IDE created by Espressif specifically for programming the ESP32 with C
  - It is interesting and seems like it could streamline flashing code to the ESP32, however it doesn’t currently work for us and seems like it will need fairly extensive setup to work
- Explored SD card documentation
  - Downloaded software in order to format SD card
    - FATFs formatting for file system
  - Determined how microcontroller should connect to SD card reader
    - Selected GPIO ports corresponding to SPI outputs
    - Includes MISO, MOSI, Clock, and CS
    - Played around with menu configurations on microcontroller corresponding to Espressif’s example code
      - Determined that SPI frequency needs to be set relatively low
- Wrote dummy code to test microcontroller clock
  - Used oscilloscope to verify clock output
  - Determined via oscilloscope that data is not being output
  - Need to further explore test SD code provided by Espressif

<u>Past week accomplishments</u>

- **Ashley Falcon:**
  - Researched SD card formatting
    - Downloaded and installed SD card software in order to set the format of the SD card
    - Ties back to FatFs documentation
  - Read into SD/MMC documentation
    - Determined what header functions in the example code were actually doing
  - Utilized oscilloscope to verify clock signal and determine lack of data signal
  - Familiarized with the menu configurations while running Espressif example code
    - Ex. SPI speed, debugging, SD formatting, etc.
- **Drew Scheidler:**
  - Verified functionality of clock signal
    - Used oscilloscope to observe clock
    - Verified correct frequency

    - Researched Fat Fs documentation
        - Gained understanding of initialization process
    - Noted and researched example code dependencies
        - esp_vfs_fat.h
            - Espressif FAT Filesystem Support
        - Sdmmc_cmd.h
            - Espressif SD/SDIO/MMC Driver
- **Hector Perez Prieto:**
    - Researched balanced wheatstone bridges and the effect resistors at high and low values had on the output voltages to then apply to our new circuit design
    - Created new circuit that had a balanced wheatstone bridge at a nominal value of 150 k ohms
        - Resulted of an output voltage roughly in the middle of our desired range of 0.1 to 1.1 V
    - Redesigned the differential amplifier to have the same gain on both the positive and negative inputs
- **Henry Hingst:**
    - Continued reading about the wifi mesh network provided by Espressif
        - Started a visualization of what the code actually does since it's positively gargantuan
        - Still unsure how to transmit data using the network
    - Brushed up on TCP vs UDP and how they're implemented on the ESP32
        - We will want to use TCP since we need to guarantee the reliable transfer of sensor data to the base station
    - Looked into an IDE created by Espressif specifically for programming the ESP32 with C
        - It is interesting and seems like it could streamline flashing code to the ESP32, however it doesn't currently work for us and seems like it will need fairly extensive setup to work
- **Yok Quan Ong:**
    - Redesigned the wheatstone bridge circuit
    - Simulated worst case analysis and make sure the output is within the range
    - Calculate the wheatstone bridge and op amp
    - Redesigned the op amp with have the same Ri and Rf value
    - Make the output close to linear
- **Wesley Smith:**
    - Recreated the wheatstone bridge as per the request of our advisor
    - Centered bridge at 150k ohms, Rf and Ri values the same

    - Ran a DC sweep in increments of 10k ohms to make sure our circuit was still outputting voltages we wanted
    - Output the data to Matlab to create a graph of this data, voltage over resistance

NAME	**Individual Contributions**	**Hours this week**	**Cumulative Hours**
Ashley Falcon	SD card formatting, MC data transmission	**8**	**53**
Drew Scheidler	SD Card Component Research, MC Lab Work	**7**	**55**
Hector Perez Prieto	Circuit Calculations and Redesign	**9**	**54**
Henry Hingst	Mesh Network Breakdown & Setup	**6**	**56**
Yok Quan Ong	Redesign, calculate wheatstone bridge	**8**	**52**
Wesley Smith	Remaking the wheatstone bridge/simulation	**8**	**56**

Plans for the upcoming week

- **Hardware Team**
    - Begin work to finalize circuit
        - Research why the circuit outputs data non-linearly
        - Is it fixable by addition of a component
        - How will this affect the ADC
        - Be able to explain/document why the data isn't linear or explain the error we would encounter if the data was linear, otherwise just have the data linear
    - Add all the components we've created thus far and ensure they all work together in simulation, wheatstone bridge, power regulator, power inverter, diode circuit
    - Make sure the diode circuit works in general
    - Get the output voltage close to linear to get ADC to give most accurate resistance values of the sensor
- **Software Team**
    - Continue reading and breaking down the wifi mesh network
        - Need to understand how to transmit data packets
        - This consist of reading the API documentation provided by Espressif and expanding the visualization of what the code is doing
    - Attempt to change the wifi mesh network to use 802.15.4 instead of 802.11
        - According to the API documentation this should be relatively simple

  - Gain better understanding of example code’s test file
    - Determine how APIs are actually being used within the example code
    - Build up slowly from the example code
  - Determine disk.io initialization
    - Where is it occurring in example code
    - How do we implement this ourselves
  - Successfully store data file from microcontroller on SD card

Summary of weekly advisor meeting

- Professor Neihart provided feedback on both the hardware and software teams.
- Hardware:
  - Change the op amp to the latest version
  - Try to fine tune the output to get more linear
  - Run the worst case again with 1% and 5%, sweep every 10k ohms
  - Plot the simulation data with the calculated version
    - Include the range of the error that shows up when the worst case scenarios are ran
  - Figure why our graph of our simulated data not linear and find ways to fix it
- Software:
  - Reiterated goals for the week
  - Suggested better understanding header files and exploring the bones of APIs
    - Particularly for gaining a better idea of computer architecture and file structure
  - Determine how to set up disk.io
    - This will actually establish data lines between microcontroller and SD card
  - Retry oscilloscope testing with trigger mode
    - Determined that signal/clock signals may be very quick and we might have missed them in our initial testing
  - Continue pushing forward in SD initialization