**Memo**

To: Professor Pisano, Professor Hirsch, Professor Osama

From: Ben Livney, Maxine Loebs, Sergio Pareja, Emanuel Perez, Noah Spahn

Team: 27 - Plants are Neat

Date: 02/25/2021

Subject: Second Prototype Test Report

**1.0 Required Software and Hardware Components**

1.1 Hardware

* 4 Adafruit Feather M0 microcontrollers
* 1 Custom made PCB
* 1 Solar panel (or benchtop power supply for indoor testing)

1.2 Software

* Arduino IDE
* MatLab IoT ThingSpeak
* RadioHead.h
* ESP8266WiFi.h
* RH\_RF95.h
* Adafruit\_Sleepydog
* RHMesh.h
* RHGenericDriver.h
* SPI.h
* WiFi.h
* HTTPClient.h

**2.0 Mesh Communication Between Nodes**

2.1 Network Procedure

The test uses three nodes. Two nodes are used as the outer nodes while the third node is the mother node. The mother node polls for three outer nodes, one of which is not connected to simulate a broken node. The use of mesh network functions are used to communicate between the nodes. The requested node listens to LoRa waves for their own specific ID to send their collected data to, otherwise it waits. In addition to this behaviour, the test also examines the success of conveying messages completely without any misplaced/missing characters.

2.2 Network Measurements

After the initialization of the nodes (flashing the code into the board), it only took one iteration to request and receive information from the desired outer node. Both message from and to the mother node was complete without any missing/mismatch characters. The behaviour was repeatable to other nodes (the rest of the network). Information was passed around efficiently by using the shortest path from the desired node to the mother node using the mesh network’s routing table. As a result, outer nodes do not fully communicate with each other unless it is necessary. Also, an error message was sent to the console when it was not able to communicate with a missing/broken node.

2.3 Network Conclusion

The test demonstrated the use of mesh network protocol to communicate successfully between outer nodes and mother nodes as well as between the outer nodes. The test is the foundation of further testing to find the longest possible range of communication between the nodes that exploit the mesh network’s routing table. The test also demonstrated the addition of more nodes will not cause any miscommunication or errors between the outer nodes.

**3.0 Power Systems**

3.1 Power Systems Procedure

To perform this test, the featherboard was inserted into the custom PCB, which was connected to

a benchtop power supply *en lieu* of a solar panel for indoor testing. The LiPo battery was

connected through an ammeter to the LiPo battery connector on the custom PCB. In order to

measure the relevant parameters for this test, the battery was connected to observe whether or

not the featherboard is turned on without the benchtop power supply. Then, the battery was

disconnected to observe featherboard activation, this time as a result of the power supply. Finally,

both the battery and benchtop power supply were activated to measure battery charging current

in relation to the power supply sink current.

3.2 Power Systems Measurable Criteria

When the battery only was connected, the battery’s sink current was 24mA, which was the featherboard’s required current draw. When the benchtop power supply only was connected, the power supply’s sink current was also 24mA, indicating that the featherboard was being properly powered from both sources. When both the battery and the power supply were connected, the battery charging current was 400mA, which is the programmed “fast charging” current on the PCB, and the power supply sink current was 424mA. This indicated that the power supply was simultaneously delivering power to the featherboard and sensors and charging the battery. This test showed that the featherboard will be able to be powered for long periods of time because of the constant charging of the backup battery and power delivery to the loads when sunlight is available, and proper battery power delivered to the loads when no sun is available.

**4.0 Data Analysis and Visualization**

4.1 MatLab Iot ThingSpeak

Iot ThingSpeak tool used to connect the uplink node, database, analytical functionality and visualisation into a html output stream.

4.2 Data Representation

Data is written and read as strings with the unique channel ID for each console uploading and doing analytics on independent data. This allows the systems to be scalable as more than one system can use the same procedures with custom labeled data as long as more space is allocated on the server.

4.3 Software:

* Arduino 2 Scripts:
  + Multi-Channel Update
  + Server Credentials
* MatLab 1 Script:
  + Analyse and Generate Visualisation of Data
* Html 1 Script:
  + Stream Visualisation to Html Webpage

4.4 Pre-Testing Procedure:

Initial Testing was done with an empty channel and no data in the ID set. Micro-controllers working as the Gateway were connected to power through non-serial usb connection. There are 2 readings that can be taken from the Gateway itself in terms of its onboard hall sensor and CPU temperature sensor. These sensors were used as data points to demonstrate the data path during the test. Ensuring the data test works within a 2.4GHz wifi network was also important for determining the rationale behind server side rejected packets.

4.5 Testing Procedure:

The testing procedure involved initialising the gateway and connecting it to non-serial usb power. The gateway was then let alone for 5 minutes to ensure no user action was required for subsequent data-point upload. The gateway was then reset manually to ensure power interruptions wouldn't interrupt the stream of data from the gateway to the server.

4.6 Measurable Criteria:

Measurable criteria in this test included expectations of datapoints uploaded into channel ID 1310790 field 1 and increment from 0-100 in steps of 9. After reaching threshold the data needs to reflect both on the thingspeak server as well as the google gauges javascript visualisation.

4.7 Measurement Results:

Testing resulted in an expected upload to channel ID 1310790 field 1 in increments on 9. After the reset the patterns started again without intervention. The Google Gauge Visualisation as well as the thingspeak App updated in line with the data within each channel field noting both the data point as well as time of successful upload. Prior to testing -301 error was found often demonstrating a failure of a packet to upload . This was corrected by disconnecting and reconnecting the gateway to the network after each upload to avoid thingspeak blocking subsequent uploads.

**5.0 Using Sensor and Collecting Data**

5.1 Setup

To set up the sensor test, the relevant code that streams sensor data to the serial monitor was

uploaded to a featherboard. The featherboard was then inserted into the custom PCB that has all

of the sensors included.

5.2 Collecting data

After letting the code run for some time, it was evident that the data that was being collected was the sensor data, as the temperature sensors read 80 degrees fahrenheit, the humidity sensor read 5%, and the lux and soil moisture sensors reacted to changes in ambient light and moisture on the probe respectively.

5.3 Measurable criteria

To determine if the sensor data that was collected was accurate, each temperature sensor was warmed via holding it with a finger, and the temperature, as well as humidity, were observed to increase. Similar results were gathered for the lux sensor with respect to increases in ambient light and the soil moisture sensor with respect to the amount of moisture increasing on the sensor’s probe.