**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: 11/22/20

Subject: First Prototype Test Report

**1.0 Required Software and Hardware Components**

1.1 Hardware

* 2 Adafruit Feather M0 microcontrollers
* 1 BQ24074 module
* 1 3.5W 6V solar panel
* 1 3.7V Li-Ion battery pack, not fully charged
* 1 I2C sensor

1.2 Software

* Arduino IDE
* MatLab IoT ThingSpeak
* RadioHead.h
* RH\_RF95.h

**2.0 Communication Between Nodes**

2.1 Network Procedure

For purposes of the test, only two nodes were used. There will be one outer node and one mother node. The mother node polls for certain node IDs to respond, and if the ID coincides with the node, then it sends a reply. The reply for this test is a simple acknowledgment . If the outer node does not coincide, then the outer node relays the message to all around nodes, so that it can find the correct node. The nodes communicate using their built-in LoRa radios.

2.2 Network Measurements

Sending the poll request to listening nodes only took two iterations (each iteration is about half a second). In other words, the mother node only sends two poll requests before receiving a reply. The information retrieved has all the information unaltered from the outer node. The outer node only sends the information if the poll requested the specific node ID, otherwise the outer node sends a reply of acknowledgement. The acknowledgement is sent successfully in one iteration. When the outer node receives a poll request for another node, it sends the poll request to other nearby nodes (in addition to sending the acknowledgment). At this point, the mother node waits for the information from the outer node, and the outer node waits for a different node to listen to it’s poll request.

2.3 Network Conclusion

The test proved a reliable method of conveying information between nodes and working a network. With the addition of improved antennas, the nodes will be able to communicate to each other in a longer range (currently about three hundred meters). The addition of other nodes into the network is possible since the protocol only depends on the node ID which is unique to each node. As a result, a network of multiple nodes is possible with the current software. Next steps would involve the addition of other test sections (solar charging, gathering of sensor data, visualization) to the current hardware and software.

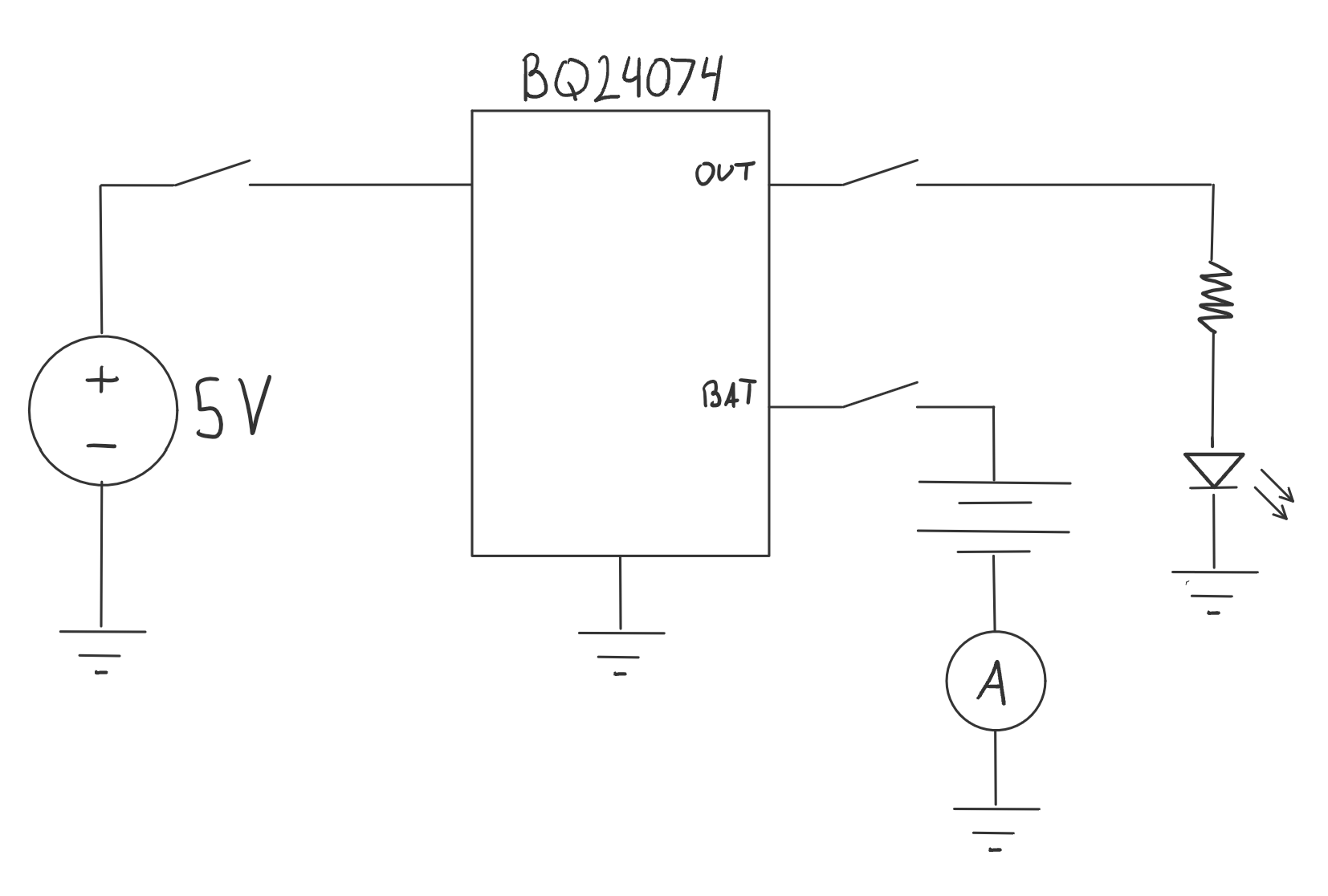
**3.0 Solar Charging and Power Path Management**

3.1 Solar Charging Equipment and Setup

For this test, a BQ24074 module, a 3.7V Li-Ion or LiPo battery, a multimeter, and a power

supply are required. The BQ24074 was connected to the power supply via its Vin and GND

terminals, and its battery output was connected to the positive terminal of the battery. The negative terminal of the battery was also connected to ground through the multimeter current setting. A simple resistor and LED were used as a demonstration load.



3.2 Solar Charging Measurements Taken

Before taking current measurements, the battery voltage was measured with the

multimeter to be 3.19V, indicating that the battery is not fully charged. While the battery

was connected to the BQ24074, the power supply was turned on to 5V. While on the 1A setting, the ammeter in series with the battery read 899mA, signaling that the battery is being charged with nearly 1A of current. Next, the ammeter and battery were disconnected and the demonstration load was connected to the BQ24074 load output while powered from the power supply. The LED was lit, indicating that the BQ24074 is able to power a load from an external source. Then, the power supply was disconnected and the battery was reconnected, and the LED remained lit, indicating that the BQ24074 is capable of powering the load from the battery with no external source.

3.3 Solar Charging Conclusions from Data Taken

Based on these tests, the BQ24074 is a great choice for solar charging and power path

management for the project. It is capable of charging a Li-Ion battery with a high

programmable charging current, and of managing the powering of a load from an external source and a battery. Not shown in these tests due to the testing occurring indoors is the solar charging, which was tested beforehand independently with a 6V 3.5W solar panel. Together, they were able to charge the battery with up to 160mA on a partly cloudy day, and with up to 50mA with obstruction of the panel. From this, we can conclude that even in poorer light conditions than expected, the battery can be charged from a solar panel with the BQ24074 to some degree, and can draw enough current from the solar panel to power the microcontroller and all the sensors.

**4.0 Data Analysis and Visualization**

4.1 Equipment and Setup

Iot ThingSpeak tool is used to connect the uplink console node, database, analytical functionality and visualisation into a html output stream. Data is written and read as an array with the unique channel ID for each unique console uploading data as well as an identical script set running different analytics. This allows the systems to be scaled as more than one system can use the same analytical analysis tools as long as more space is allocated on the server for storing additional data.

4.2 Testing Procedure

To test the web app visualization a html page was loaded receiving a visualization from the private thingspeak channel 18991. This channel is receiving data from a MatLab script running on my local machine. For the test we will note previous data points, upload new data to the private channel. Finally waiting briefly to observe the change in the webApp visualisation. Concluding this process we should have a general understanding of whether the web app is correctly receiving and visualising data.

4.3 Visualisation Measurements

The measurements from the test were taken in comparison to the measurements uploaded to the ThinkSpeak Server. A successful test was noted by identical data points being uploaded and subsequently represented in the visualisation. What we saw in this case was a successful test of a data point 43.3 posted at 4:48:37 and a subsequent data point 100.3 posted at 4:48:56 was visualised with identical values and timestamps in the visualisation.

4.4 Conclusion

After concluding the test we can be sure that the method of uploading and representing data visually is working. Data points can be uploaded with a minimum time in between of 19 seconds. Given our constraints, this method of data transfer and visualisation will work for a consol housed in a wifi accessible area. Given that the minimum time between data points is 19 second we cannot stream more data than this hypothetical maximum but given our power restraints restrict us from sending data more than once every few minutes this is of no consequence. If the console lags behind in uploading for whatever reason and has a que to get through each set of sensor data should have a 20 second stall between uploads to the data stream to avoid errors. Next steps for the development future development would be to integrate the test working on receiving data into the upload mechanic to automatically send any data received by the console to its appropriate data channel.

**5.0 Using Sensor and Collecting Data**

5.1 Equipment and Setup

For this test, we used two sensors (temperature and barometric pressure) to prove we can collect live and accurate data from the nodes. As a result, we uploaded code to read from the sensors and print the results to the serial monitor for verification.

5.2 Measurements Taken

During the test, the code for reading the sensors was successfully uploaded to the board. The code returned data readings every second (temperature and barometric pressure). The pressure readings returned were in line with normal readings at an altitude close to sea level. The temperature readings were initially coming back at about 65 degrees Fahrenheit. To show the sensor accurately picking up changes in temperature, I moved the sensor in front of my heater. The temperature began to consistently rise, changing by about 1 degree per data reading.

5.2 Conclusion

We were able to show that our sensor is responsive to changes in temperature. The data was updated accurately every second. Next steps include integrating all of our sensors onto a PCB.