Final Paper for Fall 2019 EE396 2nd Generation Relay Node: Team Bumblebee

Authors: Arnold Flores, Raellis Young

Department of Electrical Engineering

University of Hawaii at Manoa

Renewable Energy and Island Sustainability Program

Smart Campus Energy Lab

20 December 2019

Abstract: **‘**Bumblebee’ is Smart Campus Energy Lab’s (SCEL) 2nd generation communications module designed to relay meteorological data collected from weather boxes on Holmes Hall. Meteorological data that is collected from the weatherboxes on top of Holmes Hall is sent wirelessly using an XBee device inside the weatherbox to the gateway in Holmes Hall. Due to the limited range of the weatherbox sensor node, ‘Bumblebee’ was developed to extend the range of the sensor node network.

Table of Contents

Introduction **4**

Relay Node: Bumblebee Overview **5**

Block Diagram 5

Design: Schematic and PCB Layout 6

Bare Arduino Board 8

Packet Relay Testing 9

Range (Field) Testing 10

Problems and Solutions **11**

Future Work **12**

Conclusion **12**

References **14**

I. Introduction

In 2008, the state of Hawaii and the Department of Energy decided to collaborate to reduce the dependence on imported fossil fuels. Due to Hawaii being the most fossil fuel dependent state in the United States, the goal of the Hawaii Clean Energy Initiative is to become 100 percent energy clean by 2045[1]. Smart Campus Energy Lab (SCEL) was made in effort under the University of Hawaii's Renewable Energy and Island Sustainability (REIS) program to support the goals of the Hawaii Clean Energy Initiative. The REIS program's goal is to help to UH Manoa to run on its own microgrid that is powered by 100 percent renewable energy.

SCEL's goal is to help make this microgrid by designing, building and deploying "weather boxes" that can collect and send meteorological data such as temperature, humidity, and solar irradiance. These weather boxes should be designed and developed so as to be low cost so that they may be mass produced. Meteorological data that is gathered from these weather boxes will provide help in planning for future renewable energy installations and data for forecasting algorithms[2].

Bumblebee does not include any sensors, so it does not collect any data. The ideal location for Bumblebee boxes would be between distant sensors and the lab gateway. Bumblebee is based off of the third generation of sensor nodes, Cranberry, and uses many of the same components including the Atmega328P microcontroller, the Xbee Pro S2C and is powered by a solar panel and rechargeable battery. The Bumblebee box has a simplified circuit and its own PCB design and housing. Our team also does range testing and networking with Xbees.

II. Relay Node: Bumblebee Overview

Block Diagram

The figures shown below are the overall designs of Bumblebee's power and communication distribution. Figure 1 shows our power system.

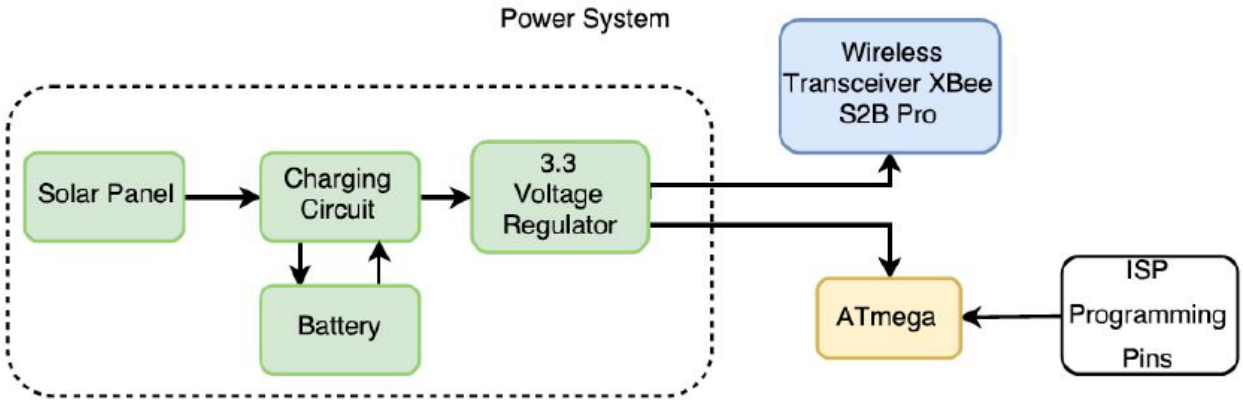

Figure 1: Power Block Diagram

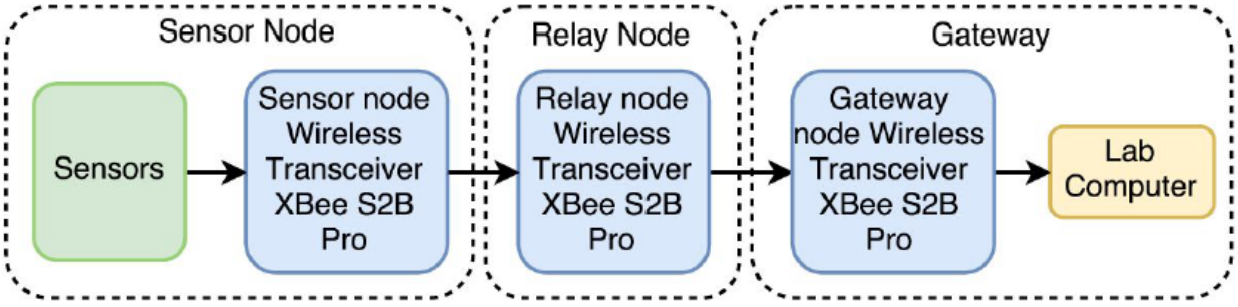

Figure 2: Signal Block Diagram

The power block diagram describes the functionality of each of the hardware components connected. To help power the Bumblebee relay node, a charging circuit and solar panel are

incorporated into the board. A 5V boost converter was implemented last semester to have the Atmega run at 5V with a 16MHz clock. Only one voltage regulator was used to supply enough current to the circuit.

Figure 2 shows how the different sensor nodes will communicate wirelessly. The signal block diagram illustrates the basic path of the data travelling from the sensor node and eventually to the gateway computer. We currently use the Xbee S2C Pro instead of the S2B. This block diagram shows that from the weatherboxes that collect data for its sensors, it will construct a data packet and send it to the relay node. Bumblebee will receive that data and send it to the gateway Xbee to be sent to the lab gateway. Overall, this diagram illustrates that data has to go through multiple Bumblebee nodes in order for the lab gateway to receive a packet.

Design: Schematic and PCB Layout

Referring to last semester's team, the problem that needed to be fixed was that we needed a 5V boost to boost the current 3.7V to power part of the Atmega and 16MHz clock. We didn't modify their schematic or PCB layout from last semester and used it as a reference instead. All the members from this semester are all new. This resulted in a bigger learning curve for us. We had no mentor to guide us, so we were a bit lost and referred to last semester's members for guidance. The main design for this schematic is based off the Cranberry weatherbox.

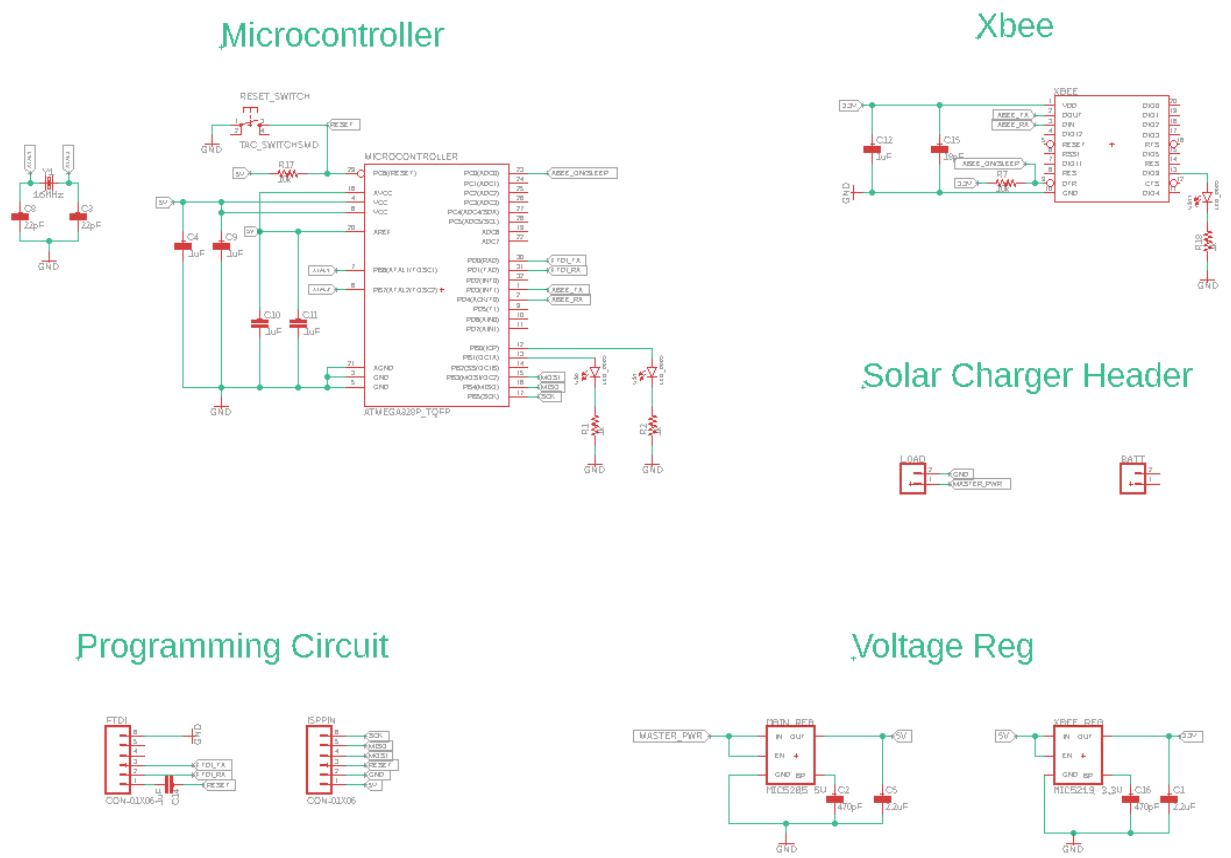

Figure 3: Schematic Layout

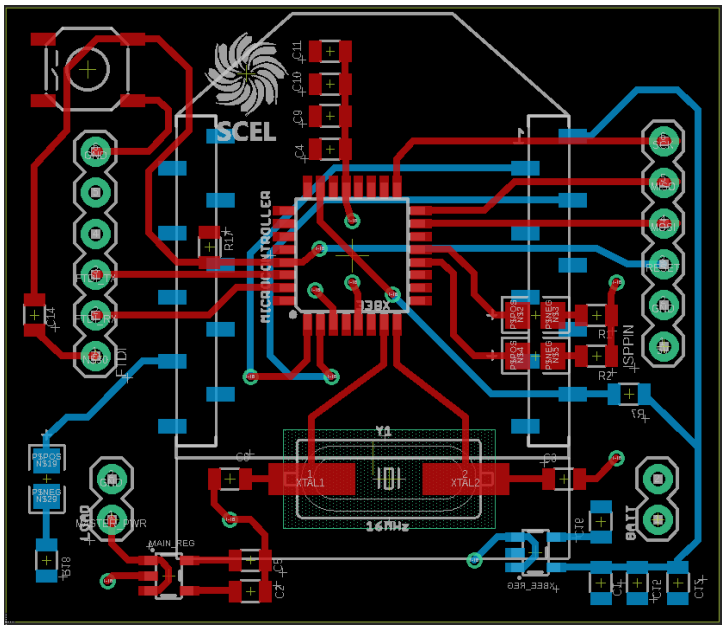

Figure 4: PCB Layout

Bare Arduino Board

From previous semesters, they used a bare Arduino board for relay testing. We looked at the parts in our Bumblebee box and it seemed like the bare Bumblebee from previous semesters were not put together or were torn apart. With this thinking, we started from scratch to build a bare Bumblebee. We were unaware that the bare Bumblebee was already built and that we took it apart. Rebuilding the bare Bumblebee helped us understand the functionality of each component and the relay node itself.

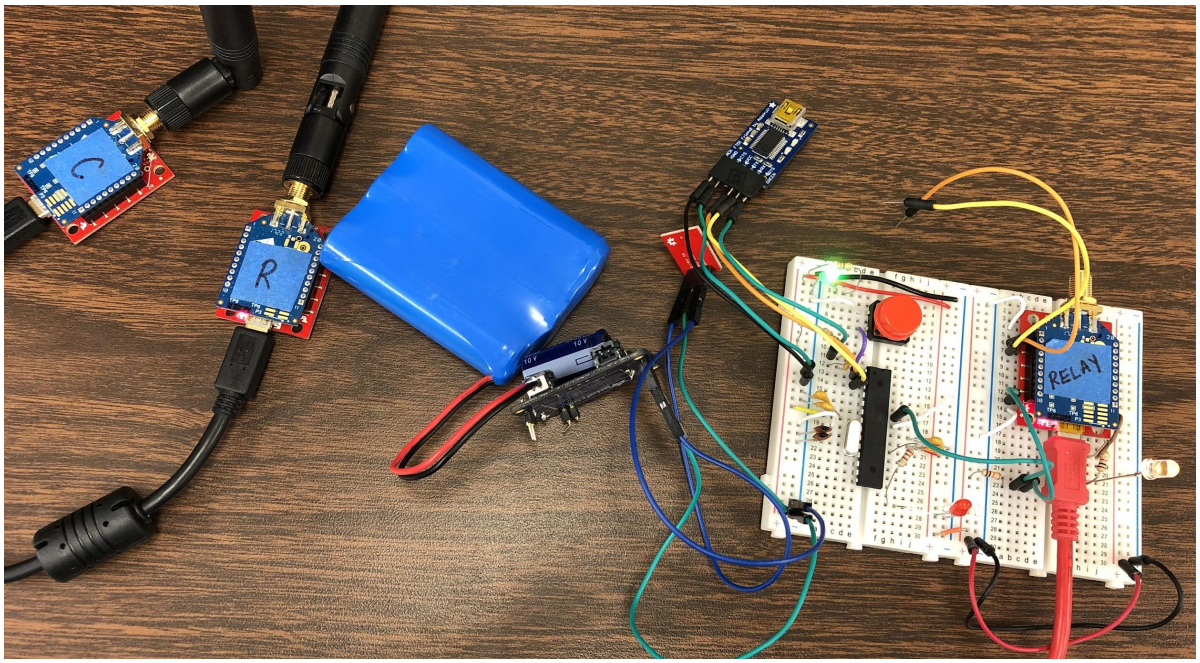

Figure 5: Bumblebee Bare Arduino Board

After putting together the board using the schematic as a reference, we were not yet able to completely test to see if it works due to time constraints. Using the code from previous semesters and help from a previous member, we were told that there was a sync error that occurred somewhere in the board. We were not able to debug that.

Packet Relay Testing

The XCTU and Arduino IDE programs were used to test the abilities of the XBee to transmit and receive packets. The first step is to configure the XBees into either AT or API mode using the XCTU program. For this project, we configured three XBees: one as an API coordinator and two as API router. The coordinator XBee acts as the gateway or receiving-end. The two routers are used for the sending-end (sensor node) and for the relay node. XBees that are configured as routers can transmit packets to other XBees, while coordinators can only transmit to itself. XBees are also identified through their address, which can be divided into two parts — serial high and serial low.

In order for the XBees to communicate with each other, their PAN IDs have to be the same. This can be changed in the XCTU console. When the destination addresses are not specified, the sending XBee can transmit to any XBee with the same PAN ID. To specify which XBee to transmit to, the destination address of the sending XBee should be the address of the receiving XBee. For the XBee that is transmitting packets, make sure to change the API mode from 1(default) to 2. If this is not changed, the XBee can still be recognized by other XBees, however, it won't be able to transmit packets.

After configuring the XBees, attach one XBee to an Arduino board and program it using the Arduino IDE program. A program to send data packets in certain time interval can be written using Andrew Rapp's XBee library for Arduino IDE that can be found online[3]. When programming the XBee in the Arduino board, make sure to change the switch into DLINE on the XBee shield and switch it back after. If not, a sync error could occur while trying to upload the code onto the board. To see whether the sending XBee is transmitting data or there is a

communication between the XBees, go back to the XCTU console monitor and close the port. Once the port is closed, there should be packets received if all the connections are correct.

Range (Field) Testing

The purpose of range testing is to take into account as many variables as possible and to gather data on how far the XBee can implement certain distances such as obstacles and weather. To conduct this testing, a built-in range test software program in XCTU should be used. The data values include local strength, remote strength, packets sent and received, TX errors, packets lost and percentage of packets received.

This is something that we were not able to test this semester because we were still trying to learn everything with the team we were placed in. We also never had a working bare Bumblebee board working until the end of the semester.

III. Problems and Solutions

The initial problem was getting ourselves familiarized with the project "Bumblebee" because we were all new members with no prior knowledge. We were unsure of which direction to head in the beginning of the semester. We were able to consult past members as to what we should do and what a good approach would be. It was unfortunate that our lab hours did not coincide with their breaks. We had to refer to past documentation from the previous semesters.

We weren't able to get the flow of the team until around mid-semester. Our main purpose was to debug the 5V boost part. We found a part in the lab that might've worked and immediately soldered the SMD onto an already populated board. When we powered the board, it immediately

got hot and started smoking. Also, the board that we soldered it on had a trace that got chipped off and the pinouts for the 5V boost part and the PCB did not correspond to each other. This meant that we desoldered the part and used wires to connect the pinouts of the part to the correct pads on the PCB. This also did not work because we smelled smoke and the part we used started smoking.

An alternate solution we proposed was to populate two new boards and test the two different parts, one from this semester and one used from last semester. We were able to start populating the two different boards, but were not able to finish due to the fact that we had no more XBee headers. We tried to order more through Ron Ho, but the order was not processed. We started ordering parts too late in the semester, so we decided to wait until next semester to order them. Another problem was that the 5V boost part we were using to debug got discontinued online so we were not able to order. We didn't know which part was best and after doing research, we were not able to come to a conclusion. We decided that we would consult with a past member in the future.

When using the XCTU program to test our bare Bumblebee board, there was a sync error. After consulting with previous Bumblebee members we believe the problem is with the bare Bumblebee board and not the code. In previous semesters the same code was used, so the sync error most likely has to do with the wiring of the bare Bumblebee board.

IV. Future Work

At this moment, we have 2 almost finished populated PCBs and a bare arduino board that we need to debug with trial and error. Next semester, we plan to continue debugging our PCB and successfully send packets from one Xbee to another. We need to ask a former member about

what 5V boost regulator part would be best to use. Maybe the problem with the board isn't the 5V boost part, but the traces in the actual board so we have to look into that. We will also order Xbee headers along with the 5V boost part we see fit.

We also want to do range testing in Holmes Hall because we were unsure how to do it this semester. Hopefully last semester's members will be rejoining the team and can help guide us in the correct direction to make our time more productive. Also, once we get a good grasp on receiving and transmitting data packets using the XBee on XCTU, we will try to help Team Guava with any problems they have.

We currently do not have access to the Holmes Hall rooftop, so we aren't able to deploy even if we were able to.

V. Conclusion

At the beginning of the Fall 2019, there was a big learning curve to catch up with the progress of past members since we are new members of team Bumblebee. We were able to adjust and learn about our team's material and concepts by referring to past reports, meeting minutes as well as the previous semesters' schematics and PCB layout.

Overall, this project was a great learning experience. We were given the chance to gain more knowledge and enhance our skills in Xbee communications and networking, PCB design, and soldering. We were able to visualize what we learn in a classroom setting and strengthen on knowledge. As new members, we encountered numerous obstacles to learn how Bumblebee really works.

Our biggest success was finding more problems to work on in the future. To us, we think of this as a positive because knowing the problems, we can work on solving it compared to

knowing nothing. We plan to return next semester as 496 students who are wiser, more experienced and ready to debug more.

References

[1] Hawaii Clean Energy Initiative. (n.d.). Retrieved May 02, 2019, from http://www.hawaiicleanenergyinitiative.org/.

[2] (n.d.). Retrieved May 02, 2019, from http://scel-hawaii.org/research/.

[3] Rapp, Andrew “Arduino library for communicating with XBee radios in API mode” Dec. 2016, https://github.com/andrewrapp/XBee-arduino.