Executive Summary

IoT Beagle Bone Cape

DESIGN DOCUMENT

Team 7
Client: Texas Instruments
TI Representative: Mark Eastley
Faculty Advisor: Nathan Neihart

Developers
Taylor Weil - Software design, Meeting Scribe
Sean Griffen - Software design, Documentation Lead
Parker Larsen - Hardware design, Communications Lead
Sterling Hulling - Hardware design, Meeting Scribe

Team Email: sddec21-07@iastate.edu
Team Website: https://sddec21-07.sd.ece.iastate.edu/

Revised: April 25, 2021

Summary of Requirements

- Initial Requirement:
    - Create an Internet of Things capes for the BeagleBone GreenGateway to support an additional wireless communication protocol.
- Additional Design Requirements/Decisions:
    - Create a Zigbee cape to allow the BeagleBone GreenGateway to act as a sensor hub for automotive applications.
    - Add an OBD2 port and supporting firmware to access a CAN bus and read diagnostic messages.

Applicable Courses from Iowa State University Curriculum

- CPR E 288: Embedded Systems I: Introduction
- CPR E 430: Network Protocols and Security
- E E 230: Electronic Circuits and Systems
- E E 333: Electronic Systems Design

New Skills/Knowledge acquired that was not taught in courses

- Zigbee Communication Protocol
- CAN Communication Protocol
- Printed Circuit Board Design

Table of Contents

1 Introduction

1.1 Acknowledgement

- Nathan Neihart: Providing us direction, helping us to stay motivated throughout the project design and Eliminating roadblocks.
- Mark Eastley: Providing valuable resources, clarification of project requirements and ideas, as well as expertise with our hardware.

1.2 Problem and Project Statement

In a world where new cars are filled with new and innovative technology, the older cars are left in the dust with nothing more than an old radio and a few electronic buttons.

The Zigbee Cape will allow the user to buy, create and connect to various Zigbee integrated devices and modules. The Zigbee Cape will physically connect over wires to a simple computer (BeagleBone GreenGateway) to allow the user to write programs to connect, track and send sensor data to either a phone, an external hub over WiFi or bluetooth communication. The addition of an OBD2 port will allow a user to access diagnostic codes and directly access information from and about the vehicle over its CAN bus, and gives the potential to control certain aspects of the vehicle while in a safe state.

This project will not only give tinkerers the ability to increase the functionality of their car, but it will also provide them with an easy method to learn how to communicate using Zigbee and CAN. We will also provide a couple external sensors that will be able to connect wirelessly to the cape.

This project will include 3 main objectives and 1 stretch objective:

Main:

- Create a cape to add Zigbee functionality to the BeagleBone GreenGateway
- Create 2 daughter boards with rudimentary sensors (temperature, humidity, and/or accelerometers) to connect to the cape over Zigbee.

Stretch:

- Either add an OBD2 port to the cape to access the vehicle's CAN bus.
- Or, create a daughterboard with an OBD2 port to access the vehicle’s CAN bus and send CAN information over Zigbee to the cape.

1.3 Operational Environment

- Since this device is intended for use in automotive applications, it may be exposed to temperatures ranging from -20*F to over 120*F. It will be dusty, humid, dry, and may experience fluctuating temperatures with rise or falls of over 60* in 3-5 minutes.
- Sensors may be directly exposed to rain, snow, ice, road salt, UV exposure, and wind in excess of 70mph if placed on the outside of the vehicle.
- Devices and sensors will be exposed to vibration from 40-1khz, and acceleration forces up to 1G. Sensors may experience several peak G’s of deceleration if placed within doors, trunks, or hoods of vehicles.
- The device may not have access to airflow, and may be located in a suboptimal location, and thus thermal management is important.

- The device is intended to be powered via a 5v micro USB connection, and thus must conform to USB usage requirements. TI's Beaglebone comes equipped with a USB port natively, and meets all requirements.
- The device could be powered off the host vehicle's 12v power system or through the OBD2 port (if the stretch goal of CanBus integration is achieved). Care must be taken to ensure that the device does not fully drain the host. (Voltage monitoring and auto-shutdown).

1.4 Requirements

This project is to be an open-source demonstration of the TI BeagleBone Green-Gateway as an Internet Of Things hub. As such, it needs to demonstrate the ability to be utilized effectively as a hub for internet enabled sensors and devices. We need to have a developed library of software pieces that can enable a user to quickly and easily demo our project and continue developing. It is open source, so heavy documentation that is user friendly is a priority.

1.5 Intended Users and Uses

- To properly design an end product that will provide the user with a platform from which to develop further implementation and projects.
- Users are interested in IOT, and are relatively familiar with programming languages such as C, JavaScript, and/or Python, and can edit and compile their own code.
- User has the interest and capability to assemble a cape onto a BeagleBone, and supply sufficient power to the device.
- Users do not use this device for industrial purposes, rather for personal use.

1.6 Assumptions and Limitations

- Assumptions:
    - Open source projects can be used for commercial, for-profit and private sector purposes.
    - Assume that the number of Zigbee connections limited to the max that the Zigbee controller can handle (240 devices).
- Limitations:
    - The system must operate at 12-14v automotive voltage and include power-filtering, or use built-in 5v USB power regulation.
    - UL certification is not performed, assuming all sub-components fulfill all licensing requirements.
    - CE certification is not performed.

1.7 Expected End Product and Deliverables

- The cape and daughter boards will be listed publicly on Github as an open source project that anyone can access and download files from. They can then send those files to be manufactured, and they'll have their very own cape and daughter board like us.
- The technical documentation shall include a full parts list, as well as the purpose of each component and a description of the possible substitutes that might be available (i.e: this 45 mH inductor is critical to set the clock for the microcontroller, these components specified by Zigbee documentation, etc).

- Any other items that will be delivered to the client shall also be included and described, unless their definition and description are covered by existing documentation (i.e: no documentation needed for the Beaglebone, micro USB cable, or 5 v power adapter).
- Examples might include a household power supply to eliminate the need for batteries, a user's manual, or other project reports.
- There shall be at least a one-paragraph description for each item to be delivered.
- Soldering of PCB boards by beginning of August. We want to have our 1st iteration PCB designs back from printing, and be assembling and testing them during the first weeks of August to ensure we have sufficient time for iteration.

2 Project Plan

2.1 Task Decomposition (in no particular order)

1. PCB design and manufacturing
    a. Component selection and testing
    b. Design validation in software
    c. Manufacturing spec validation
2. Zigbee communication library
    a. Connection establishment between hub and peripheral devices
    b. Data validation on send/receive between 1 hub and 1 peripheral device
    c. Parallel communication between 1 hub and multiple peripheral devices
3. Data management/offloading
    a. Rudimentary data storage on BeagleBoard
    b. Connection establishment between WiFi router and beagleboard
    c. Connection establishment between bluetooth device (phone) and beagleboard
    d. Data validation on send/receive for bluetooth and WiFi
    e. Off load on-board data on Wifi connection
4. Integrations with 3rd party services
    a. Connection to open-source home automation software (HomeAssistant OpenHAB)
    b. Connection to Amazon, Microsoft, and/or Google cloud APIs for data storage/access
5. CAN communication library
    a. Connection establishment between cape and attached CAN bus
    b. Data read/parsing validation through OBD2 port
    c. Data write validation through OBD2 port
    d. Safety interlocks

2.2 Risks And Risk Management/Mitigation

- Perhaps put the cape and board in an enclosure so it can't get wet easily, or a conformal coat to protect from corrosion.
- If CAN communication gets implemented, and writing to the CAN bus is feasible, ensure no data writing occurs while vehicle is in an unsafe state (i.e. on or out of park depending on the amount of state feedback is available), and that reading data from the CAN bus does not cause any overloading
- Daughterboards can be off-the-shelf components
- Ensure passive power draw of cape and beagleboard do not cause excessive battery drain of the vehicle

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

1. PCB design is manufactured and validated
2. 2-way parallel Zigbee communication is achieved between cape and peripheral devices
3. 2-way CAN communication is achieved between vehicle and cape
4. Cape can connect to 3rd party services and off-load data through WiFi and can connect to a phone over Bluetooth and display data through a companion app

2.4 Project Timeline/Schedule (Starting week of March 8th)

Week 1: Research Datasheets and dev boards.

- Select 2x sensor types, order

Week 2-3:

- Receive hardware and work on dev boards

Week 4:

- Know what hardware is needed to build out software, breadboard prototypes.

Week 5-6:

- Design software libraries and hardware design for cape and daughter boards

Week 7:

- Finalized PCB design, Prepare to manufacture. Demo Prototype.

Week 8:

- Send off PCB design for manufacturing, order SMD/bespoke components

Summer Break:

- Receive PCB's, solder and test , iterate PCB's if necessary

Week 9:

- Demonstrate on a real PCB.

Week 10:

- Develop documentation and user libraries.

Week 11:

- Evaluate next PCB iteration

Week 12:

- Document 1x sensor implementation

Week 13:

- Document 2nd sensor implementation

Week 14:

- Develop software libraries for multi-function

Week 15:

- Send for manufacture PCB rev2.0

Week 16:

- Software development for data-collection

Week 17:

- Data processing

Week 18:

- Portfolio and packaging.

Week 19:

- Full-feature benchtop demonstration

Week 20:

- Real-world demonstration.

Week 21:

- Submit final project.

2.5 Project Tracking Procedures

Timeline and requirement progress tracking will be mainly done using Trello. A combination of Git commits and the official CHANGELOG in the GitLab repository will show a direct timeline of what code is changed when.

2.6 Personnel Effort Requirements

Task	Software Team Hours	Hardware Team Hours	Explanation
Meetings	40 H	40 H	Each week, our group meets with our advisor and our TI rep to ask questions and give updates
Standup Discussions	30 H	30 H	Each week, our group spends a few hours catching each other up on progress and collaborating
BeagleBone setup	10 H		Getting the beaglebone hardware into a working state and learning the programming interface
Zigbee dev board testing	20 H	0 H	Learning how to program the zigbee controller by using TI software and tutorials
BeagleBone UART testing	10 H	0 H	Programming the beaglebone as a UART master device
KiCad Setup	0 H	15 H	Learning how to use KiCad and setting up the libraries on Git
Zigbee Cape Schematic Symbols	0 H	10 H	Creating schematic symbols for parts
Zigbee Cape Schematic	0 H	10 H	Creating the schematic for the Zigbee Cape

2.7 Other Resource Requirements

In addition software development, we will be designing and manufacturing the hardware for our project. We will require development boards to test our and develop our software on, and then transition to building custom boards using these components. We will need access to tools such as solder irons and hot air rework stations, to solder parts together, and connections to companies that can print our custom circuit boards. A CAN message generator is needed to test CAN communication (could be a car or desktop software). Also, access to a car for environment testing. For software development we will be using examples provided by Texas Instruments to program development boards and the BeagleBone.

2.8 Financial Requirements

As of April 2021, the fiscal resource requirement is unknown until we have our parts completely picked out and our boards designed. Cost per board is not expected to exceed $50, as the TI development board is similarly priced and featured. As we have 4 group members, we plan to purchase and assemble 5-6 boards. This means our financial requirements may grow to nearly $300. This estimate does not include the cost of the prototyping hardware, of which we have already received from TI.

3 Design

3.1 Previous Work And Literature

All of the technology we are implementing into this project already exists. We will be utilizing the Zigbee wireless protocol, a common technology used in IoT projects. Additionally, we will be interfacing with a variety of sensors using our selected Zigbee controller. Our aim is to utilize these pieces of technology to create a hub as an interface for hobbyists and future developers to expand on our project for their own personal use.

The greatest standout of our project compared to similar designs, is that we will be creating an out-of-box-experience. Our users can print out circuit boards and download our code and immediately start utilizing the hardware for themselves. While the Zigbee communication will be open-source and available to the user, they will not need any understanding or programming knowledge in order to get started. In order to accomplish this goal, we will utilize datasheets of development boards already in place, research best practices for wireless technology, and provide detailed documentation for any software libraries we create.

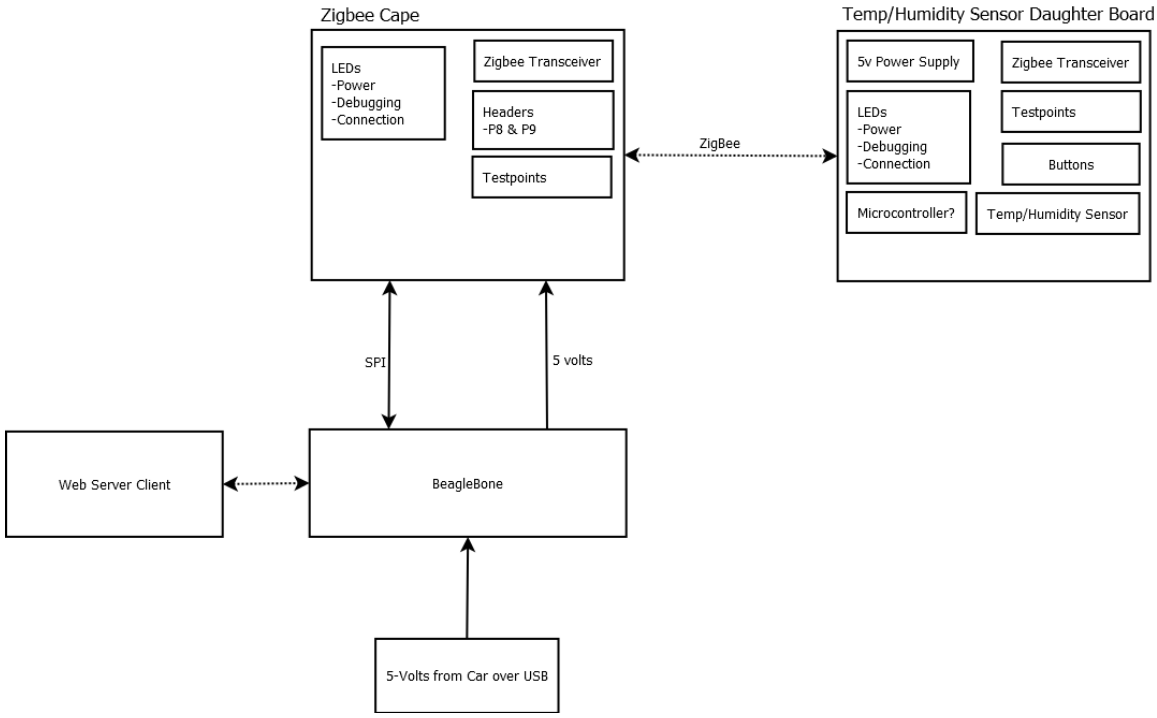
3.2 Design ThinkingMore description about the previous work.

The cape needs to be in the same form-factor as the BeagleBoard it's attaching to, and needs to operate on relatively low power. It also needs to be easy to place in a car, and have all it's critical connections unexposed to mitigate the potential for dust, dirt, or water to cause a short or damage it. Additionally, the software libraries used and third party services should be widely available and well documented as we keep the end-user in mind.

We had proposed additional ideas that would increase the functionality and usefulness of being in a car. One idea was to connect to the car's canbus OBD2 port to utilize onboard signals. This idea was tabled until we feel confident about Zigbee and utilizing external sensors. We hope to make a daughter board that features a canbus OBD2 port that transmits data to our cape design in the future.

3.3 Proposed Design

Our proposed cape design will incorporate a Zigbee microcontroller and numerous feedback LEDs and debugging endpoints. A high-level schematic for a complete system is provided below:



This schematic shows how our cape connects to the beaglebone, and how wireless communication is handled between the cape and the daughter boards, and how the beaglebone connects to third-party clients.

This cape has a dedicated CAN transceiver, pass-through GPIO from the BeableBoard it's connected to, a dedicated Zigbee MCU, and a dedicated on-board sensor of some kind (temperature, light, humidity, etc). Designing the cape with a dedicated Zigbee antenna allows a user to connect any Zigbee certified device to control and get data from. Having a CAN controller allows a user to access diagnostic messages from the car and even send actions for the car to perform like turn on, turn on the a/c, heat, headlights, windshield wipers, etc. The BeagleBoard already has WiFi and Bluetooth built in, so we don't specifically need to integrate those into out cape design.

3.4 Technology Considerations

No technology limitations have come up during the design phase, but any that do will be listed here in the future.

For data reporting and aggregation, we have chosen to focus on connecting to a local webserver and Amazon Web Services. These technologies are commonly used in IoT projects so the end-user could easily implement our design.

3.5 Design Analysis

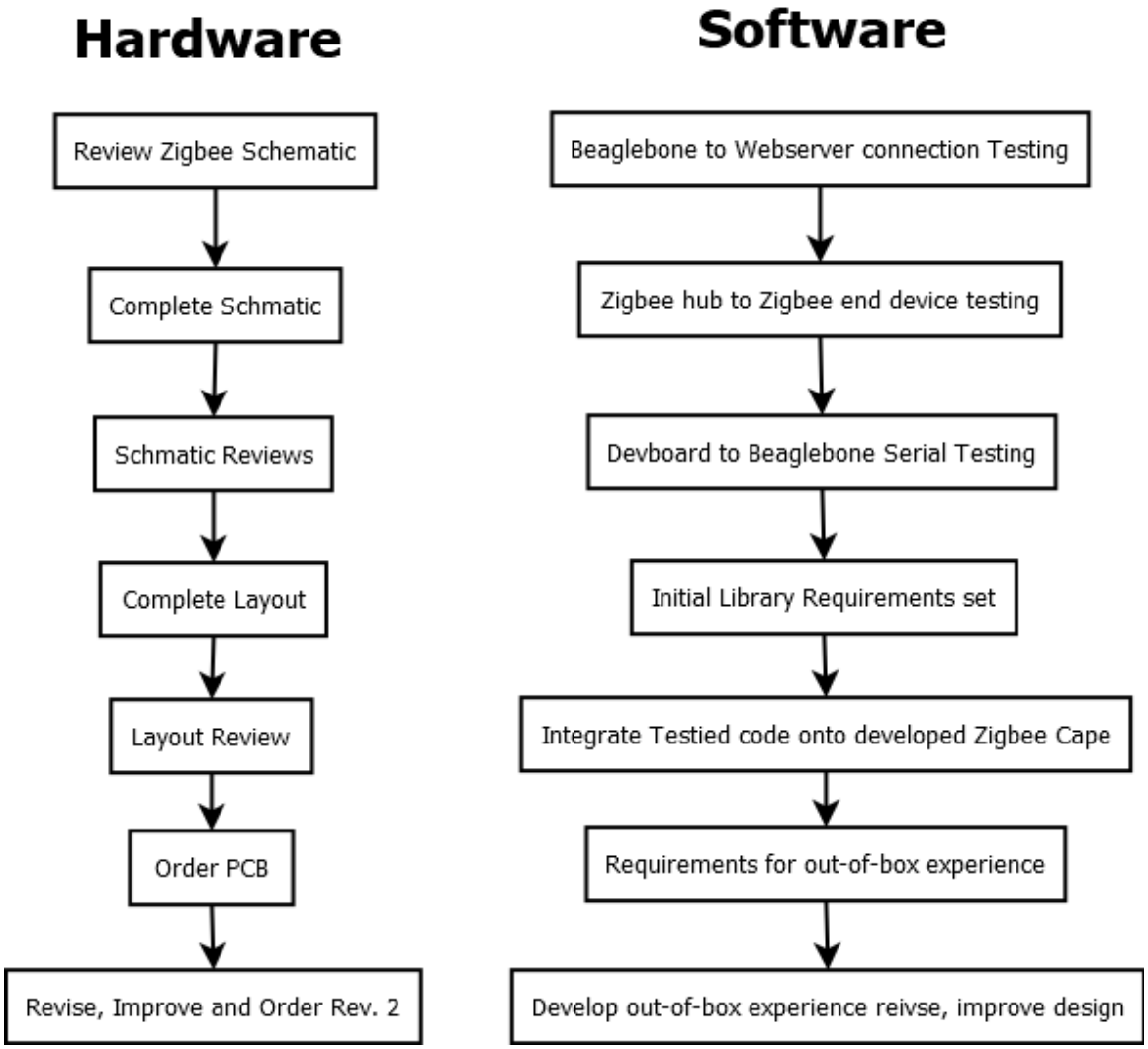
We have brainstormed several ideas for implementing our Zigbee transceiver into a cape. One of our promising ideas that we decided against was to have the Zigbee transceiver on its own removable board that could double as a remote sensor. This would have the effect of streamlining the creation of remotes sensors, as they would have the same PCB. We decided against this as it would still require us to design two PCB's, and wouldn't be relevant due to the prolific

availability of off the shelf Zigbee compatible sensors. We have decided to have our Zigbee integrated into our BeagleBone Cape.

3.6 Development Process

We are trying to follow a pseudo-agile process, using Trello as a time and task keeping resource, and using git and it's methods (commit messages, and an official changelog) as a version control system for hardware designs, and software versions.

3.7 Design Plan



4 Testing

Since we are still in the development phase of the project, most of the testing has been done on development boards and with example programs. As the project progresses, we will be testing functionality and reliability between software and hardware components.

4.1 Unit Testing

Hardware:

Once we receive the board back from manufacturing, we will need to make sure everything is operational. There are many different parts but most of them can be split into 4 different groups:

- Power and Grounds: Once we receive the board back manufacturing, the first thing that we will need to check will be the power and ground nets. If these nets are not all connected and at the same voltage level, there is a very low chance that the other part of the circuit will work as expected. In order to prepare for this step, it will be very important to expose these nets at critical locations or test points
- Headers and BeagleBone Connections: The connection between the BeagleBone and Cape will be completed through the use of headers. We will need to make sure that the boards make a good connection both physically and electrically.
- Programming/JTAG: We will need to make sure that the cape can be programmed successfully. This will be completed through the use of the onboard LEDs.
- ZigBee Communication: This will be one of the last things that we will test when we bring the board up. This will involve using a known working transceiver and attempting to communicate with it.

Software:

Initial library design and functionality can be completed without the use of custom hardware being designed. Testing this on our provided development hardware will involve the following:

- Sending multiple different types of data between Zigbee end-devices and the Zigbee hub.
- Sending data from multiple end devices at once.
- Ensuring data sent over Zigbee is property transferred to the beaglebone.

Once the custom hardware is deemed working, software developers will port working code to the new hardware and perform similar tests. At this point work can be completed on the “out-of-box” experience aspect of our design.

4.2 Interface Testing

Combining Hardware and Software

Once both the hardware and software are working individually. The next step will include combining the two parts. At this stage we will test the code on the PCB and make sure that both parts are working in harmony. The steps we are planning to take are as follows.

- Check hardware: Make sure the PCB is working properly on its own. Perform tests described in section 4.1.
- Check software: Make sure that the software is working correctly on the test board. Perform tests described in section 4.1.

- Add to out-of-box functionality to increase immediate usefulness. This will also ”stress test” key aspects of our hardware and software designs.
- Bring features together incrementally and make corrections as needed. At this point we will have clear design and functionality objectives to compare actual test results to.

4.3 Acceptance Testing

Throughout the process of this project we will complete a list of requirements or tests that we must pass in order for all parties to accept our project. This list will be approved by every member of the team and the clients that we are working with including Mark Eastley and Nathan Neihart. These requirements will include but are not limited to functionality of the final product, documentation for future users and presentations showing the completion of this project.

4.4 Results

Hardware:

As we complete the initial PCB design, we are making detailed notes and documentation of our design decisions and changes. This way we can have more productive design meetings and we can better understand our results. As we progress through this project it will be really important to reference and update these documents to better learn from our mistakes and prevent us from backtracking.

Software:

In preparation for programming the custom PCBs, we have been primarily testing development boards equipped with the Zigbee protocol. In our testing, we have discovered TI’s library for device to device communication. In this library, we will be able to configure one device as a “coordinator” and the rest of the devices as “end-devices”. This will create a network by using the beaglebone shield as an access point for all of the other daughter boards to communicate with.

5 Implementation

Hardware:

After the conclusion of the first design of the printed circuit board, we are planning to start out the semester by powering up the circuit board. We will test all the components separately and then hopefully together if that goes well. Depending on the results, we will make corrections and improvements to the board in an attempt to improve performance. Our goal is to get a working PCB that the software team can successfully program and communicate through.

Software:

Following the testing, we plan on writing a specific library of code that expands off the example code provided by Texas Instruments. Instead of sending a signal to toggle a light, we will be sending a JSON payload over Zigbee with data collected from the sensor units. Additionally, we will utilize our findings of communicating between the Zigbee controller and the beaglebone to forward that message back to the main device. Here, the data is ready to be displayed or forwarded to a cloud service. This implementation will use the custom PCB once it is designed and manufactured.

6 Closing Material

6.1 Conclusion

So far, we have created a rough design outline, received approval from our client for said design, and have a timeline for things moving forward. We also have preliminary test hardware on the way from our client so we can get familiar with the tools we'll be using for the reminder of this project.

6.2 References

"LAUNCHXL-CC1352R1." LAUNCHXL-CC1352R1 Development Kit | TI.com www.ti.com/tool/LAUNCHXL-CC1352R1#order-start-development.

"CC1352R1 LaunchXL." CC1352R1 LaunchXL - Zephyr Project Documentation, developer.nordicsemi.com/nRF_Connect_SDK/doc/1.0.0/zephyr/boards/arm/cc1352r1_launchxl/doc/index.html.

Zuo, Baozhu. "Seeed Studio BeagleBone® Green Gateway." *Seeedstudio*, wiki.seeedstudio.com/BeagleBone-Green-Gateway/.