Software and System Design of the Northern Nomad Tiny House

By:
Theodore Hronowsky - 101008637
Zein Hajj-Ali - 101020677
Adam Staples - 100978589

Supervisor: Professor Donald Bailey

A report submitted in partial fulfillment of the requirements
of SYSC-4907 Engineering Project

Department of Systems and Computer Engineering
Faculty of Engineering
Carleton University
April 9, 2019

Abstract

The Northern Nomad Tiny House smart home systems were a random assortment of non-communicating systems that were not centralized. They followed no plan or design methodology and in some cases hindered the house from achieving its goals to be independent from the internet. The solutions put forth in this project were designed to centralize the systems so that they would be easily viewable and controllable from inside the house. This included bypassing proprietary systems in favour of openly communicating systems that can be connected together. A sensor polling system had to be built from scratch and the data compiled in a format that was cross-compatible with the original proprietary system. Another important result of this project is the focus on documentation. A lot of research needed to be re-done, as the information required was not easily accessible or available. Therefore, this project's documentation will be as easily accessible as possible, so that any future additions or extensions to this project are aided rather than hindered by the work that has already been done.

Acknowledgements

The software integration team for the Northern Nomad tiny house would like to express their deepest appreciation to all those who provided the team the opportunity to work on this project. A special gratitude to our 4th year capstone project supervisor, Professor Donald Bailey, whose contribution, guidance and suggestions helped us to write this report and complete our project. The team appreciates the comments and suggestions by both Prof. Bailey and the second reader, Prof. Majumdar, that have helped to improved our presentation skills and content.

Furthermore, the team would like to acknowledge with much appreciation the crucial role of the past 4th year students, under the supervision of Prof. Bucking, who provided necessary info and background to complete the task of integrating software into the Northern Nomad. A special thanks to the many graduate students who provided us with access/materials/know how to facilitate the design process.

Table of Contents

List of Figures

List of tables

1.0 INTRODUCTION

This document is a report of findings on the software integration into the Northern Nomad Tiny House for the project supervisor, Prof. Donald Bailey and second reader, Prof. Shikharesh Majumdar as part of a 4th year capstone project for SYSC 4907. The purpose of this report is to present the findings and accomplishments of the group. This report includes a project overview which discusses the background, motivation and statement of the engineering problem (section 2.0), an engineering project section that describes key attributes demonstrated (section 3.0), technical and design details about the project (section 4.0) and the conclusions and recommendations from the group (section 5.0).

2.0 PROJECT OVERVIEW

2.1 Problem Background

The Northern Nomad Tiny House was a project started by civil engineering, architecture/ architecture engineering, and industrial design students at Carleton University.[1] The Northern Nomad Project was supervised by Professor Scott Bucking. Some of the goals were to keep the house as sustainable as possible by staying at net-zero water and energy emissions. It was also designed to be comfortable and adaptable to the end-user's daily habits. Some of the graduate students working on the house decided to install some newly developed insulation methods in the walls of the house, so as to test its efficiency using the sensors placed at strategic points in the house. The adaptability was planned to be done using machine learning algorithms such that the

house can for instance turn off its air conditioning systems when it is vacant, and adjust its temperature when anticipating the users arrival.

Figure 1: Northern Nomad Tiny House

2.2 Problem Motivation

The goals stated above required the use of many different sensors and systems around the house and connecting them in a way that facilitates their monitoring and control. Many systems that were expected to be ready for implementation in our solution had not been installed as of the start of this project. The water reclamation system, an atmospheric water generator, was to be installed last fall but the arrangement fell through, so there was talk of creating a new system from scratch. The solar panels on top of the house are mounted but not connected to the house, and thus not producing any power. The solar hub, which is used to analyze the panels and corresponding batteries, has not yet been installed in the house. Some of the systems were already installed and connected, albeit not in the most efficient way. A Local Area Network was installed in the house, though it is currently only being used as a gateway to the CUWireless

network. The LAN is an extremely important feature, since one of the house's goals is to be self-sustaining and completely independent from the internet. Many of the components connect to each other wirelessly over WiFi, but some also require an internet connection. A Wink hub was bought and installed in the house as a way to connect devices that use different communication protocols together.

The performance of the insulation that was being tested in the walls of the house needed to be monitored in some way. The team that built the house decided on using wall sensors from SMT. They had not realized that the only way to read from these sensors was to use the proprietary software SMT provides along with their proprietary branded data logging modules.

2.3 Problem Statement

This project team's goal was to connect these systems such that they can be monitored, controlled, and their data accessed from a central system. The group started as a team of 6, but it was apparent that it would be more efficient for us to split into two groups of 3. One group would tackle the monitoring of the systems, and the other would deal with the problem of compiling the information in a format and system that is accessible by the end user. Our group focused on the monitoring and data collection aspect.

Due to many of the systems not being functional at the time of the beginning of the project, the decision was made to tackle the SMT wall sensors first, since some graduate studies were dependent on the research that they would make available. Our goal was to make the data accessible without the need for a laptop being brought in and connecting to the proprietary data loggers every time a reading needs to be taken. Another part of the goal was to send the data to a designated email server or host it on a server that is accessible to the graduate students remotely.

2.4 Proposed Solution

Originally the process was to be a quick one, as the wall sensors already had software to collect data and output all necessary information through it. However, the software was only available on windows PCs and as such would not be compatible with Raspberry Pi used for collection. In order to get around this issue, the first option was to emulate a windows machine on the Raspberry Pi. This was not successful and the option of a hardware solution was investigated, this was also unsuccessful. This left us with the final proposed solution, and the one currently being implemented, splicing the current wall sensors to a newly created circuit board. The board would then send the collected data, with the use of a subscriber publisher service known as MQTT, directly to the Home Assistant. That data could then be processed using a python script, or similar, created by the other project group.

2.5 Accomplishments

While the project was subject to many changes and deviations from the original project proposal, the team made many accomplishments towards the progress of the project. Through alternatives and changes to the scope of the project, the group was able to redefine the objectives in timely fashion to produce positive results.

As can be seen above, there were many complications with the original proposed solution, which has led to the final solution listed above. To summarize, the original software designed for the wall sensors was not compatible with the integration system that was determined to be most effective. As such the group moved away from software integration to creating a new hardware system that could be polled directly and separately from the installed data loggers for the purposes of storing and displaying the relevant data.

The final accomplishment is that of the 32 to 1 multiplexer system, that can read from all the wall sensors in the house and send that data directly to the Home Assistant for formatting. The system currently exists purely as a prototype however, it can be easily transferred to a format that could be printed to a small form in order to properly install it into the Tiny House.

Aside form the physical prototype, a great deal of time has been spent researching all of the different aspects of the Tiny House's systems to gather information on what is feasible and what is not. Many system design flaws have been found in the installation of the systems in the Tiny House, these can be averted and fixed. The majority of the gain from this year's project is the knowledge of what can and can't be done as well as how the systems should be created in the future to facilitate proper integration. This information is recorded and can be passed on to future students working on this project.

2.6 Overview of Remainder of Report

The remainder of this report will focus on the technical aspects and details of the project consisting of software integration into the Northern Nomad tiny house. Section 3.0 details health and safety concerns (3.1), engineering professionalism in the project (3.2), the use of project management (3.3), the justification of the suitability of each degree program (3.4) and each individuals contributions (3.5). Next, the design aspects (Section 4.0) of the project are explored. In this section, the background and terminology (4.1) needed to clarify the problem is defined. This includes but is not limited to related work, relevant background and any other details necessary to understand the problem set out. The design details (4.2) are then elaborated through

the groups accomplishments to arrive at a solution. Section 5.0, the conclusion and recommendations of the group are discussed.

3.0 THE ENGINEERING PROJECT

In any engineering project, especially a 4th year capstone project, it is important that key fundamentals and attributes are demonstrated. This section, the engineering project, is meant to showcase the knowledge and skills that have been attained throughout an engineering degree at Carleton.

3.1 Health and Safety

When completing an engineering project where the use of software and hardware are necessary, there are often health and safety concerns that accompany. In order to address these concerns, it is crucial to follow standards and practices for safe operation in environments such as laboratories. In the case of the Northern Nomad Tiny House, these concerns were minimal as most time spent working on the project was within a computer lab. Safety concerns regarding software was not an issue however the team did follow safety procedures and practices whilst handling and testing hardware and electronics. For example, circuits and other prototypes were carefully planned out to ensure that the power requirements were met. In this case diagnostic tools such as multimeters were used to ensure safety when dealing with electricity. By understanding wiring conventions used and theory behind electrical fundamentals , it eliminated the associated risk when working with electricity. More so, general lab procedures such as no food or drinks, or ensuring no one worked alone in potentially dangerous situations was put into practice to eliminate and diminish the chance of an accident. By following these general

procedures and practices, the Northern Nomad team provided a healthy and safe environment to work in.

3.2 Engineering Professionalism

As engineering students with intents of becoming Professional Engineers the general requirement of professionalism was applied throughout the course of the project. High regards to ethics, responsibilities, health and safety as well as importance to the environment, well being of society and sustainability was observed. For example, when designing the early prototype, consideration to the environment was reflected by choosing sustainable devices as the hardware to be used. By choosing components that would positively impact the environment, the group upheld its duty to the environment by considering sustainable components. More so, the members of the group acquired and maintained a body of knowledge to be able to make informed decisions. The group continued to renew the knowledge by staying on top of the current trends in technology relevant to the project.

Demonstrating professionalism means taking responsibility for the work produced. Each iteration, mistake, or setback was analyzed so that opportunities for improvement could be applied in the future. Throughout each deliverable and iteration of the design process, the group ensured that the quality of work produced was at its highest. Ensuring that milestones were set and reported to the project supervisor for feedback each week meant that expectations of the client (supervisor) were met or on track. In every aspect of the project, each team member showed professionalism through their motives, attitudes, emotions, and characteristics.

Also, the team took pride in coordination within the group but also with working successfully with other professionals to complete tasks pertaining to the project. Most

importantly, the characteristic of management that was upheld, implies the responsibility of the members of the Northern Nomad group. Understanding that the work of an engineer has an impact on people, society, and the environment and ensuring that the regard to these issues are high is crucial.

3.3 Project Management

In any successful long term team project, it is important to put in place management techniques and processes that are to be used to coordinate, manage, and perform the project in order for the outcome to be positive. From day one, the team employed many management techniques to ensure that the project would be completed in timely fashion. For example, some of these techniques involved:

- Visioning and planning the project potential in advance in order to understand the goals and scope of the project
- Recognizing the potential and skills of each team member
- Periodically evaluating, adapting and adjusting the project upon experiencing setbacks.
- Applying control processes to allow for action upon changes

Through the use of these techniques and discipline, the group established a way to organize the project.

More so, the project was divided into milestones and deliverables which kept the team on track in terms of progress on the software integration into the Northern Nomad tiny house. The deliverables and milestones are shown below:

Deliverables and milestones for SYSC 4907 Northern Nomad Tiny House:

- Project Proposal - Monday, September 24th 2018

- Progress Report - Friday, December 7th 2018
- Oral Presentation - Tuesday, January 29th 2019
- 1st Draft for Final Report - Friday, March 8th 2019
- Poster Fair - Friday, March 15th 2019
- 2nd Draft for Final Report - Sunday, April 7th 2019
- Final Project Report - Tuesday, April 9th

These deliverables and milestones, along with weekly meetings, including both meetings with the team members as well as meetings with the project supervisor, allowed for the team to manage and divide time efficiently. A weekly minute breakdown stating the minutes spent and work done each week can be found in Table A.1 of Appendix A. By creating a project proposal the group was able to define the statement of the problem and the proposed solution. Establishing project information such as issues that may arise, risks/events that could affect the project's objective as well as the projects life cycle resulted in a complete understanding of the project timeline and specifics. Moving forward, the progress report allowed the group to report on the deviation from the original proposal.

In terms of the design of the project, the project was designed in an incremental, iterative manner. This linear approach allowed us to build on top of previous iterations until a fully functional model was developed. Within each iteration, the project management processes were applied in order to define actions needed to complete the work. To facilitate the design process, some design and development tools were used. By implementing these techniques, the team was able to carry out tasks in an organized and thought out manner. Communication within a group project is key to success. Using slack, a team based collaboration service, intercommunication

between the group was performed. Using this application allowed for meetings between the group members to be set up. General discussion of the project was carried out through slack as well. In terms of completing deliverables, Google Drive, along with Google Docs and Google Slides were used, enabling a collaborative environment to work in. The use of real-time, online editing meant that all members did not need to be in the same room at all times to work on the project deliverables. Lastly, Arduino IDE was used as the code development environment for the project prototype. The environment allowed for writing and uploading of programs to the NodeMCU.

3.4 Justification of Suitability for Degree Program

From a purely engineering perspective this project implements many key concepts that are taught in an engineering degree. The first being considerations towards sustainability. As an engineer this is a topic that should never be far from design decisions, especially during the early phases of planning. The second is interdisciplinary integration, ensuring that systems created from many different walks of engineering come together to provide a reliable service or function. This step is crucial during an engineering design process because it allows for an early understanding of how unique and multifunctioning components or devices will be connected together in order to create a unified system.

As Computer Systems Engineering and Software Engineering students, the software integration into the Northern Nomad tiny house project boasts multiple important concepts. Real time monitoring, embedded systems and network communications are all integral to this project and are large areas within our programs in the current industry. As well, this is the first year of a

potentially multi-year project so ensuring all of our code, implementations and documentations are easily readable and accessible will be of great importance and provide a good learning opportunity.

3.5 Individual Contributions

This section itemizes the contributions from the group members to the project. While the project was approached to have a collaborative, team driven environment, some aspects were completed individually. Each member's contributions to the project and report are listed in sections 3.5.1 and 3.5.2.

3.5.1 Project Contributions

Each individual's contributions to the project that are separate from team contributions are listed as follows in table 1 below.

Student	Project Contributions
Theodore Hronowsky	Raspberry Pi testing, Python script to see if serialization was an option, deliverables
Zein Hajj-Ali	Raspberry Pi testing, Python script to see if serialization was an option
Adam Staples	Multiplexing prototype, python scripts

Table 1: Individual project contributions

3.5.2 Report Contributions

The individual contributions to the final report by each group member is summarized in table 2, below:

Student	Project Contributions
Theodore Hronowsky	Draft Outline,Opening Formalities,

	Acknowledgments, 3.0 The engineering project(3.1 - 3.5), section 2.0(2.6), 1.0 Introduction, Appendices
Zein Hajj-Ali	Abstract, 2.0 Project Overview (2.1 - 2.3), technical sections(4.1 - 4.2)
Adam Staples	2.0 Project overview (2.4-2.5) 4.1 Background and Terminology Conclusion, Recommendations References, appendices

Table 2: Individual report contributions

4.0 SYSTEM DESIGN AND CONSIDERATIONS

4.1 Background and Terminology

The goals of the Northern Nomad Tiny House project were to make a habitable tiny house that is independent from the hydro grid, internet, and uses smart home features autonomously for added sustainability. The software integration project allowed a group of Computer Systems Engineering and Software Engineering students develop and connect the systems of the house in a way that enable the use of these features for the stated goals. Many of the systems that were installed in the house were either incompatible, or not installed in a manner that allows for communication between the different protocols. A diagram showing the various systems in the Northern Nomad Tiny House is shown below in Figure 2.

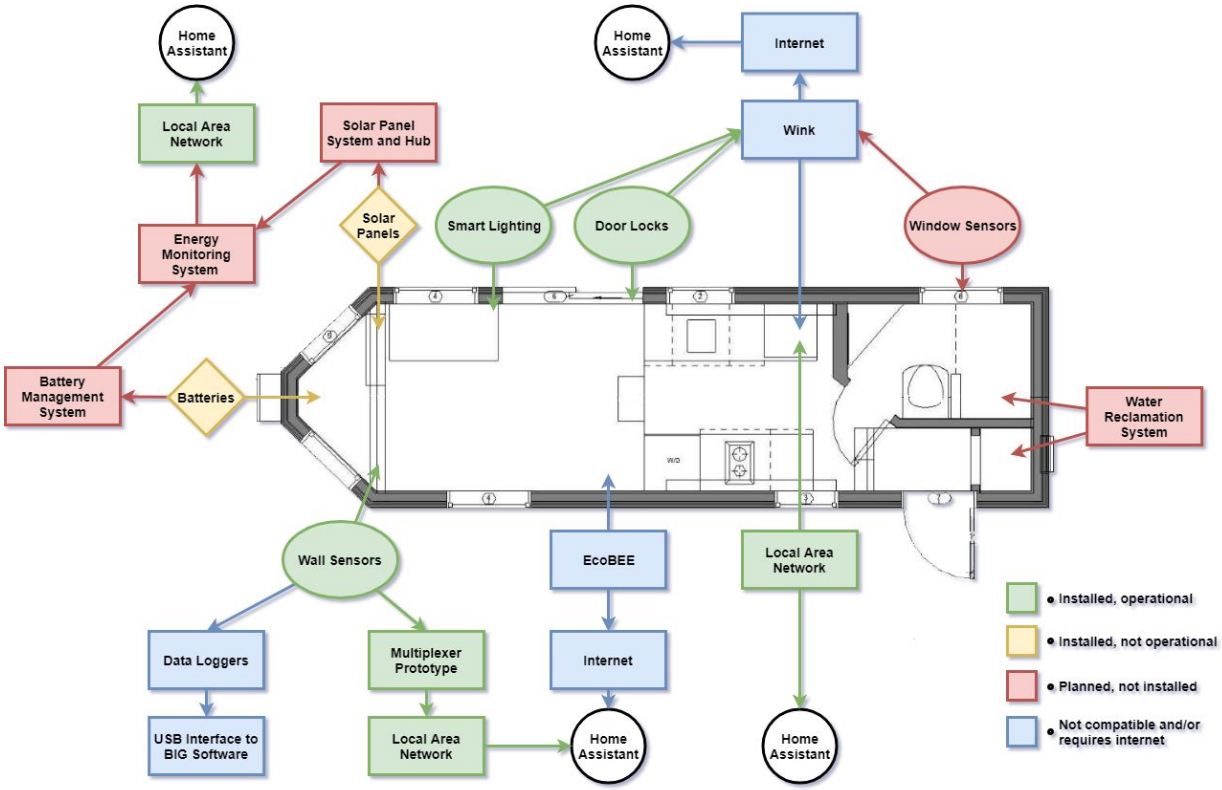

Figure 2: Diagram of Systems in Northern Nomad Tiny House

The Systems Diagram, as well as the diagrams showing the areas of focus of both project groups, can be found in Appendix F.

The following section will break down all components investigated, their viability for integration and the state of their implementation. It is important to remember that as the Northern Nomad Tiny house is under constant development these comments only reflect the state of the building during the September 2018- March 2019 time frame.

Smart Features

An earlier effort was made to install smart features though there was little initial functionality to the system, and multiple major system flaws. A Wink hub, which communicates on both the ZigBee and ZWave smart home protocols, was installed and connected to the local area network.[2] Both ZWave and ZigBee are wireless protocols used for the control and communication of smart home devices. The Wink was connected through the network to an Alexa and then the Wink online server. When connected to the internet the Alexa was able to use simple voice commands to turn on lights and at one point, though it was not functional to demonstrate to us, to unlock the sliding door of the house. The eventual goal was to have sensors and motors on the windows to open and close them, as well as inform the central system when they were open. All of these features would ideally be connected to an app that would allow remote access to the features as well.

The major detriment of the system is the Wink itself, as it requires the internet to access almost all of its functionality. As mentioned before, access to the internet will not always be guaranteed and everything should be functional without it. Security of the system was also in question and had not been investigated at all. As well, with the wink using both ZigBee and ZWave protocols, there was unneeded complexity to the project, since either one alone could have sufficed for the uses of the house.

Ultimately this aspect of the project was least important, but more reasonable to install than other options. Thus it was made the secondary goal of the other project group along with the central system, which will be discussed in detail further in the report.

Water Reclamation System

The water reclamation system to be installed was the GENAQ S50 Stratus.[3] This system would pull water both from the outside and inside atmosphere but also from steam in the shower. With the ultimate intention of supplying all of the water for the house while still leaving the option for a direct connection to a water supply be made. The option for an additional connection was vital as the weather conditions play a heavy role in whether or not enough water will be reclaimed. Both reclaimed water and water pumped directly in would be held in a water tank underneath the floor of the house. In terms of systems integration the goal would be to be able to monitor the volume of the water tank, and if possible the rate of water being generated by the reclamation system.

However, the system itself was entirely theoretical aside from the water tank itself. Originally the reclamation system was to be ordered from Europe, however that plan was eventually scrapped in favour of creating a system from scratch. However this process was never started, as such no implementation of the system currently exists.

As the only part of this system existing is the currently unused water tank, there was little reason to implement and no ability to test the system to an acceptable degree, as such other systems were investigated.

Local Area Network

The local area network (LAN) consists of a simple wireless router, that connects to any wireless device in the house, as well to any outside wireless internet signal. In the current state of the house this allows use of the wireless LAN to connect to the Carleton University Wireless

internet. This system works well, and required no changes for the project. The biggest piece of information that must be taken into account when regarding this connection is that the internet must be considered optional for all systems. There are multiple systems, such as the EcoBee and Wink that were making active use of the internet access, which is a violation of one of the core design concerns of this project. As such going forward this link would be used exclusively for LAN connections with systems limited to house itself.

EcoBee

The EcoBee is a smart thermostat system that is cloud-connected.[4] An internet connection is required for the use of some of the smart features, since data is sent to the company's servers before enabling some features or performing a calculation. As mentioned earlier, this violates the requirement that an internet connection must be optional. The system itself if very intuitive and can be connected via home assistant, however it is ultimately not usable due to the internet requirement. This system will eventually have to be replaced if it is to be connected into the main system.

An alternate method of using this is to either purchase individual sensors and place them around the house to gather data, or purchase a new system that will not require an internet connection to communicate with the central system.

Solar Panels

The solar panels for the tiny house were installed before the beginning of this project, however they have not been fully operational during that time frame. They were described to be

operational but turned off until the rest of the energy and battery systems were installed. The solar panels themselves would provide information to a solar monitoring system, the Context ComBox Communications device, which would in turn pass any data to the energy management system (to be discussed further in the report).[5] The monitoring system itself could not, without modifications provide data to the central system, however the energy management system could provide that functionality.

Ultimately due to the solar panels not being functional and their data unreadable, this was determined to be a system that could not be connected at this time.

Battery Management System

The power either supplied from the solar panels, or by a direct connection to an energy grid, was to be monitored by an Orion BMS Jr.[6] This system, similar to the solar monitoring system, would keep track of incoming and outgoing power from the batteries. The batteries themselves would be the central point from where all power within the house comes from. The battery management system however, had no way to communicate with the central system without intermediary system, as such it would also be connected to the energy management system.

The batteries themselves were installed though not in use, and the battery management system was not installed at all. Thus, this part of the house could also not be connected into the central system.

Energy Management System

Both the solar monitoring system and the battery management system is intended to provide information the the energy management system known as TED, The Energy Detective.[7] This module would collect both of the data, and with its existing compatibility with home assistant send all that information to the central system. Once the system is connected it could be displayed in any way required. The exact nature of the data that would need to be collected was not obtained, however those working on the house would be able to retrieve that information.

This system could, in theory, provide most of the actual data for the system and be relatively easy to collect due to the compatibility with Home Assistant. Unfortunately neither the TED or either the battery or solar monitoring systems were implemented to actually retrieve any of that data. As such this system was not suitable to be connected.

Wall Sensors

The SMT wall sensors are not compatible with any other protocol in the house, and require a proprietary gateway interface to communicate with a Windows computer and relay the information stored in the data loggers.[8] The data loggers poll the wall sensors every ten minutes and store the data until they detect a working gateway interface. A picture of the data loggers can be seen below, in Figure 3.

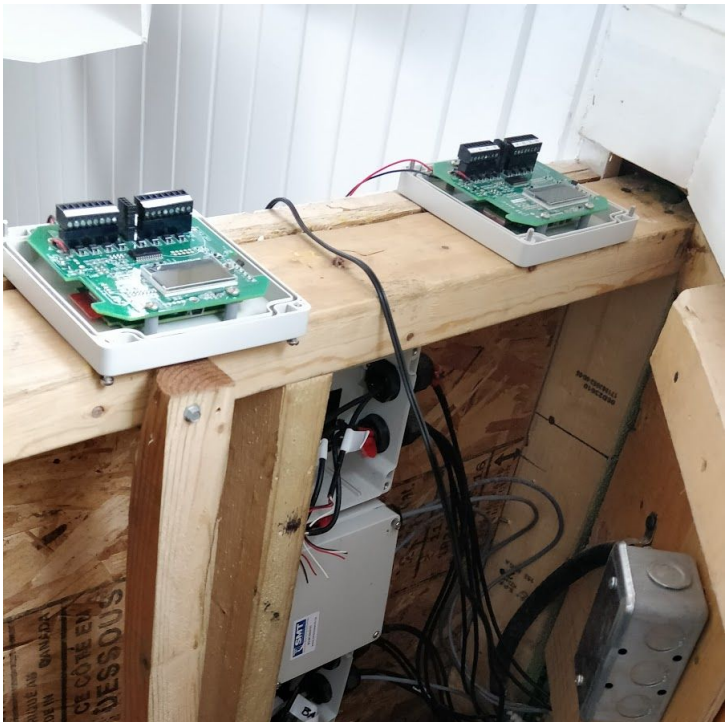

Figure 3: SMT Data Loggers Before Installation

There is no way to directly connect the loggers or the interface to a PC or otherwise running anything other than a Windows operating system. This was identified as a problem, since part of the goal was having a centralized system in the house where all data can be accessed and viewed, without the need for bringing an outside piece of equipment, like a laptop, into the house. This is shown in Figure 4, below.

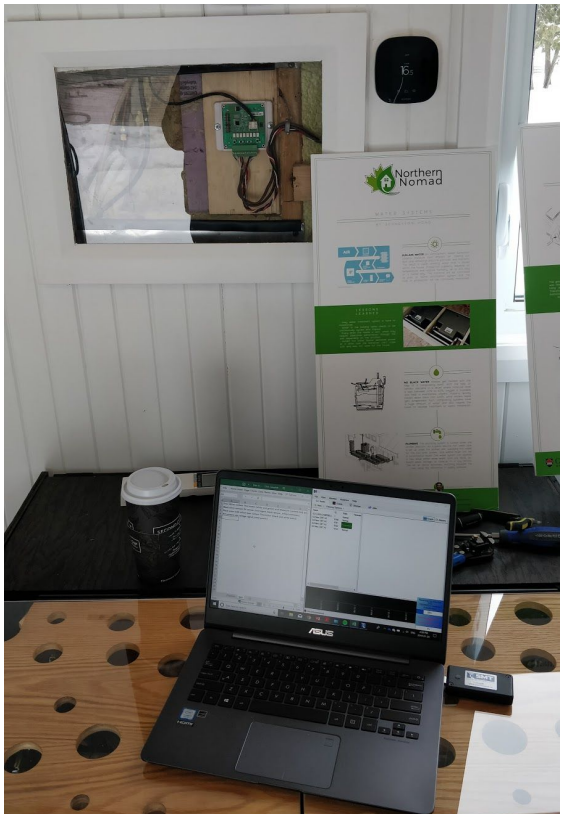

Figure 4: SMT Data Logger Interface Plugged into Windows Laptop

Depicted above, is a Windows laptop with the USB data logger interface plugged in. This allows for monitoring of the wall sensors through a graphical user interface. It was this scenario that the group intended to avoid and create a workaround for.

Whilst this system is not functional due to it's requirement for a Windows PC, it has the potential to provide the most useful information, as the data collected would be directly used for research. As well, there were many options to work around the Windows PC issue, as such this system was the ultimate focus of this project group.

Central System

The central system of the tiny house is where data from all around the house would be collected, formatted and ultimately displayed. Originally the closest thing to a central system was the wink paired with the wireless LAN. However as mentioned above the wink only worked with a select amount of smart features and required an internet connection. As such it was important to find a new system that would be accessible by many different devices and not require connection to the internet. To this end the Home Assistant platform was identified as extremely robust and compatible with a very wide variety of smart features and sensors, both proprietary and open systems. To host the home assistant, a Raspberry Pi 3, was chosen for it's small form factor but powerful and open source codebase.[9] Additionally the Home Assistant had an entire operating system, Hassio, specifically designed for the Raspberry, this allowed easy compatibility with the two systems and alot of freedom in development due it being open source.[10]

As the most important piece of this system, obviously the central system had to be a focus, as such the other group spent the great majority of time ensuring the system could be connected to a wide variety of systems. For more information regarding this system please see the report as written by the other project group.

4.2 Design Details

As mentioned in the above section, and as can be seen in the figure below and in appendix F.2, the focus of the project was placed on the wall sensors. The wall sensors provided the largest opportunity for substantial development, due to it's completed implementation and relatively simple access points, the physical sensor connections themselves. It also provides

research data that is currently being analyzed, as such having this system connect would be most beneficial.

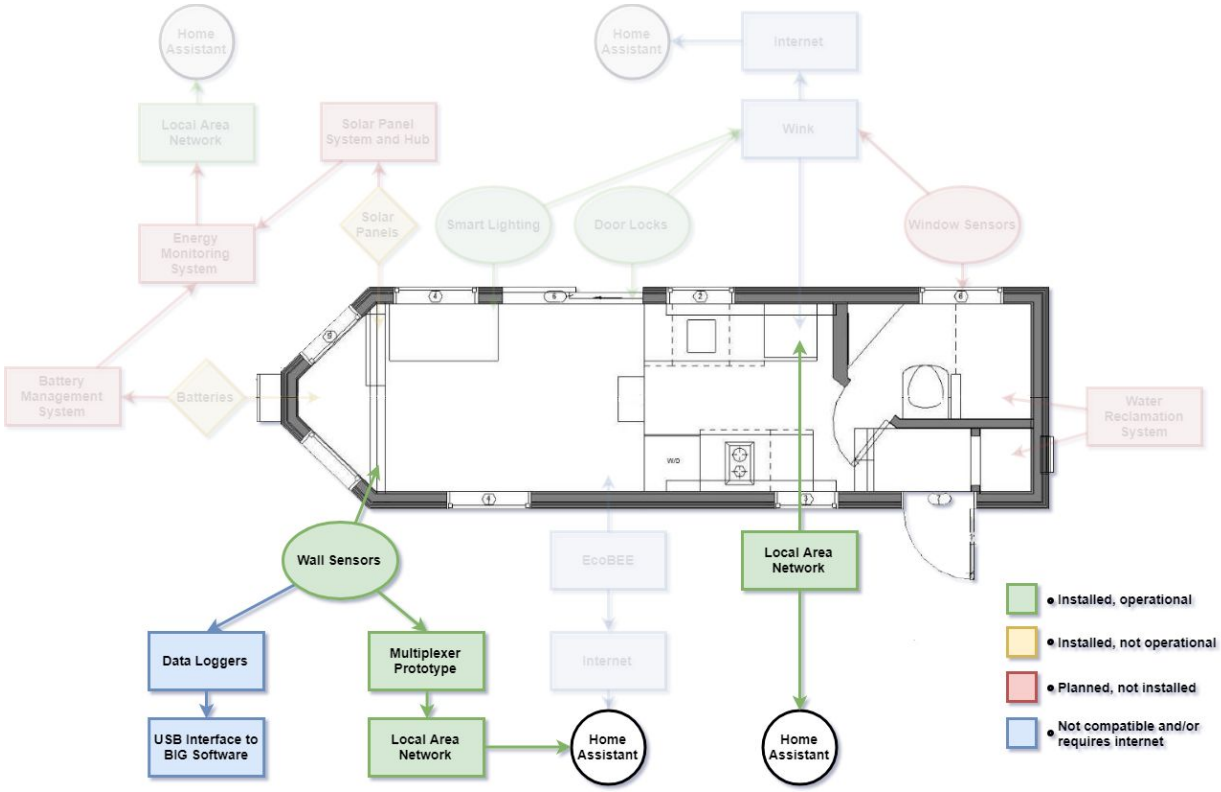

Figure 5: Overview of the Wall Sensor System.

The first step the team took was to do some research on the capabilities and limitations of the SMT wall sensors, the data loggers, and the accompanying software. The USB gateway interface device used to read from the data loggers requires running the Building Intelligence Gateway (also called BiG) software from SMT. Reading from the installed wall sensors meant that one of the team members needed to bring a laptop running Windows to the house, plug the gateway interface device in, and run the BiG software, to read any information from the data loggers.

As mentioned above, the centralized system was to be run on a Raspberry Pi, since it is cheap, easy to aquire, and runs the HomeAssistant (HASS.io) software quite easily. Since SMT doesn't provide a version of the Building Intelligence Gateway software for Linux systems, we needed to find a workaround. The first option that was explored kept most of the system intact, so it had the least risk involved. It was decided to try using a piece of software called WINE. WINE, which stands for Wine Is Not an Emulator, allows for installing and running some windows dependencies and libraries on a Linux machine, thereby simplifying the installation and running of some Windows-only programs on Linux. One drawback is that it causes some slowdown, but since the data loggers were only polling the sensors at a maximum of once every ten minutes, this solution seemed viable. The BiG software also ran only on x86 architecture. After doing some research, an x86 processor emulator was found that runs on the Raspberry Pi and tested to make sure it worked independently. The dependencies, including Microsoft .NET and SQL Server, were installed through both WINE and the x86 emulator, and the BiG software was installed without a hiccup. Running the software came up with an error, and after multiple days of troubleshooting, it was apparent that it would be more productive to take a different approach.

4.2.1 Project Final Design and Solution

Instead of running the software on the same Raspberry Pi that will be running HomeAssistant, it was decided that the gateway interface device would be circumvented entirely, which meant going around the data loggers as well. The new solution consisted of splicing into each sensor before it reached the data loggers. This is as simple as running a jumper cable from

the sensor to some interface where we can collect the data. There were twenty-eight analog sensors in the walls of the house, which monitor temperature, humidity, and moisture levels. The data collected would have to be converted into the required format/units using modified manufacturer provided formulas, and organized in the same fashion as the data outputted by the data loggers in conjunction with the BiG software. This new design was to facilitate the portability and adaptability of both systems. The data would have to be compiled into both CSV and JSON formats. The CSV file would be cross-compatible with the BiG software, and the JSON format would be used to transport the data in a readable format to the HomeAssistant Raspberry Pi for viewing.

In order to complete this new design, new hardware was required that fulfilled the needs of this design. Many options were considered, including an Arduino YUN to collect sensor data, since it had on-board WiFi, or an RF module, which could be used to transmit or receive radio signals between two devices. An ESP8266 WiFi module was chosen, since it can run some basic Arduino code and perform some simple calculations, as well as providing the capability of sending the compiled data to the HomeAssistant Raspberry Pi. For these purposes, it was best to acquire an ESP8266 development board, since it comes with a serial interface chip to upload test code, and breakout pins to connect the components for testing. A Raspberry Pi would be used to simulate the completed HomeAssistant Raspberry Pi and run a Python script that formats the received data into CSV and JSON formats.

The ESP8266 development boards that were acquired only had one analog I/O pin. This meant that to connect all 28 sensors at once, a multiplexing system would be needed. A few Texas Instruments 8-to-1 multiplexers/demultiplexers were bought, and built into a 32-to-1

multiplexer/demultiplexer. This uses 3 pins at the first level to pick which multiplexer to enable, and 3 pins at the second level to pick which sensor to read from. A prototype was first built using demultiplexing for proof of concept. This system used LEDs in place of the sensors to replicate sensor data acquisition. The prototype built can be seen in Figure 5, below.

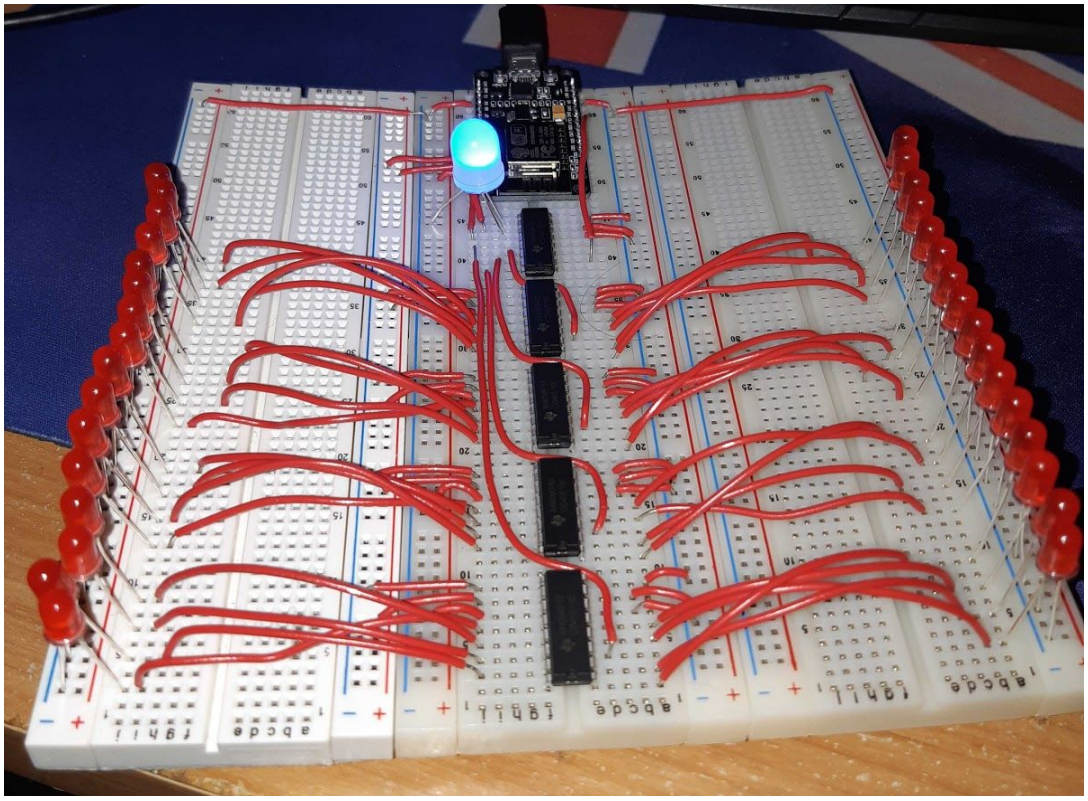

Figure 6: Prototype of demultiplexing system

The Arduino code to read from all 28 sensors cycles through each one, reading the value, and storing it every ten minutes. A computer aided designed wiring diagram for the prototype can be seen in Figure B.1 of Appendix B. The data is sent to the Raspberry Pi over the Local Area Network, via the MQTT messaging protocol to publish the results such that the HomeAssistant software can read from and display them. The MQTT protocol is a publisher subscriber structure, hosted by the Home Assistant. The Home Assistant will create a set of topics that an access point can subscribe to or publish to. In the case of this system, there will be

two topics, one to collect data and one to send that data. The Home Assistant will publish to the collect data topic, and subscribe to the send data topic, thereby listening for when the data is ready to collect. The NodeMCU will subscribe to the collect data topic, listening for when it is time to collect data, and publish to the send data topic, where the Home Assistant will collect the results. The System block diagram explaining the interactions between the NodeMCU and Raspberry pi, via the MQTT structure can be found below and in Appendix E.

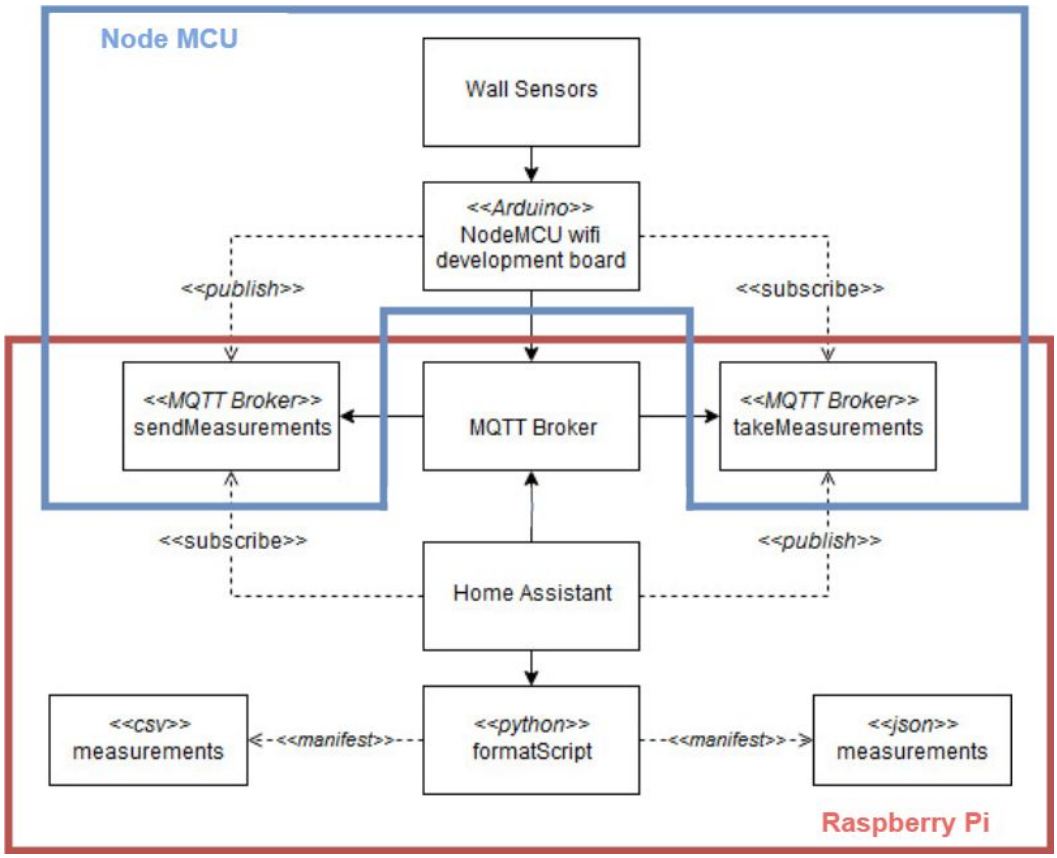

Figure 7: System block diagram, with responsibilities broken up by system.

The Arduino source code for the completed prototype can be found in Appendix C. Also, the source code to create an MQTT client on the NodeMCU can be found in Appendix D.

4.3 Components of completed prototype

The multiplexing prototype was built with components available at any electronics or hobbyist store. The components used are listed below:

- Breadboards: 3 breadboards were used as the means to wire our components together. Using 3 breadboards meant that we could evenly distribute the components across the boards for a less busy and simplistic design.
- NodeMCU ESP8266 WIFI Development board: An open-source firmware and development board was used as the main component to prototype the internet of things enabled multiplexing system. The reason why this component was chosen is that it is open-source, interactive,programmable, low cost, simple and WI-FI enabled as well as small in size. Given the budget and the Northern Nomad Tiny House's constraints, this chip was the best fit to allow wireless capabilities into the design. A program to poll from the sensors sequentially was uploaded to this chip, to carry out the functionality of the design. The NodeMCU ESP8266 Development board is pictured below in Figure 6. A more detailed pinout can be found in Figure B.3 of Appendix B.

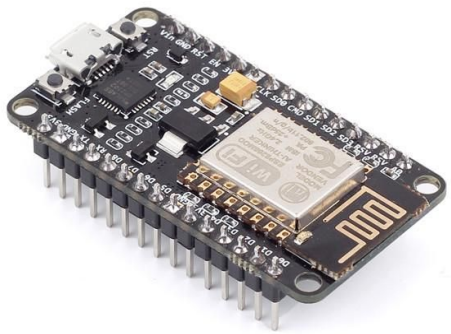

Figure 8: NodeMCU ESP8266 Development Board [11]

- Texas Instruments CB4051BE 8-to-1 multiplexers: The CD4051B is an eight-channel multiplexer that has three binary control inputs (A, B, and C) and an inhibit input. The three binary signals select one of eight channels to be turned on and connect one of the eight inputs to the output. As mentioned before, 5 of these multiplexers were used to create the polling system to accept 32 inputs, and feed them all into a common output. The Texas Instruments CB4051 BE is depicted in Figure 7, below. A more detailed pinout and block diagram can be found in Figure B.4 and B.5 of Appendix B.

Figure 9: Texas Instruments CB4051BE 8-to-1 Multiplexer [12]

In addition to these main components, other components such as Red LEDS, RGB LEDS, as well as various wires were used to complete the multiplexing prototype.

5.0 CONCLUSIONS AND RECOMMENDATIONS

5.1 Recommendations

There are a lot of potential extensions for this project. The major issue is that it requires the Tiny House to have installed the systems, and have them be operational and accessible.

An appropriate short-term goal is ensuring the wall sensor multiplexing board is printed to an appropriate circuit board, a casing found, and the wall sensors spliced into the board. This will allow the data to be easily sent to the Home Assistant for conversion and display.

Many other systems, such as the water reclamation, solar hub and battery management systems, would all be ideal candidates for integration. However there a no other systems that are currently connected and functional, to the point where it would be appropriate to connect them to the Home Assistant hub. This will obviously change as the Northern Nomad Team continues construction, so frequent check-ins would be recommended in order to keep as updated as possible.

In the meantime, a team of three or four Computer Systems and Software students could comfortably work on implementing other quality of life smart features into the home. This includes features such as, AC and Heating control, occupant tracking, automatic lighting, automatic locking and many other similar features. As well, since the house should be able to function with and without internet, a database and server system could be created to store, when offline, and upload, when online, any information collected for research purposes. This should also allow remote access to make analysis of the information easier, as is a primary purpose of the Northern Nomad Tiny House.

The current systems implemented were all done using a Raspberry Pi and Home Assistant. Unfortunately, near the end of the project, an update for Home Assistant was released, which caused the Home Assistant side of the project to become not functional. Extensive testing and attempts to get it working again were made, however, with no success. It is recommended that time be spent updating the Home Assistant side of the software to the point where it is

compatible with the current release. It would be advisable to stay with both a Raspberry pi and Home Assistant and continue to expand on them. This is to ensure as much cross compatibility as possible. Both the Home Assistant and Raspberry Pi are very common systems that have a lot of available information, which help with troubleshooting, and are compatible with a very large variety of systems.

5.2 Conclusion

The project goal was to connect the many different sensors of the Northern Nomad Tiny House such that they can be monitored, controlled and their data accessed from a central system. Specifically, the SMT wall sensors were the first goal.

Our solution involved the use of the smart home automation software, Home Assistant, and a variety of intermediary hardware and software depending on the system. The majority of the systems were not operational, as such, the system that was focused on was the SMT wall sensors. These sensors monitored humidity, temperature and moisture within the walls to test the effectiveness of the insulation. In order to receive data from the wall sensors a simple, multiplexing circuit was created and connected to a Wi-Fi module that sent the incoming analog signals to the Home Assistant via an MQTT connection.

At the conclusion of the project, the prototype of the circuit has been created and is functional for all twenty-nine wall sensors installed in the house. Though at the present time installing the prototyping board in the house is not appropriate as a more form fitting circuit board should be printed.

This project involved a lot of communication with multiple different members of the Northern Nomad Tiny House project team. As such there may be small but important bits of information that are not properly recorded, or exist only in some emails. If clarification is required it is recommended that contact is made with the Tiny House team or, in the case of details relating specifically to this project, Adam Staples at adam.staples@carleton.ca.

6.0 REFERENCES

[1]"Northern Nomad", *Northern Nomad*, 2019. [Online]. Available: http://www.thenorthernnomad.ca/. [Accessed: 09- Apr- 2019].

[2]"A Simpler Smart Home", *Wink*, 2019. [Online]. Available: https://www.wink.com/. [Accessed: 06- Apr- 2019].

[3]"Atmospheric water generator by GENAQ. Get water from air.", *GENAQ*, 2019. [Online]. Available: http://www.genaq.com/. [Accessed: 06- Apr- 2019].

[4]"ecobee | Smart Home Technology |", *Ecobee.com*, 2019. [Online]. Available: https://www.ecobee.com/. [Accessed: 07- Apr- 2019].

[5]"Solar Monitor System - Conext Combox | SE Solar", *SE Solar*, 2019. [Online]. Available: https://solar.schneider-electric.com/product/conext-combox/. [Accessed: 07- Apr- 2019].

[6]"Solar Monitor System - Conext Combox | SE Solar", *SE Solar*, 2019. [Online]. Available: https://solar.schneider-electric.com/product/conext-combox/. [Accessed: 07- Apr- 2019].

[7]"TED The Energy Detective", *Theenergydetective.com*, 2019. [Online]. Available: http://www.theenergydetective.com/. [Accessed: 07- Apr- 2019].

[8]*Smtresearch.ca*, 2019. [Online]. Available: https://www.smtresearch.ca/smt-product-list. [Accessed: 07- Apr- 2019].

[9]"Raspberry Pi 3 Model B - Raspberry Pi", *Raspberry Pi*, 2019. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. [Accessed: 07- Apr- 2019].

[10]H. Assistant, "Hass.io", *Home Assistant*, 2019. [Online]. Available: https://www.home-assistant.io/hassio/. [Accessed: 07- Apr- 2019].

[11]"NodeMCU GPIO with Arduino IDE | NodeMCU", *Electronicwings.com*, 2019. [Online]. Available: https://www.electronicwings.com/nodemcu/nodemcu-gpio-with-arduino-ide. [Accessed: 06- Apr- 2019].

[12] http://www.ti.com/product/CD4051B. (2019). *CMOS single 8-channel analog multiplexer & demultiplexer with logic-level conversion*. [online] Available at:http://www.ti.com/product/CD4051B [Accessed 6 Apr. 2019].

7.0 APPENDICES

Appendix A: Weekly Minute Breakdown

Week	Minutes (approx.)	Details
Sept 5 - 14th	0	Project selection and registration
Sept 17 - 21st	200	Initial meeting, project planning and writing of project proposal
Sept 24 - 28th	180	Submission of project proposal
Oct 1 - 5th	240	Meeting with precedent Nomad team, tour of Northern Nomad tiny house + planning and project management
Oct 8 - 12th	180	Researched relevant material. Tried to get Big SMT proprietary software onto RPI
Oct 15 - 19th	240	Tried to get Big SMT proprietary software onto RPI. Attempted many workarounds to bypass dependencies etc.
Oct 22 - 26th	N/a	Fall Break
Oct 29 - Nov 2nd	180	Acquired data loggers, tried writing scripts to pull data off them.
Nov 5 - 9th	180	More research and problem solving regarding the data loggers.
Nov 12 - 16th	200	Revising of plan in preparation of progress report, research of alternatives
Nov 19 - 23rd	200+	Worked on Progress Report
Nov 26 - 30th	200+	Worked on Progress Report
Dec 3 - 7th	120	Written Progress Report submitted

Dec 10 - 14th	200	Raspberry Pi testing, Attempting to explore other alternative and ideas.
Dec 17 - 21st	200	Raspberry Pi testing, Attempting to explore other alternative and ideas.
Jan 7 - 11th	200	Redesign of solution, preparation for Oral presentation
Jan 14 - 18th	240+	Worked on Oral presentation slides and script
Jan 21 - 25th	240+	Practiced and Rehearsed Oral Presentation
Jan 28 - Feb 1st	120	Oral Presentation made.
Feb 4 - 8th	120	Ordering of components
Feb 11 - 15th	180	Built and assembled prototype
Feb 18 - 22nd	N/a	Winter Break
Feb 25 - Mar 1st	180	Worked on Prototype and code
Mar 4 - 8th	300+	Preparation for Poster Fair + Worked on first draft for final Report
Mar 11 - 15th	300+	Poster Fair Preparation + Poster Fair
Mar 18 - 22nd	180	Worked on Prototype and code
Mar 25 - 29th	350+	Worked on Final Report
April 1 - 5th	350+	Worked on Final Report
April 8 - 9th	180	Final Report Submitted

Table A.1: Weekly minute breakdown

Appendix B: Wiring Diagrams and Pinouts

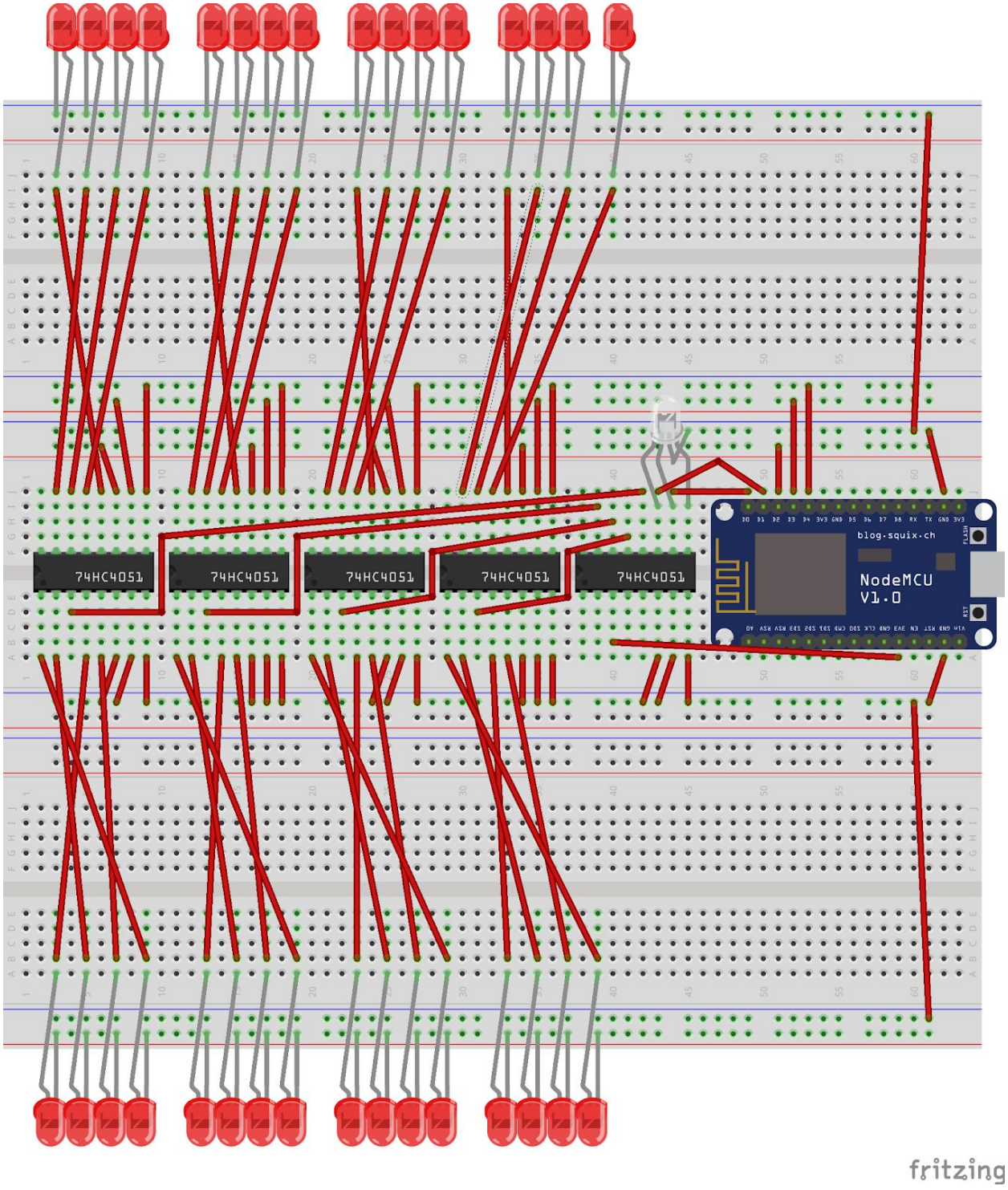

Figure B.1: Detailed Wiring Diagram for Multiplexing Prototype

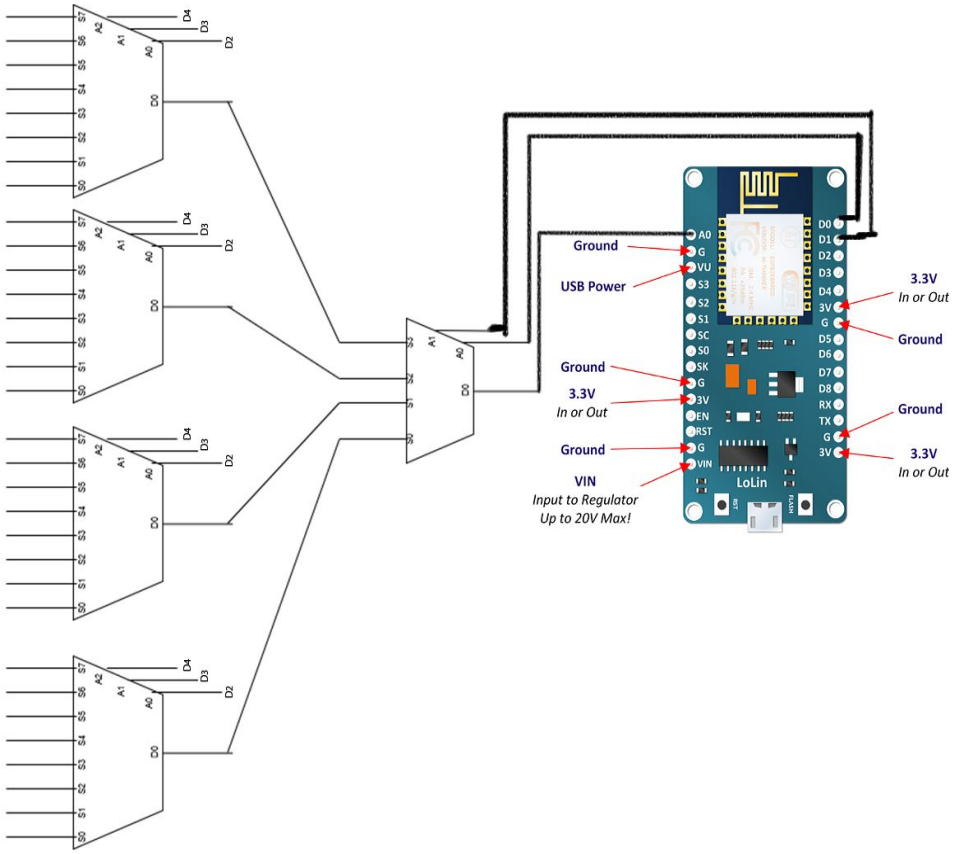

Figure B.2: High Level Wiring Diagram for Multiplexing Prototype

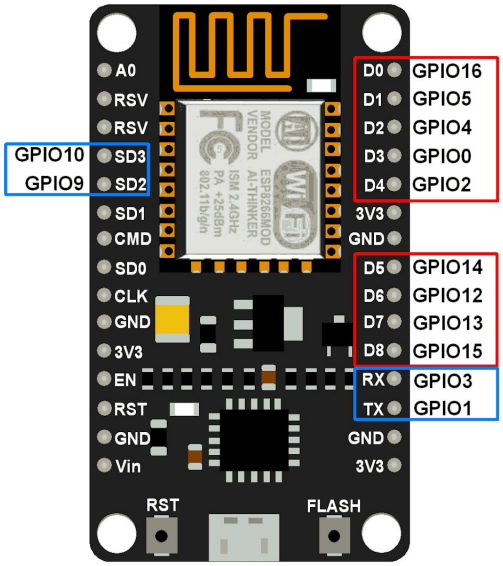

Figure B.3: Pinout for NodeMCU ESP8266 Development board [11]

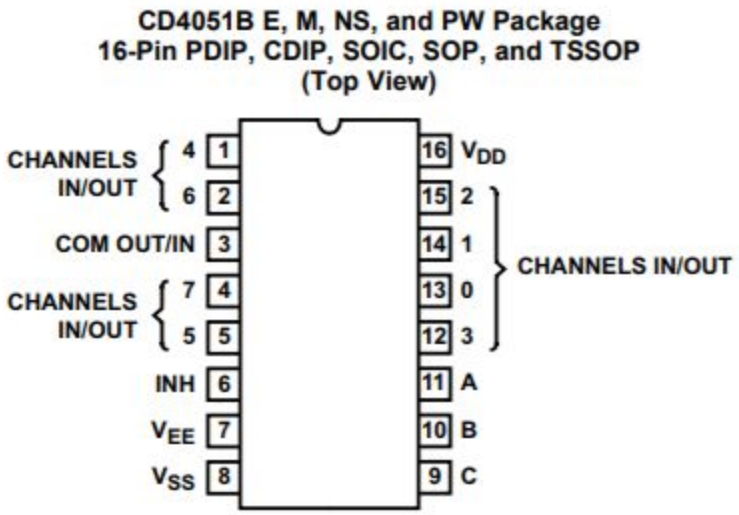

Figure B.4: Pinout for Texas Instruments CB4051 E Multiplexer[12]

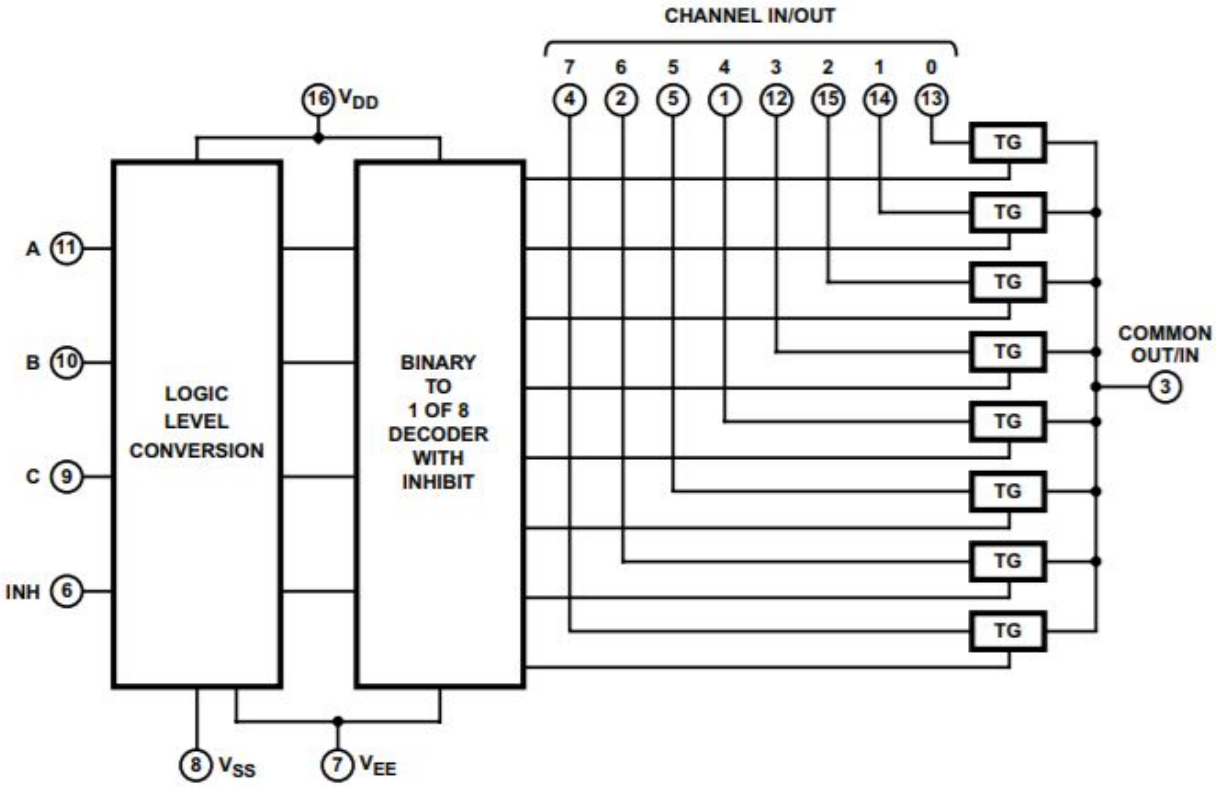

Figure B.5: Functional block diagram for Texas Instruments CB 4051 E Multiplexer[12]

Appendix C: NodeMCU/Mux System Source Code

Prototype NodeMCU and MUX system sensor polling code

```
//variable setup
int select;
int sensor;
void setup() {
  //Pin out setup, to ensure all digital and analog are ready for us
  pinMode(16,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(0,OUTPUT);
  pinMode(2,OUTPUT);
  pinMode(A0, INPUT);
  sensor = 0;
  select = 0;
  //show serial monitor
  Serial.begin(9600);

}

//main loop
void loop() {
  //switch the main controller mux
  if(select == 4){
    select = 0;
  }
  //switch the sensor that is being ready on the currently selected mux
  if(sensor == 8){
    select++;
    sensor = 0;
  }

  //bit mask the select and sensor variables to send the correct
  //digital outputs to the muxes.
  digitalWrite(16, HIGH && (select & B00000001));
  digitalWrite(5, HIGH && (select & B00000010));
  digitalWrite(4, HIGH && (sensor & B00000001));
  digitalWrite(0, HIGH && (sensor & B00000010));
  digitalWrite(2, HIGH && (sensor & B00000100));
  //short delay to show that reading is being taken
  delay(125);
  //print out to the serial monitor what is being read
  Serial.println("Mux #" + String(select) + " is being accessed for sensor
     #" +String(sensor) + ". The value is: " + String(analogRead(A0)));
```

```
  //update sensor at end of loop.
  sensor++;

}
```

Appendix D: MQTT Source Code (Created by other Project Group)

MQTT Server Source Code for NodeMCU

```
#include <ESP8266WiFi.h>
#include <Wire.h>
#include <PubSubClient.h>

#include "DHT.h"

#define DHTPIN 12 // what digital pin we're connected to NodeMCU (D6)

// Uncomment whatever type you're using!
#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);

#define wifi_ssid "dlink-7A14"
#define wifi_password "fjbbb95101"

#define mqtt_server "192.168.0.101"
#define distance_topic "sensor/distance"

#define TRIGGER 5
#define ECHO    4

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
  Serial.begin(115200);
  dht.begin();
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  pinMode(BUILTIN_LED, OUTPUT);
}

String macToStr(const uint8_t* mac)
```

```
{
  String result;
  for (int i = 0; i < 6; ++i) {
    result += String(mac[i], 16);
    if (i < 5)
      result += ':';
  }
  return result;
}


void setup_wifi() {
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(wifi_ssid);

  WiFi.begin(wifi_ssid, wifi_password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");

        // Generate client name based on MAC address and last 8 bits of microsecond counter
      String clientName;
      clientName += "esp8266-";
      uint8_t mac[6];
      WiFi.macAddress(mac);
      clientName += macToStr(mac);
      clientName += "-";
      clientName += String(micros() & 0xff, 16);
      Serial.print("Connecting to ");
      Serial.print(mqtt_server);
      Serial.print(" as ");
```

```
    Serial.println(clientName);


   // Attempt to connect
   // If you do not want to use a username and password, change next line to
  if (client.connect((char*) clientName.c_str())) {
   //if (client.connect((char*) clientName.c_str()), mqtt_user, mqtt_password)) {
    Serial.println("connected");
   } else {
    Serial.print("failed, rc=");
    Serial.print(client.state());
    Serial.println(" try again in 5 seconds");
    // Wait 5 seconds before retrying
    delay(5000);
   }
  }
}


void loop() {

    if (!client.connected()) {
      reconnect();
    }
    client.loop();

    // Wait a few seconds between measurements.
    delay(2000);

    long duration, distance;
    digitalWrite(TRIGGER, LOW);
    delayMicroseconds(2);

    digitalWrite(TRIGGER, HIGH);
    delayMicroseconds(10);

    digitalWrite(TRIGGER, LOW);
    duration = pulseIn(ECHO, HIGH);
    distance = (duration/2) / 29.1;

    Serial.print(distance);
    Serial.println("Centimeter:");
    client.publish(distance_topic, String(distance).c_str(), true);
}
```

Appendix E: System Block Diagrams

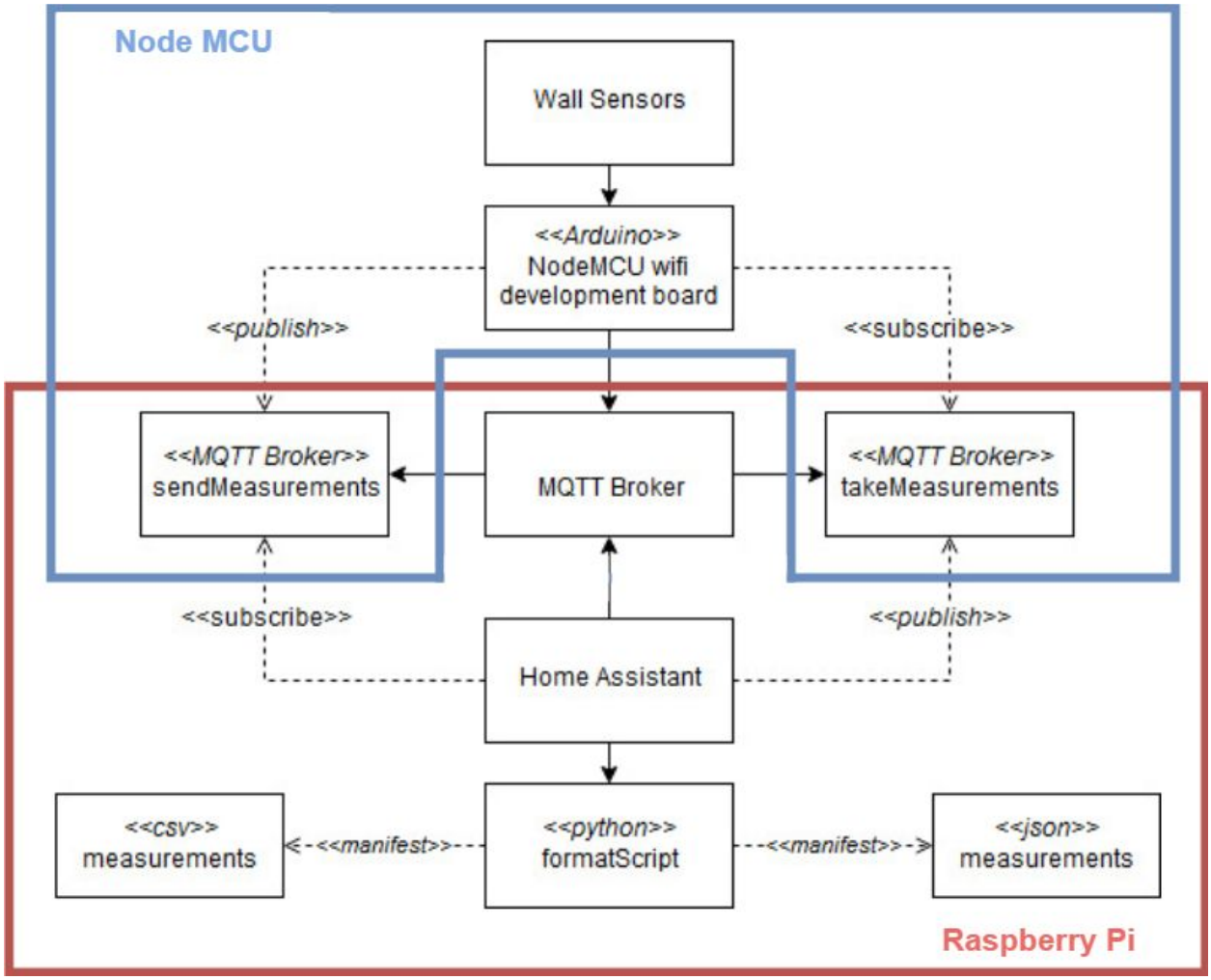

Figure E.1: System block diagram, with responsibilities broken up by system.

Appendix F: System Component Diagrams

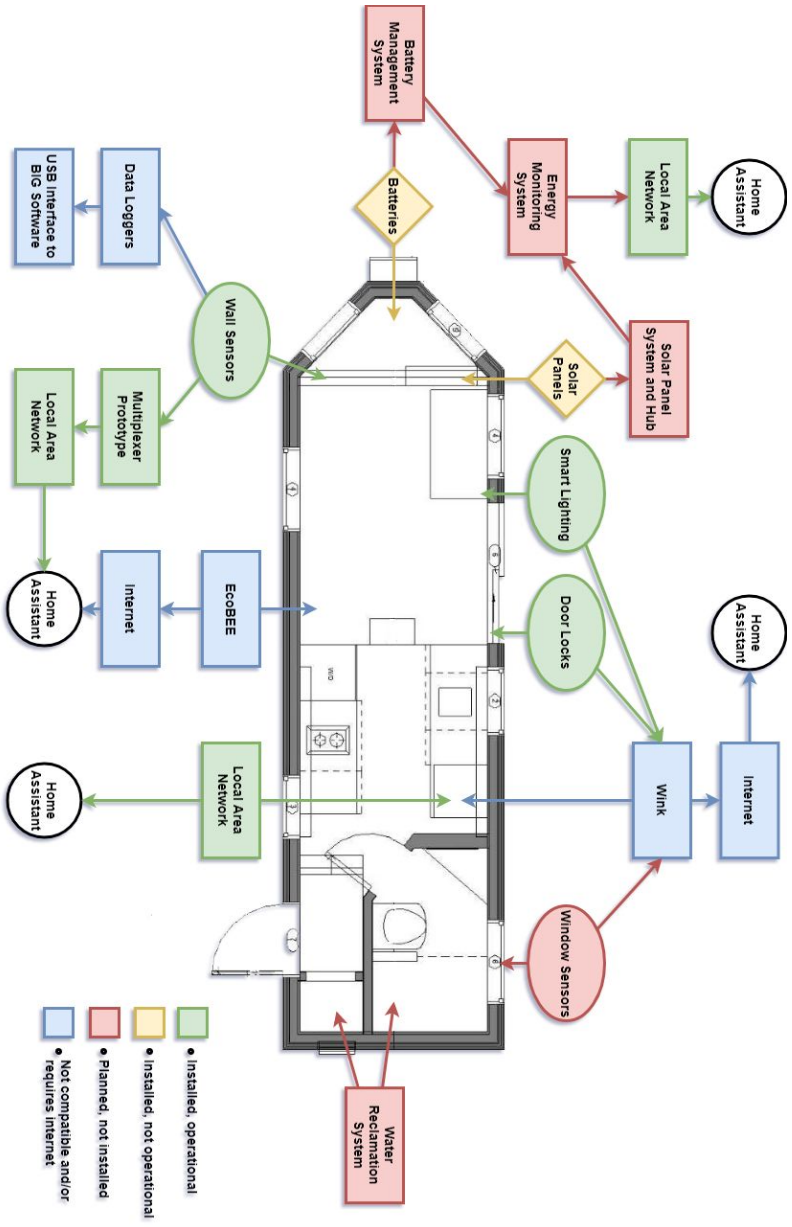

Figure F.1: Tiny House full System component diagram.

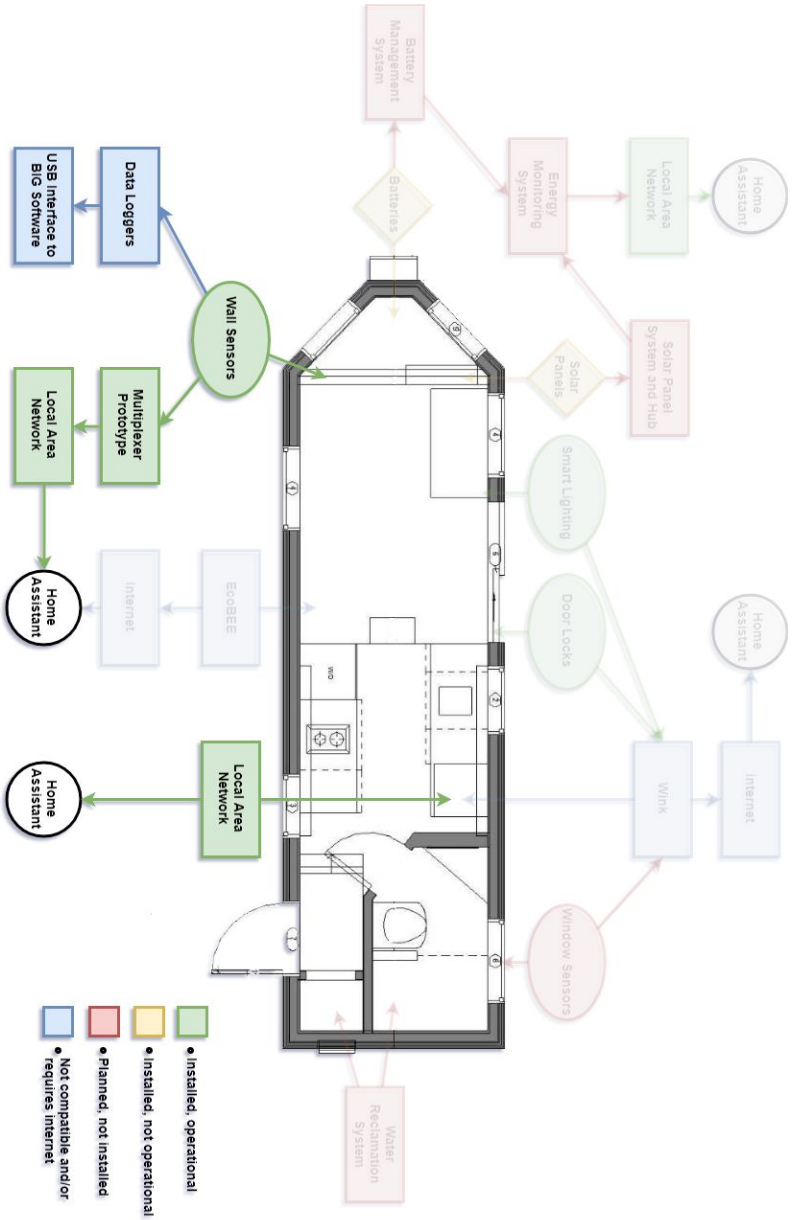

Figure F.2: Tiny House System Component diagram with important features for this project.

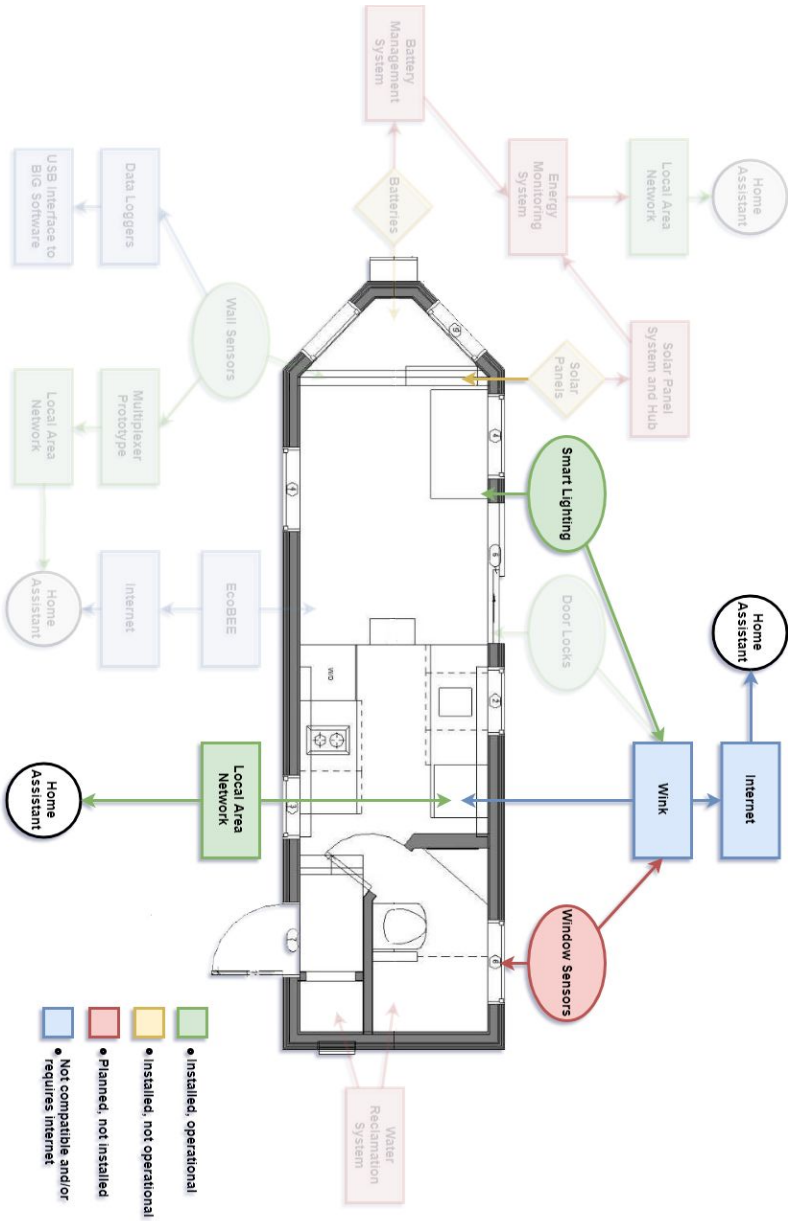

Figure F.3: Tiny House System Component diagram with important features for the other project group.