CSE 181A/EECS 159A

PowerAid: A Smart, Wireless Triple-Relay Metering Load Controller

A Project Plan

by Team 22: The SNAKbyters

March 18, 2016

Team Members:

Nina Tamashiro, Electrical Engineering

Sonum Hingorani, Computer Science and Engineering

Katherine Tran, Computer Science and Engineering

Aditi Bhatia, Electrical Engineering

Mentors:

Professor G.P. Li and Dr. Michael Klopfer

Calit2/Calplug in Collaboration with Smartenit LLC.

Executive Summary (5 points)

Through the Zero Net Energy Initiative and Policies, California aims to reduce the consumption of non-renewable energy sources and increase the use of renewable energy sources by 2020 [3]. By 2020, IoT will be a trillion-dollar industry worldwide as the current standing is a multi-billion-dollar energy management industry [11]. The prevalence of IoT devices are consistently growing in exponential value, and there is an increasing expansion in this field with development in many new product and designs, especially relating to energy management [18].

IoT smart-home devices have been increasingly marketed to the consumer population. Typically, the mention of smart-home devices relates to a load device of 120V, the standard for U.S. residential plug loads, but there is a great need for an IoT smart-home devices of 240V. For high loads, the wiring of the device is different from that of a 120V load. There are two relays, for high and low speed operation and one relay for neutral. The way in which the relays are turned on relates to the operational speed of the device. In this document, we refer to the capability of this device to work with 240V, and in doing so we are also referring to this wiring scheme. Because high load, high current devices have been less targeted to the consumer population, a higher load, higher current 240V device has been a niche market that has been greatly overlooked by the IoT market [1]. In collaboration with CalPlug and Smartenit Incorporated, an IoT company, this project design targets a product to monitor, manage, and control the power and energy consumption of 240V appliances. The main innovative component of the product includes the power load controller. Through tools and support provided by CalPlug and Smartenit, the demo controller is designed and built as well as the development of supporting software and network components. By creating a platform for energy management of large load IoT devices, this project tackles energy management and control in various atypical devices.

A full demo video of this project can be found at [Team 22 Full Demo 2016].

Table of Contents

Chapter 1. Introduction (15 points)

1.1 Motivation (5 points)

The major motivation of this project reflects energy conversation, management, and control. Energy efficiency has been a main target throughout the United States in order to increase economic prosperity by effectively utilizing energy resources to the fullest potential in order to diminish the carbon footprint generated [3, 4]. Within California, there has been an initiative to increase power efficient devices through policies such as the Title 20, which establishes appliance efficiency regulations to various products which include but is not limited to HVAC units as well as other electronic appliances, and Zero Net Energy Policies and Initiatives, which applies to all new residential and commercial construction to be zero net energy by 2030 [6]. In addition to these policies, there has been a popularized graphical depiction of the net load curve, colloquially known as the "duck curve," which displays the electrical demand on the power grid over a 24-hour period. From the Figure 1.1.1 shown below, California Independent System Operator, an organization which oversees the operation of bulk electric power system, transmission lines, and electricity market, has been able to determine the peak energy consumption times in order to reference non-peak energy consumption periods [2]. Through the use and application of our designed device, the user and the local utility are in a position to control, save, or reduce energy and loads throughout the typical day, especially in relation to the peak times [9]. The design goals of this project contributes to the goals outlined by the energy-efficient policies mentioned.

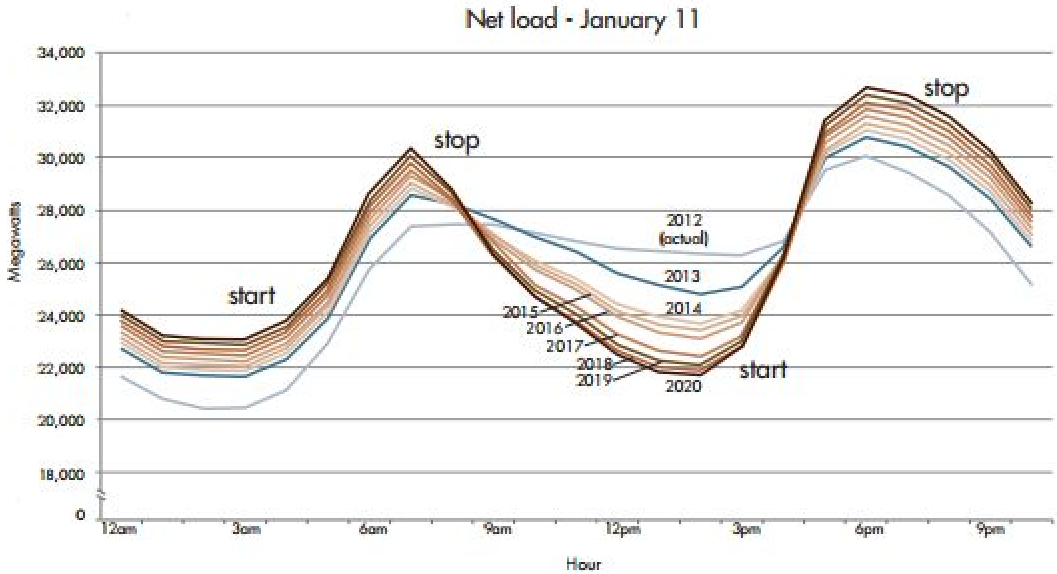


Figure 1.1.1: Peak energy consumption times,

In the scope of the project, the target appliance are pool pumps. In regards to the application of pool pumps, by measuring the peak times of electricity consumption as well as the amount of power and energy used by the pool pumps, the device is scheduled to avoid the peak energy consumption times, which in turn lowers cost.

1.2 Potential Impact of Work (10 points)

1.2.1 Relation to EE, CpE, and CSE (2 points)

Through the application of electrical engineering circuit design concepts and computer science concepts in algorithms, data analysis, and computer networks and architecture, the IoT spectrum encompasses a vast majority of disciplines which integrate amongst one another to create “smart” integrated products. Within the scope and design of this project, the hardware plays an integrated part in the design of our circuit. The design of the printed circuit board (PCB) is finalized. The implementation of the project is a proof of concept with the initial design and scope of the hardware to demonstrate wireless control of high loads. The PCB is going through the manufacturing process and is to be included in the project after being manufactured. The firmware aspects of the project provide a means to display the immense compatibility and capability of the load controller as each acts as a platform to control mechanisms of the PCB.

The software relates to the computer science aspects as it provides control to the user entailing network protocols, latency, data storage and analysis, and other software engineering principles. The network protocols act as a form of communication among the devices, and for this project the main requirement is for the project to be cloud based messaging protocol, all the while minimizing latency with commands exchanged among the device. Stretching further into the computer science fields, informatics principles can be used to analyze the data in order to provide further control to the device. Essentially, the IoT metering load controller is built on the basis of a dynamic concept design. The device can be integrated with various other pre-built firmware and software designs in order to manage power consumption in various different applications. Because of the dynamic capability of the design in the load measurement of the project, electronic devices of various different spectrum up to 240V can be used and monitored through this project.

1.2.2 Global Context (2 points)

Since interest in energy efficiency has been overwhelmingly increased throughout the world, energy efficient regulations, policies, and products are key to global competitiveness as energy and power consumption exponentially increase. Currently, in all of North America and some regions of South America, the residential plug load voltage standard remains 100-127V. In large group of remaining regions, which include the vast majority of Europe, Asia, Africa, Australia, and the other parts of South America, there is a difference in the residential plug load voltage standard of 220-240V. This distinction in the standard has resulted from a desire to increase power through the decrease of voltage drop from the same copper wire diameter. Because the use of higher standard voltages of 220-240V in these countries, the application of this project becomes highly applicable to a broader context around the world [16]. With the continuing broadening horizons, IoT applications have become a new invested method in which people are providing efficient, effective solutions to power management and control [9].

1.2.3 Economic Context (2 points)

The IoT industry is an expanding field within the domains of energy consumption, management, and control. Most of the IoT smart-appliance industry has targeted devices such as the smart light bulbs, security systems, and thermostats, but this project targets the application of IoT to 240V, high current load control devices. The major appliances that can be integrated with the control system of this project would be HVAC units, pool pumps, and heaters within a typical household. Although the main appliances being targeted within this project entails HVAC units, pool pumps, and heaters within a typical household, there is a large network of appliances that can relate to the design and standards, within other household appliances such as refrigerators, chillers, washing machine and dryer, EV chargers, and electric fireplaces as well as industrial machines such as circulation and control in spas, chillers in large factories, aquarium water pumps, and much more. Also, as mentioned in the global context, many regions have 220-240V standard, which entails all products in this region. The project provides a basis for the integration of these 240V devices in order for these devices to become a major player in the IoT smart-appliance domain. The broader impact of this project touches upon the idea of creating complete integration of all of these large, load devices in order to find a best-effort solution to highly monitor, conveniently control, and thoroughly reduce power consumption [10].

1.2.4 Environmental Context (2 points)

Electricity is generated by various fuels. Fossil fueled electricity generation release carbon dioxide, sulfur dioxide and nitrogen dioxide, which contribute to air pollution and damage the environment [10]. Nuclear fueled electricity generations creates radioactive waste, which is costly to manage and monitor. Other fuels for electricity generation also release other hazardous waste. Thus, reducing power consumption and usage is directly related to the decrease in greenhouse emissions and other hazardous waste materials. In order to combat the environmental footprint, the goal is to decrease power consumption overall by monitoring and analyzing the timings in which appliances are used and scheduling certain appliances to run in non-peak hours of the day. The trends in the graphical data allows for further data analysis to notice any particular trends in order to utilize the amount of power consumed. This project minimizes power consumption, which therefore reduces the negative environmental impact, and enables a convenience factor for our users to be environmentally friendly [5].

1.2.5 Societal Context (2 points)

Through the visibility of power usage and consumption, there is increased awareness and advocacy for an efficient power mechanism for the management and control in the energy of devices [7]. This IoT device changes how individuals interact with their devices, as their social awareness of their own power consumption, management, and control. This project provides a means for the consumer to control and monitor the power usage in their devices, which expands into the home appliances and utilities. Furthermore, one possible application would be allowing utilities to monitor various neighborhoods and cities in order to monitor and combat the high pulls in the electrical grid as shown in the net-load curve. This would create a widespread discussion on the roles of utilities and the involvement it has on power consumption and management.

Chapter 2. Background and Related Work (10 points)

2.1 Background (5 points)

The increasing new advancements in IoT technology enables the low-cost production and development of new devices and applications. This project utilizes an IoT edge device that acts as a controller to monitor power. The IoT edge device consists of triple-relay load controller integrated with a System on Chip (SOC), wifi chip, in order to connect with an application platform which allows the user to monitor and control the 240V device. The load controller measures the power delivered to the load through a current sensor and can report various parameters such as real and apparent power based on industry standards. The triple relays eliminate the need for additional controllers when handling two-speed appliances such as pool pump motors. The System on Chip is an integrated 802.11n standard compliant WiFi chip, which acts as the master chip within the printed circuit board, thus reducing providing a more efficient means of control. The WiFi chip connection is based on the firmware of the Espressif ESP8266. The integrated messaging network protocol is Messaging Queueing Telemetry Transport (MQTT), which has no standard encryption. This project was able to implement encryption to create a protected standard for the messaging. The software creates a platform to control and monitor the status of the device as well as display the graphical data and analysis that is detected by the load controller. By having a WiFi chip that also acts as the controller, we reduce the number of components needed, thus reducing our bill of materials and the complexity of the PCB.

2.2 Related Work (5 point)

Although there are a vast majority of IoT power control and management devices, there remains a small percentage of devices that is accommodating to 240V devices. The closest competitors would be those companies interested in IoT smart devices within the 240V domains. Within the United States, the major direct competitor is Insteon with their dual relay load controller. Other smart-home competitors that are within the 120V domain are Samsung and Honeywell with their integrated devices [8, 12, 13].

The advantages of our IoT edge device is the MQTT protocol which is standard, the WiFi range, and the industrial design. There are major distinctions in which this project provides a better solution to the power control and management of 240V devices. Most smart-device systems require an integration within specific devices that meet certain standards, regulations, and protocols. Thus, one advantage is in the compatibility of our product. Zigbee has been a major player in home automation system, lighting systems, and utilizes as it creates a mesh network for which low-powered devices, often battery powered devices, can interact and communicate with one another. Since most hub devices rely on Zigbee, Zigbee integrated devices are mainly only able to be compatible with each other [17]. This is the major component within the Samsung and Honeywell products [12, 13]. The Insteon device is mainly compatible within their own hub [14]. This project utilizes a generic SOC wifi chip, which not only enables a more scalable integration with vast amounts of products for multi-platform compatibilities but also enables our device to have longer, extended range of communication on high load, high current devices, thus differentiating this product from competitors. In addition, because of the simplistic compatibility and easy application of our device design, the bill of materials of our device is lower since other competitors require a hub of integration of their own, set product. Another benefit is demonstrated in the triple-relay of our product. The triple-relay aspect of this project device plays an essential distinction in order to create finite control in multiple speed devices. This is a major differentiation from the Insteon device since it is dual-band, which does not allow them to control multiple speeds [15].

Chapter 3. Project Design (30 points)

3.1 System Block Diagram (5 points)

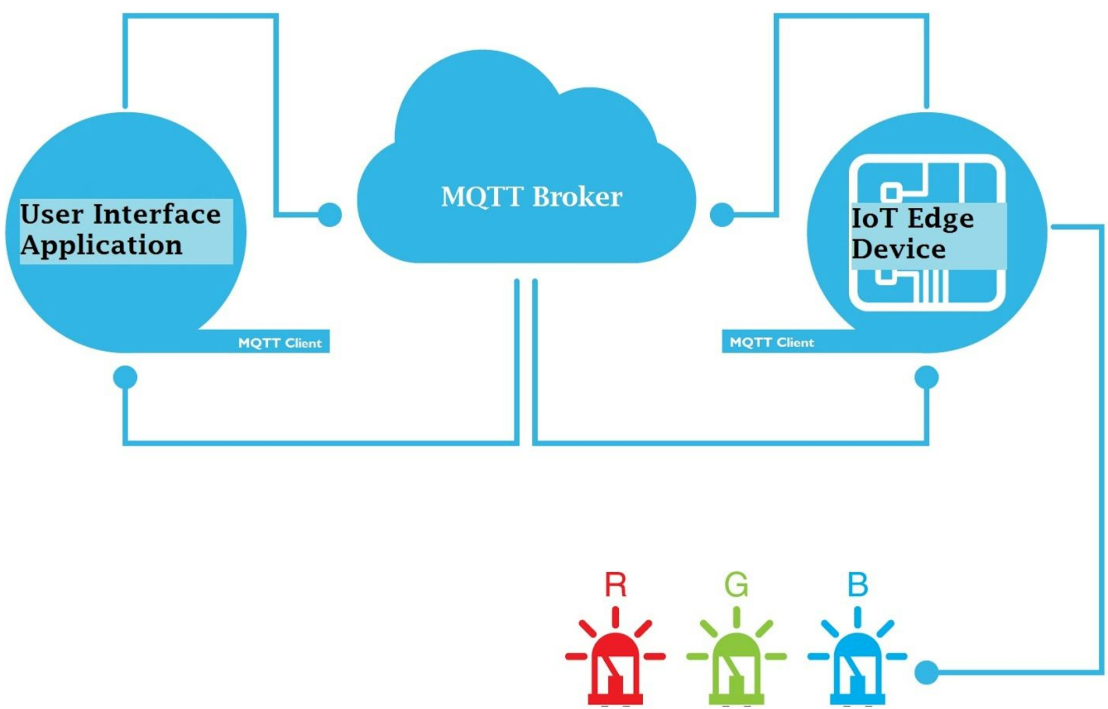


Figure 3.1.1: MQTT connections, between the broker and the clients.

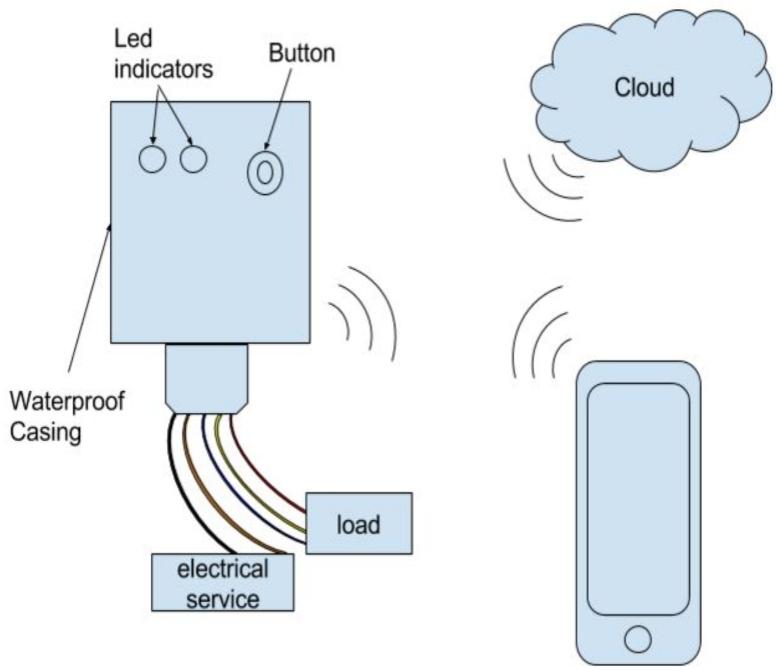


Figure 3.1.2 (on the left) displays system view, which includes the 3 main components: hardware (load controller), firmware (cloud server), and a representation of the software application

Figure 3.1.2: A system view of our device: hardware, firmware, software. Note: *the Cloud is a representation of our MQTT broker, which allows for other cloud-based devices to interact with the MQTT broker.*

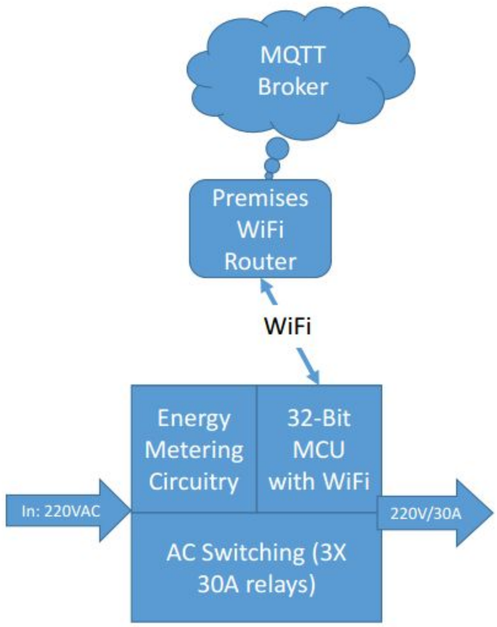


Figure 3.1.3 (on the left) displays the system level of the firmware design of this project.

Figure 3.1.3: A system level of our device: firmware.

3.2 Subsystems (15 points)

The system architecture of the device are broken down in three subsystems: hardware (load controller and wiring), firmware (SOC, WiFi protocols, network connection with the cloud server, relay of data through networks), software (demo software application to graphically display data and unit of user interface and user control of the device). Figure 3.1.2 displays the system architecture.

The first subsystem, on the far left of Figure 3.1.2, is the hardware subsystem consisting of solely the IoT metering load controller. The load controller is the most integral part of the entire system since this is where the input is taken to be read, processed, and controlled. The internal mechanisms of the Load controller are broken down in Figure 3.1.3, which is the block diagram for the load controller. The load controller interacts directly with the load and electrical service, connected to both via wires. The controller completes the connection between the AC current from the electrical service, with some added components and connections in between. These components and connections will allow for the user to control the power to the load from the electrical service, and monitor the power consumption of the load. The main components within the load controller include a SOC, microcontroller and WiFi module, to take and transfer data input, a metering chip which monitors power consumption of the load, a push button for user interaction, and LEDs (push button and LED details in next section) to provide status on the controller. (Note: The SOC is an integrated chip containing a microcontroller and WiFi chip. Thus, the SOC is split between the two subsystems of hardware and firmware.)

The second subsystem, at the top of Figure 3.1.2, is the firmware of the device which includes the SOC WiFi chip, MQTT broker, network protocols, server connection, cloud storage and database. The SOC chip contains the WiFi module, which is the main component for the communication of the data transferred from the load controller to the server. The WiFi module allows the IoT edge device to connect to surrounding WiFi by initializing the edge device to access point mode. Once a WiFi configuration is initialize, the default WiFi is configured to the device. The Wifi module is able to connect to secure WiFi network thus providing a layer of security to the edge device. The MQTT broker, the cloud server, is a processing point for the commands and data provided by the IoT edge load controller device to the software application and vice versa. MQTT, Message Queuing Telemetry Transport, is a messaging protocol that uses a publish-subscribe method between the MQTT broker and client(s). Through the use of network protocols and connections, the MQTT broker processes the data from and to the MQTT clients, load controller and demo software application. The MQTT messages are also encrypted in AES-128 to add a layer of additional protection among the transmitted messages. Within this project, there was no pre-existing AES encryption methods for MQTT with the WiFi chip. Therefore, a new software library was generated in order to create the AES-128 ECB encryption on the transmitted messages. The mode of encryption chosen is electronic codebook (ECB), which was adequate since the transmitted messages are solely 16-bit strings. In order to successfully encode and decode the transmitted messages, the messages must be associated with the appropriate key and generated code, which used base64 to further cipher the message. Otherwise, the messages are not deciphered properly. There are two keys required for encryption and decryption, one on both sides of receiving and transmitting the data. Our current implementation has a tag on the IoT edge device and software application with the private key value printed on it. Future implementations of the encryption design would include protecting the link using initialization vectors and generating an IV for protection of future communication on the link, and preventing a security breach in the system if the printed key on the Edge device is compromised. This added layer of security allows further protection for the MQTT design in order for the IoT edge design and software application to not respond to garbage commands and only respond to known commands.

The third subsystem, bottom right of Figure 3.1.2, is the software application.. The software application is a demo application with the sole purpose of showing the capabilities of the device. The software application is the user interface, which displays the data graphically in order to show the different power trends of power consumption, usage, efficiency, and performance as well as gives the controls and settings of the device. Through the software application, the user can set timers for when the load should turn on and off, view power consumption data, turn the load on and off, and view all devices connected to various load controllers. The software application is coded in the Processing 3.0 environment. Processing allows for a quick mock-up interface application for computer-based applications. In the implementation of this project, the Processing web-based application is manipulated and generated also as an Android application. Thus, the software application is able to run on a computer desktop as well as an Android tablet and phone.

3.3 Interfaces (5 points)

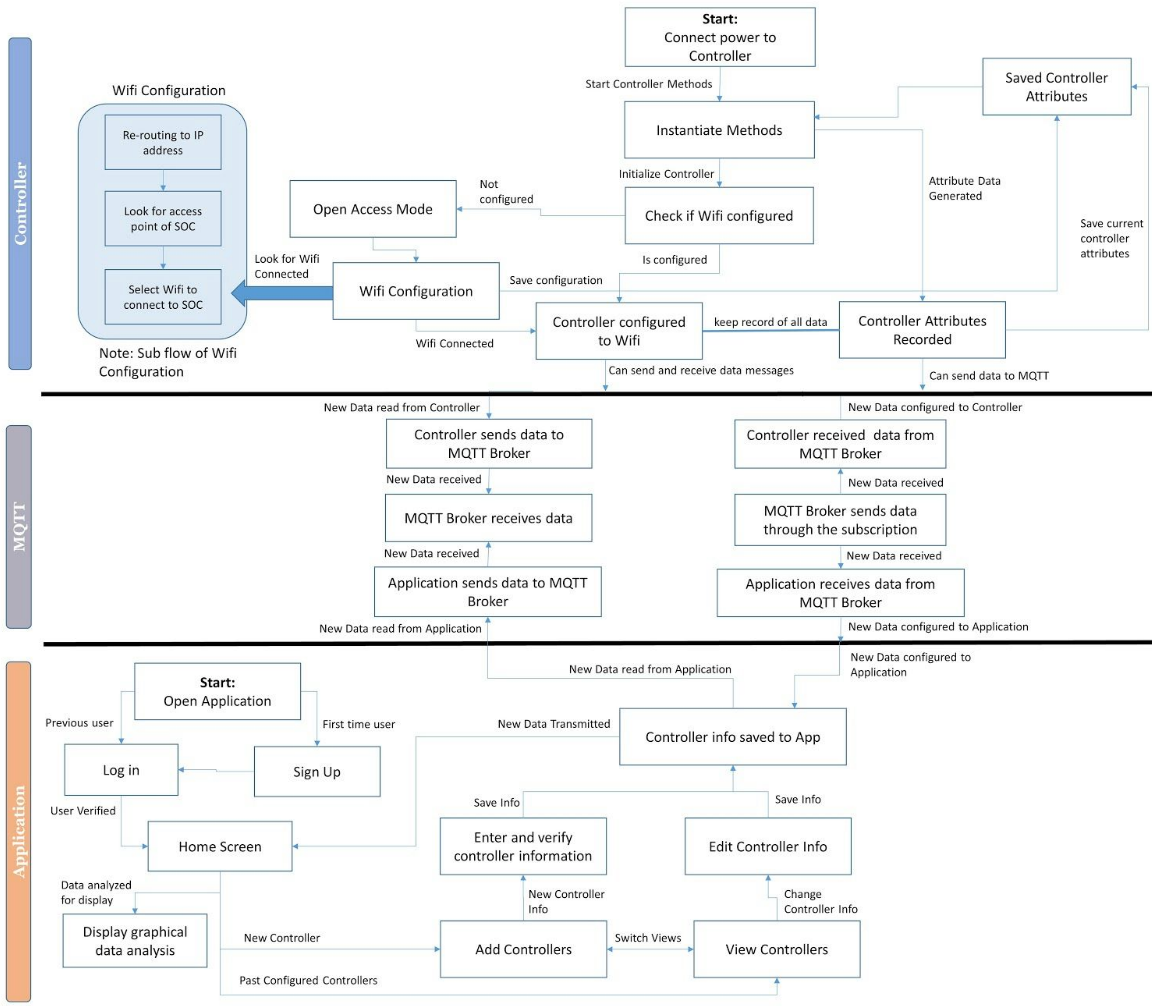


Figure 3.3.1: The UML Diagram displaying the interactions among the software application, MQTT layer, and the finalized IoT edge device

Subsystem 1, the load controller shown in Figure 3.3.1, interacts via the user and environment through a push button and LED indicators. These indicators allows the user to understand the mode of the controller while the push button allows the user to change settings. One LED is the Load Status LED, which communicates whether the load is energized. This LED has two colors, green and red, and is green if there is high power, red if low power, and flash green/red if both loads on. These Load Status LED signify the high-low speeds of the pool pump device.

The second LED is the Wifi Status LED, and indicates communication between the controller and the network. The Wifi Status LED fast flashes to indicate the device is searching for a network, and display a solid color to indicate it has joined a network. The period in which the WiFi chip is searching for networks is known as access point mode. During the access point mode, the WiFi chip acts as a router and searches for any available WiFi in the area. Once the specified WiFi is chosen, the IoT edge device saves the WiFi credentials as default and joins the network.

However, should the LED fast flash for 5 seconds, this means bind to Load 2 Out. Binding to Load Out references binding the controller to other devices, and can be done for both Load 1 and Load 2. The status LED will have a short blip and pause for 5 seconds in order to specify bind to Load 1 Out. The status LED flashes 2 blips per second in order to indicate the controller has been disconnected from the network. When the status LED flashes once per second, this specifies that an open permit to join to access mode, meaning that the device serves as a router and allow other devices to connect to it. The push button can factory reset the device, bind to other devices, and serves as a local control point. For a factory reset, the user should press the button while powering the device and then release. If the device has already been connected to a network, the button should be pressed for 20 seconds, after which the Status LED will go out for 1 second before beginning to rapidly flash (indicating that it is searching for a network). In order to allow for a soft reset, the button should be pressed ten times, after which both LEDs will turn off for a second, before the device rejoins the network. To leave a network, but preserve meter calibration settings, the button should be pressed for 15 to 17 seconds, and the Status LED will begin fast flashing. Requesting firmware updates is done by pressing the push button 6 times, and the Status LED will blip off every second while the update is in progress. To enable control for networks, the button should be pressed for 10 to 12 seconds, and the status LED will blink 3 times to denote success. For an open permit join, the button should be pressed for 6 seconds, and will allow other devices to join for up to 4 minutes. If a close join to a network is desired, the push button should be held for 3 seconds while the device is in open join. To bind the controller through Load 1 Out, the button should be pressed once quickly and then within ½ second, pressed and held for 3 seconds. For Load 2 Out, this process is repeated except pressing the button twice quickly instead of once. Finally, pushing the button will cycle the from an Off state to Load 2 On, to Load 1 On, with each press of the button.

This subsystem will in turn communicate to both subsystem 2 and subsystem 3. It will communicate power consumption data through firmware and network protocols to the cloud (subsystem 2) and current status of the load and controller to the software application (subsystem 3). This subsystem will communicate this information via a WiFi module.

The user does not directly interact with the second subsystem, the firmware, network protocols, and the cloud. This system interacts with the other two subsystems. It receives data from subsystem 1, the load controller. Once the data is processed, it sends data to subsystem 3, or the software application. In turn, this subsystem can receive data from subsystem 3 as well, in order to analyze user preferences.

The third subsystem, the software application, will be the primary interaction point for the system to the user. Through the application, the user can communicate to subsystem 1(after initial setup). The application will send commands to subsystem 1 on behalf of the user the status of the controller from subsystem 1. Additionally, the user can view data from subsystem 2. The application can also communicate data to subsystem 2, so that it may analyze user responses to power consumption levels.

3.4 Technology Options (5 points)

The main technology options were system on chip (SOC), circuit schematic and design tools, and programming languages.

With the choice of the system on chip (SOC), we chose to go with a WiFi supported chip because of its capability, compatibility, and support. There are different types of chips within the wifi or bluetooth domain, but we choose the SOC because this was a chip in which our corporate sponsor was more familiar with and had better reference material for the SOC. Among the SOCs, the chip of choice was the Espressif ESP8266 because of the reference material and community support as well as the low-cost and efficient performance. The ESP8266 chip also has a high receiver sensitivity and is programmable with the Arduino IDE, which we have chosen to use for this project.

For the circuit schematic and design tools, among the different tools of choice between EagleCAD and Cadence, we are working with EagleCAD since it was an available, no cost option that gave us the capability to design the needed schematic diagram. Microsoft Visio is also a tool that was used to build the basic block diagram and schematic for the basic circuits as well as the software application.

For the firmware on the SOC, among the various programming languages, we initially narrowed it down to C or C++. We have chosen to program in C++ since this language is supported by the SOC and is an object-oriented language. Our mentors, Smartenit, made it a deliverable to set up the environment with Eclipse. While we were able to get an software environment running in Eclipse, which was intended to be one of our deliverables to Smartenit, there was a compatibility issue with Eclipse in C++ for the SOC. In the end, the software is implemented in Arduino C. The software application is a basic demo application that displays the possible potential of this project. For the design and implementation of demo software application, the implementation is done in Processing in order to generate a user interface for the IoT edge device. In future implements the project is to use MEAN stack and MongoDB to work with the data and display the data graphically.

Chapter 4. Evaluation (20 points)

4.1 Experimental Setup (5 points)

The initial design and scope of the project is to be utilized for high load devices of 240V such as pool pumps and HVAC units. Given the large dimensions of the subject devices, the demonstration of the project needed to be further constrained in order to be presented and easily moved. Thus, our demonstration is a proof-of-concept device, in which two loads connected to the device to simulate the high-low speeds of the pool pump. Within the demo, the setup consists of three relay controllers. Two of the relays are connected to separate loads, and the third relay is connected to neutral ground. If one of the relays are switched on with the third neutral relay, each of the respective loads are switched on. This is to exemplify the low speed of the pool pump. If all three relays are switched on, this exemplifies the high load speeds of the pool pump.

For these loads, we use two lamps with incandescent light bulbs, which are used as resistors. We specifically chose incandescent light bulbs because they have no harmonic noise and have a unity power factor, causing these bulbs to have a purely resistive factor. Each of the two lamps represented our two loads, Load 1 and Load 2 respectively. Current calibration was done in order to provide accurate current measurements. From our system, data was collected from three different circumstances: only load 1 on, only load 2 on, both loads on with varying light bulb wattage. Our demo uses 120V to power the two loads, however, this demo will also work with a 240V load. The different light bulb wattage is used for this purpose in order to simulate high and low speeds of a 240V load, such as a pool pump.

For our demo, the third subsystem shown in Figure 3.3.1, i.e the software application, is satisfied by an Android application created using Processing. From this interface, the lamps connected to the device can be turned on/off. The interface for the application can be seen in the figure below. Making use of the current sensor in our device, the current pulled by loads is displayed on the application. The current sensor used in the demo does not have the capacity to properly monitor power since it does not take into account voltage and power factor. Thus, to obtain accurate current readings, it is necessary to calibrate the ADC values, which we will discuss in further detail in the subsequent paragraphs. In the future application, a sensitive metering chip, ADE7953 is used properly measure power. The ADE7953 is able to take into account power factor in order to determine accurate readings of power and voltage, which is then used to display graphical trends for the power usage and consumption.

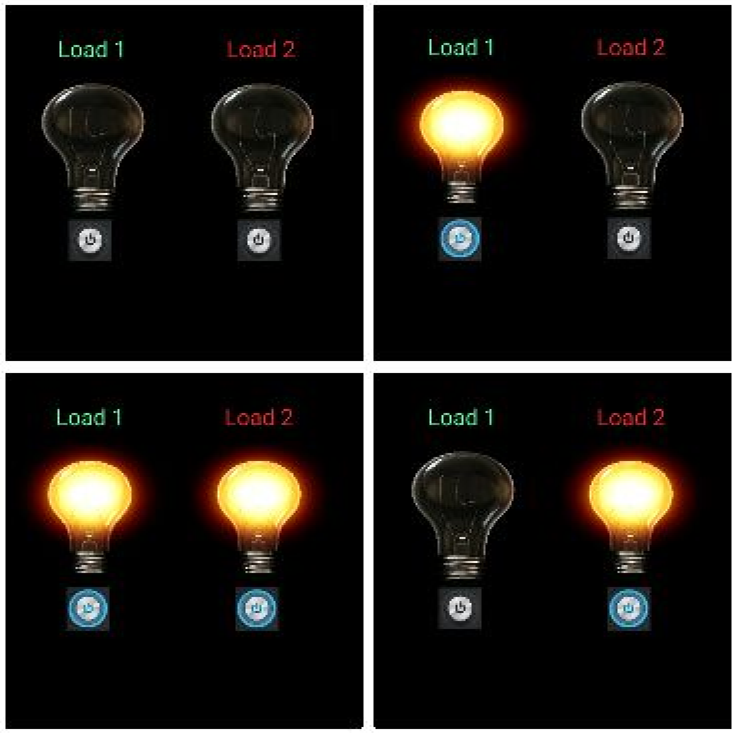


Figure 4.1.1: Interface of the software application. From this interface, the user can control the loads and also view the current pulled from the load(s). From top left, moving clockwise: Both loads turned off, load 1 turned on, load 2 turned on, both loads turned on.

We plotted the current measurement readings for each of the loads at the different wattages in order to find the calibrated linear regression. To obtain an accurate current calibration, we use different wattage light bulbs and measure the 10 bit ADC values, ranging from 0-1024. We plotted these values against the RMS current obtained with a HOBO data logger. The HOBO data logger used is manufactured by Onset and is UL certified, providing measurement accuracy within 0.5%, more information on this device can be found in Appendix B. In Figure 4.2.3, we see that the relation between the RMS current and ADC values are, for the most part, directly proportional to each other. We see an exception for low wattage bulbs, as seen in Figure 4.2.4.

To account for the nonlinear nature of the low wattage bulbs, we incorporate a polynomial function under a threshold in order to achieve accurate readings. The firmware uses the m (gain) values and b (offset) values displayed on each graph to calculate the current values based on the ADC values read in via the sensor. The data measured is shown below in Table 4.1.1 - Table 4.1.3. These results are restricted to the light bulb loads used within the demo itself. Other types of loads such as a heat gun would require more experimentation and calibration.

Table 4.1.1: ADC and A(RMS) data for Both Loads

Load 1	Load 2	Total Wattage	ADC	A(RMS)
0	0	0	704.2	0
15	0	15	704.82	0.122
0	25	25	705.96	0.21
15	25	40	708.26	0.332
75	0	75	715.68	0.604
75	15	90	719.74	0.726
100	0	100	722.7	0.815
100	15	115	727.46	0.937
150	0	150	739.52	1.308
150	15	165	743.12	1.43
100	75	175	742.68	1.419
150	25	175	745.88	1.518
150	100	250	760.04	2.123
150	75	225	755.42	1.912

Table 4.1.2: ADC and A(RMS) data for Load 1

Load1	ADC	A(RMS)
0	690	0
15	691.1	0.122
25	692.25	0.21
75	702.25	0.604
100	709.75	0.815
150	726.75	1.308

Table 4.1.3: ADC and A(RMS) data for Load 2

Load2	ADC	A(RMS)
0	689.56	0
15	690.25	0.122
25	691.25	0.21
75	701.25	0.604
100	708.75	0.815
150	725.75	1.308

4.2 Experimental Results (15 points)

The following data plots display the graphical distribution of the tables.

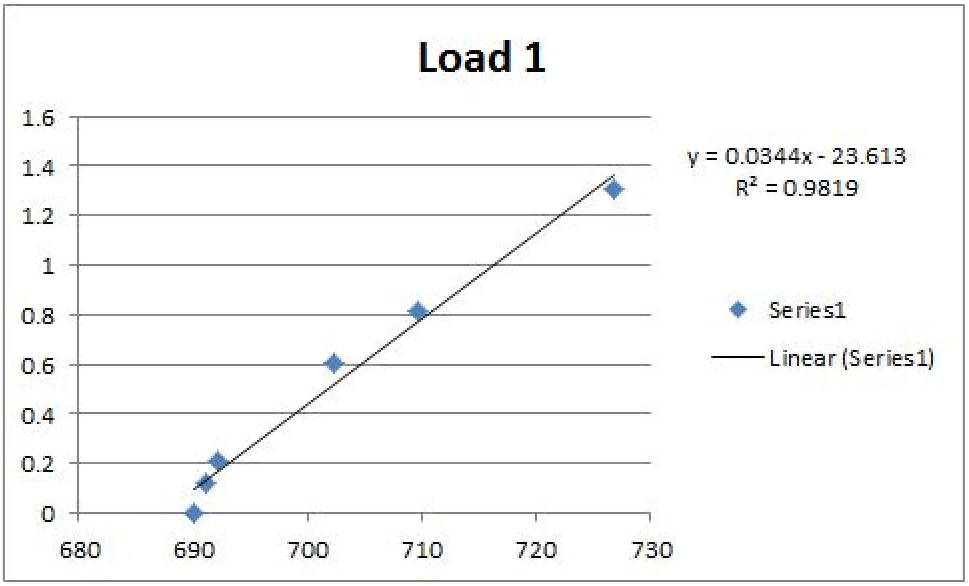


Figure 4.2.1: The ADC values v. current RMS for Load 1

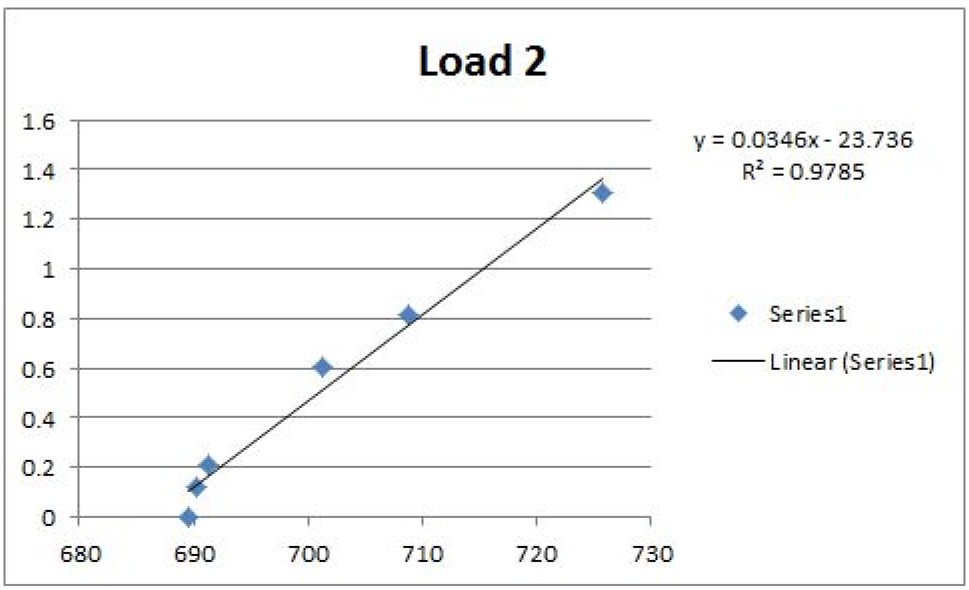


Figure 4.2.2: The ADC values v. current RMS for Load 2

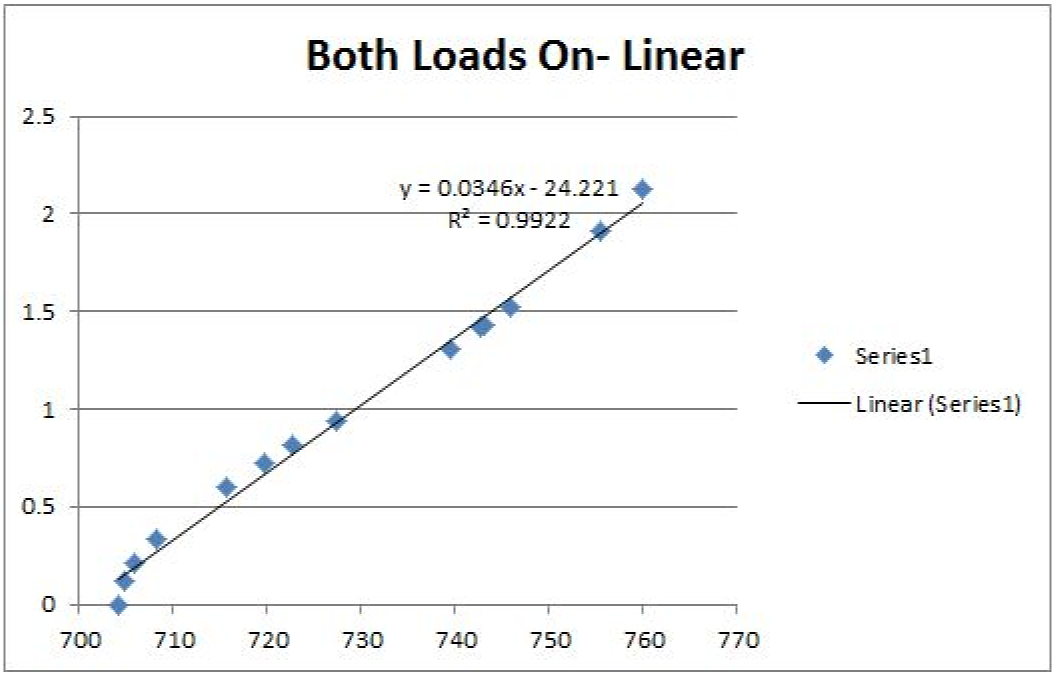


Figure 4.2.3: Using a linear regression, for all the light bulbs. From this linear regression, we obtain a gain of 0.0346 and an offset of -24.221

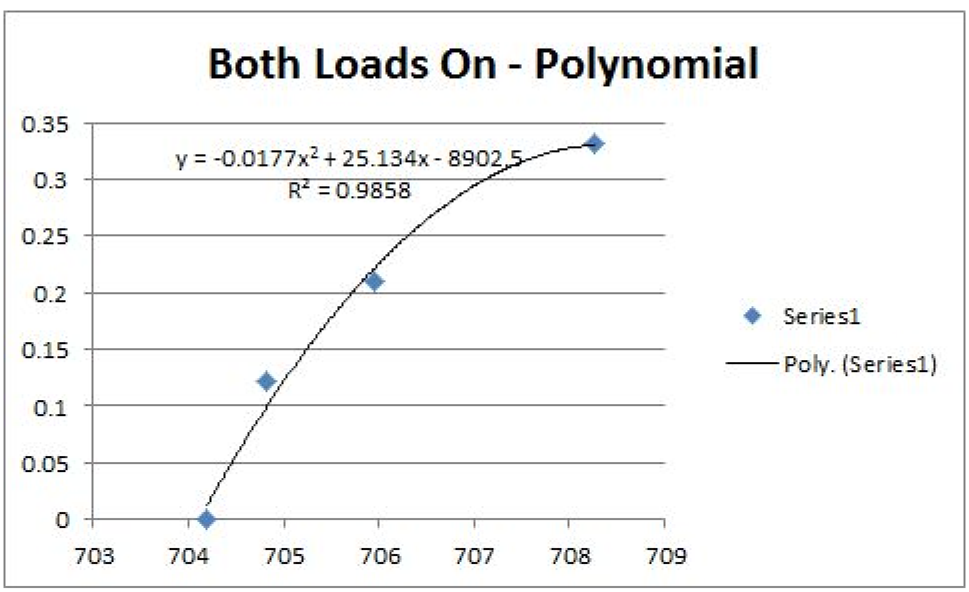


Figure 4.2.4: Notice in Figure 4.2.3 the lower values are nonlinear. Taking the data obtained from the values that were exhibiting nonlinear behavior (the four values on the left most side), we use a polynomial regression to obtain more accurate calibration values.

Based on the Table 4.1.1-4.1.3 and Figures 4.2.1-4.2.4, the current measurement is calibrated in a linear regression. Although most parts of the graphs displayed a linear regression, there is a nonlinear regression in the graphs once the ADC values are below a certain threshold. In order to provide further accurate readings, there needed to be further changes in the calibration for the nonlinear regression in the graphs. A threshold value was chosen to signify the point of change among the nonlinearity and linearity of the graph. Below a certain threshold, the graph displayed a predictable nonlinear characteristic. After testing and verifying the graphs among one another, the nonlinear characteristic is consistently a polynomial quadratic regression. Thus, above a certain threshold, the values are calibrated with a linear regression, and below a certain threshold, the values are calibrated with a polynomial regression. Because the current sensor in which the project is currently design is not able to produce extremely accurate measurements, the calibration is done to provide more accuracy to the measurements. In the future design, a more, sophisticated metering chip will be used to provide precise, accurate power and voltage measurements.

Chapter 5. Lessons Learned (15 points)

The main purpose of the project was to create a functional demonstration presentation to present the IoT metering load controller. The project demonstration is a proof-of-concept in which the hardware design is based on specific, sophisticated hardware parts such as a detailed PCB and an accurate, metering chip. Since the PCB and precise metering chip, ADE7953, were unable to be manufactured and delivered in time to be implemented in the design, it was important for us to choose materials similar to the parts as the project demo was made. It was necessary to make the overall design and architecture of the hardware and firmware to be dynamic to easily integrate the parts in future designs.

In order to meet deadlines, A significant amount of time was spent on setting up the Eclipse environment for the software implementation of the project in regards to the firmware on the Espressif8266 chip of the IoT edge device. The Eclipse environment was eventually set up with the correct configuration for the Espressif8266 chip, but additional newer Espressif8266 chips were bought to help with testing. Because of the newer design of the chip, the Eclipse environment, for some reason, was not able to be implemented on the new chip. Thus, in the end, the software implementation was programmed in Arduino C.

5.1 Project Management (7 points)

5.1.1 Team Organization (5 points)

The team consisted of two CSE majors and two EE majors, which provided a complementary team dynamic. Because the team members had experience and knowledge in software and hardware, the project is applicable to all members in their own respective fields. The dynamic skillsets allowed team members to help one another and learn from their own respective specialized fields to help progress the project even further. Through collaboration and teamwork, although each team member had their own respective assigned tasks, team members were also able to fundamentally all aspects of the project in order to effectively present the demonstration. Many of us did not have experience with the particular hardware architecture design with the Wifi chip or metering chip and software programming on MQTT or Espressif8266 chip, but we were all able to learn during the process and in the end benefit from learning this new skillsets.

In order to consolidate goals and expectations, early in the quarter, a design requirements and specifications documents was reviewed in regards to the hardware design and software application design. Once the designs were finalized, the group began to build the project and all production of the demo made was fundamentally based on the design requirements and specifications. To meet the deadline, any non-essential modifications on the requirements and specification were taken as notes and is to be used in the future implementation of this project.

Overall, the main point of the project was to provide a deliverable proof-of-concept demonstration. We were able to achieve this goal and worked extremely hard to meet our requirements and specifications given any circumstances. Through teamwork and dedication, we were able to create a solid demonstration proof-of-concept project.

5.1.2 Resources (2 points)

We were able to receive adequate funding and resources to finish the project demo. Through UROP funding, the parts of the project were bought locally from electronic stores such as Microcenter and Frys and online from Amazon, Digikey, and other online component sites. To meet for team collaboration as well as physically putting together the project, the CalPlug space was used as our working environment. Although the initial design of the PCB and metering chip were built and manufactured, errors were found on the boards, which needed to rectified in order to be correctly implemented into the project. As the PCB and metering chip needed to be reconstructed, the PCB and metering chip, ADE79753, are being manufactured and produced and will be implemented in the future designs of this project.

5.2 Technical Approach (5 points)

The initial design specifications were made by the project mentors from CalPlug and Smartenit. Based on the design specifications, we finalized the design requirements given by the Smartenit team. By solidifying the design specifications and requirements, the project design became stabilized and was set with goals and actionable items. The action items were split amongst the time with set deadlines. This project heavily relied on the integration of all of the parts to move in sync with one another. The main design of the project was to implement a triple-relay IoT metering load controller device, which is connected to the cloud and can be controlled remotely on a software application through WiFi. Smartenit had an initial design model based on a similar pre-existing device, which was based on Zigbee. The main distinction of our project is the implementation of WiFi and the MQTT protocol, which allows for a cloud-based application as well as encryption on the messages.

The WiFi module was first built to establish a WiFi connection with the Espressif8266 chip of the IoT edge device. Once the WiFi module was completed, the MQTT module was then layered in on the firmware to act as an MQTT client to communicate with the MQTT broker in the cloud. On the side, the Processing application, software user interface, was also made as another MQTT client to communicate with the MQTT broker. Once a connection could be established for each MQTT client, messages were relayed between the MQTT clients and MQTT broker to ensure solid communication of messages and commands. To further enhance the MQTT messages, a mode of encryption, AES-128 ECB, was added to encode and decode messages for a layer of protection to prevent intrusion via eavesdropping or unauthorized control making the system tolerant against false commands by discarding messages received that have been encrypted using the wrong encryption key. Finally, the firmware was ready to be tested and incorporated into the hardware design.

The hardware design used a simpler current sensor and circuit design to fully implement the project. Three relays were connected as switches to power on and off the loads. The firmware was later adjusted and tested to implement these switches to power the loads. The current sensor was added to the circuit in order to measure the ADC values. The firmware calibrated this values to provide more accurate readings given that the current sensors did not provide proficient, accurate values. The LEDs were added to signify the status of the WiFi and the status of the loads, in which firmware was added to simulate the states.

The software and firmware of the initial project design is fully implemented and functional. There were some problems with the hardware parts being manufactured and produced in time, which led to an alteration of the design. The project demonstration is a finalized proof-of-concept for the IoT metering load controller and is to be further developed by the Smartenit team with the PCB design and precise metering chip, ADE7953.

5.3 Concluding Remarks and Credits (3 points)

We were able to successfully complete a demo for our project, which serves as a starting point for the IoT-MLC that enables WiFi, MQTT with encryption, and a calibrated current sensor. Smartenit and CalPlug have plans to continue the efforts of our senior design project.

We would like to acknowledge and thank the following people and organizations for helping with the project:

- Professor G.P. Li and Michael Klopfer (academic mentors) for guidance and support throughout the entire project
- Smartenit team (corporate mentors) for guidance and support for the project requirements and specifications
- Calit2/CalPlug for the tools and working environment
- UROP for the funding for materials for the demonstration
- Microsemi for the tools and working environment
- Nicholas Farabee (an undergraduate researcher for CalPlug) for assisting with the development of the MQTT software in our project.
- ESP8266 community for the development and documentation for software libraries used such as WiFiManager, LED, Button, FSM, which served as the backbone for our firmware development
- The Processing communities for the development and documentation for libraries, which served as the backbone for our application development

References (5 points)

[1] "Annual Energy Outlook 2015." US Energy Information Administration, n.d. Web. <http://www.eia.gov/forecasts/aeo/>.

[2] "California ISO - Documents By Group." California ISO - Documents By Group. N.p., n.d. Web. 13 Dec. 2015.

[3] "California Renewable Energy Information." California Renewable Energy Statistics & Data. N.p., n.d. Web. 13 Dec. 2015.

[4] "California Renewable Energy Overview and Programs." California Renewable Energy Overview and Programs. N.p., n.d. Web. 13 Dec. 2015.

[5] "Electricity and the Environment." Electricity and the Environment. N.p., n.d. Web. 13 Dec. 2015.

[6] Fogel, Cathleen. "California's Zero Net Energy Policies and Initiatives." California's Zero Net Energy (n.d.): n. pag. Http://www.cpuc.ca.gov/. 18 Sept. 2013. Web.

[7] Haye, Sarah, Naomi Baum, and Garrett Henderson. "Real Prospects for Energy Efficiency in the United States." Energy Efficiency: Is the United States Improving? (2010): n. pag.American Council for an Energy-Efficient Economy. Web.

[8] "HOME AUTOMATION." Honeywell: Honeywell Home Automation. N.p., n.d. Web. 29 Nov. 2015.

[9] Mitchell, Cynthia. "Why California Needs to Think Differently About How It Supports Energy Efficiency." Greentech Media. N.p., 09 Jan. 2015. Web. 13 Dec. 2015.

[10] "Power Scorecard Homepage." Power Scorecard Homepage. N.p., n.d. Web. 13 Dec. 2015.

[11] "Putting Business First in the Internet of Things | Iot-Inc." The Business of the Internet of Things. N.p., 09 Sept. 2014. Web. 13 Dec. 2015.

[12] "Samsung SmartThings Home Monitoring Kit." Samsung Electronics America. N.p., n.d. Web. 29 Nov. 2015.

[13] "Smart Home Market Analysis By Applications (Security, Lighting, Entertainment, Energy Management, And HVAC) And Segment Forecasts To 2020." Smart Home Market Size And Share. N.p., n.d. Web. 13 Dec. 2015.

[14] "Smarthome Home Automation Superstore." Smarthome Home Automation Superstore. N.p., n.d. Web. 13 Dec. 2015.

[15] "Supported Technologies." OpenHAB. N.p., n.d. Web. 29 Nov. 2015.

[16] "Why Isn't There a Standard Voltage around the World?" World Standard. World Standard, n.d. Web.

[17] "ZigBee: The Open, Global Wireless Standard for Connecting Everyday Devices." The ZigBee Alliance. N.p., n.d. Web. 8 Nov. 2015.

[18] Bradley, Joseph, Joel Barbier, and Doug Handler. "Embracing the Internet of Everything To Capture Your Share of $14.4 Trillion." (n.d.): n. pag. Web.

Appendix A

The source code can be located through the following link.

In the folder for the source code the following files can be found:

1. Application_mqtt_LED_GUI_encryption.zip
    a. mqtt_LED_GUI_encrypition.pde
    b. Images used in application
2. Firmware_mqtt_LED_GUI_encryption.zip
    a. mqtt_LED_GUI_encrypition.ino
3. Arduino Libraries
    a. AES.zip
    b. Button Library
    c. FSM Library
    d. LED Library
    e. WiFiManager Library
    f. Arduino-base64-noprogmem.zip
    g. pubsubclient

The Arduino IDE v1.6.7 with the ESP8266 board (from the Arduino IDE board manager) was used to compile and flash the firmware. The firmware was flashed to to a ESP8266 12E module.

Processing 2.2.1 was used for development of the application software. This application can be run on processing via a computer. Furthermore, the processing can be used as an application on mediums such as tablets by using the processing application and wondershare's mobilego to allow the driver's to be recognized.

A.1 Listing of mqtt_LED_wifi_encryption.ino

This file is the firmware file compiled and flashed onto the ESP826612E via the Arduino IDE.

The firmware relies on the following libraries:

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

#include <ESP8266WebServer.h>
#include <DNSServer.h>
#include <WiFiManager.h>

//The following include is for led FSMs
#include <FiniteStateMachine.h>
#include <LED.h>

//The following include is for button input
#include <Button.h>

#include <Base64.h>  //the arduinobase64 library had to be modified to remove the need for the avr/progmem library
#include <AES.h>
#include <AES_config.h>
```

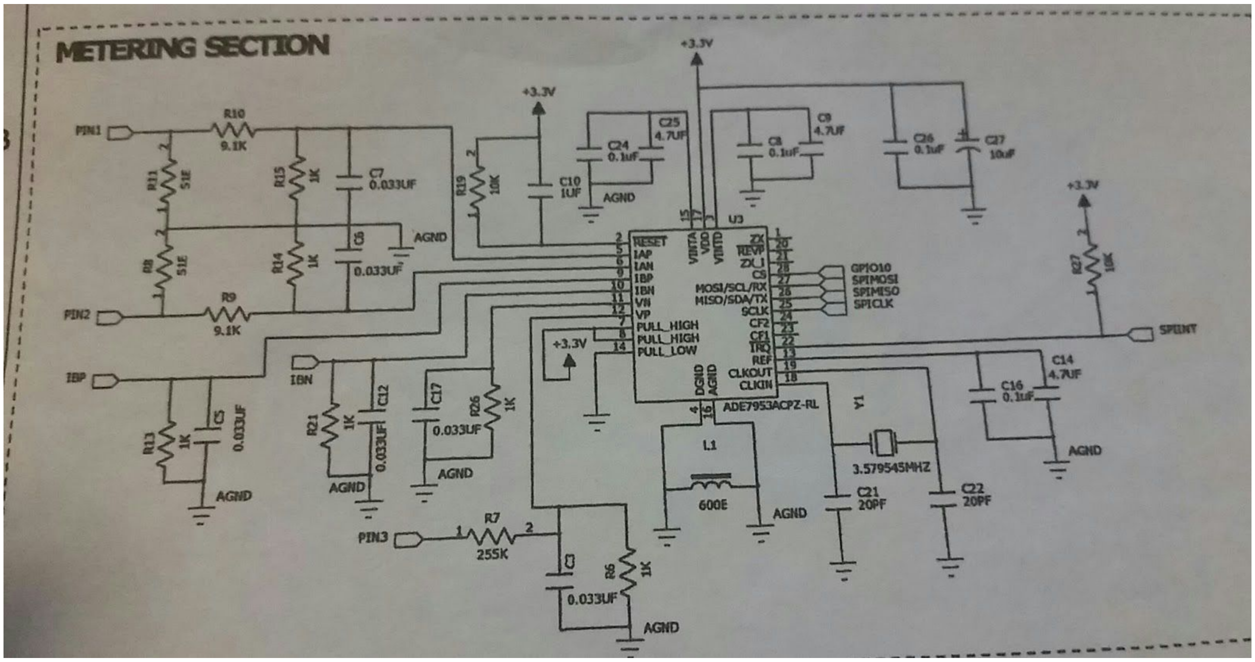
Some libraries such as AES,LED, Button, and FiniteStateMachine have been included as folder or zip uploads with the source code. Other libraries such as ESP8266WiFi may require import through the Arduino library manager which can be done via Sketch->Include Library and proceed to look for the library under the Manage Libraries or Contributed Libraries options.

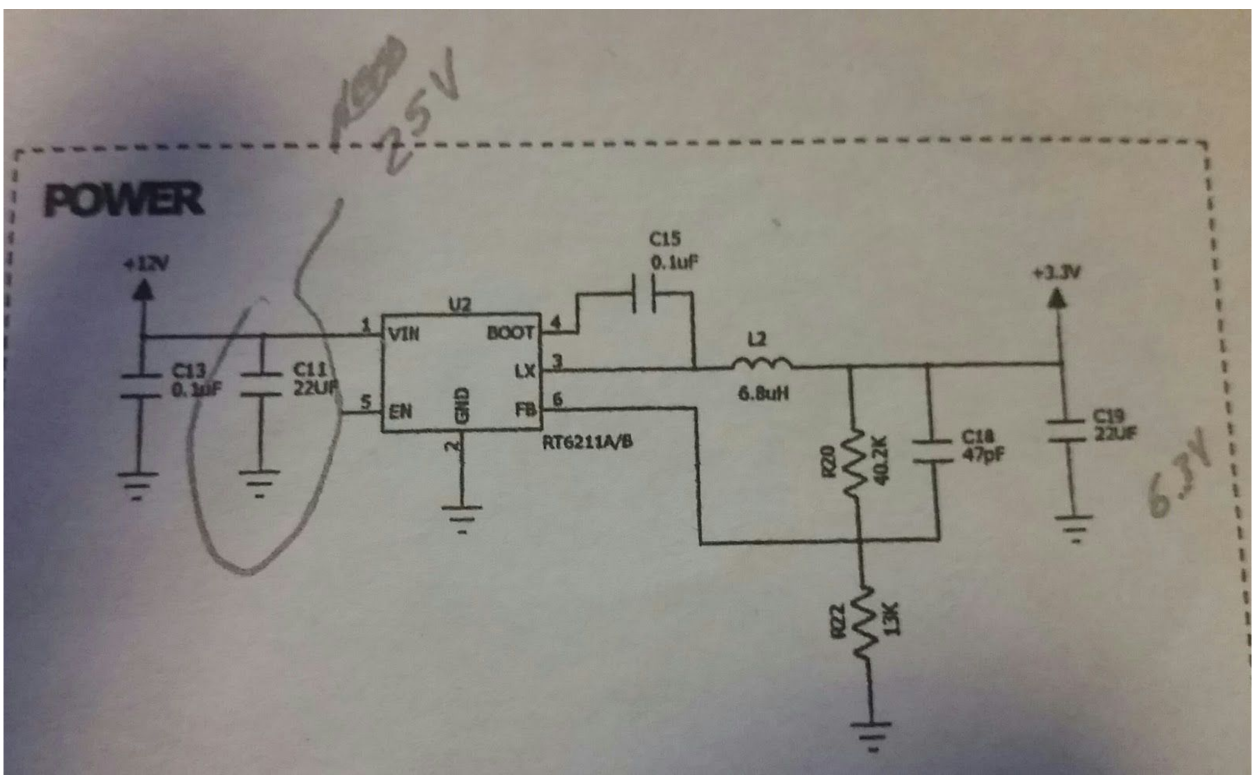
A.2 Listing of mqtt_LED_wifi_encryption.pde

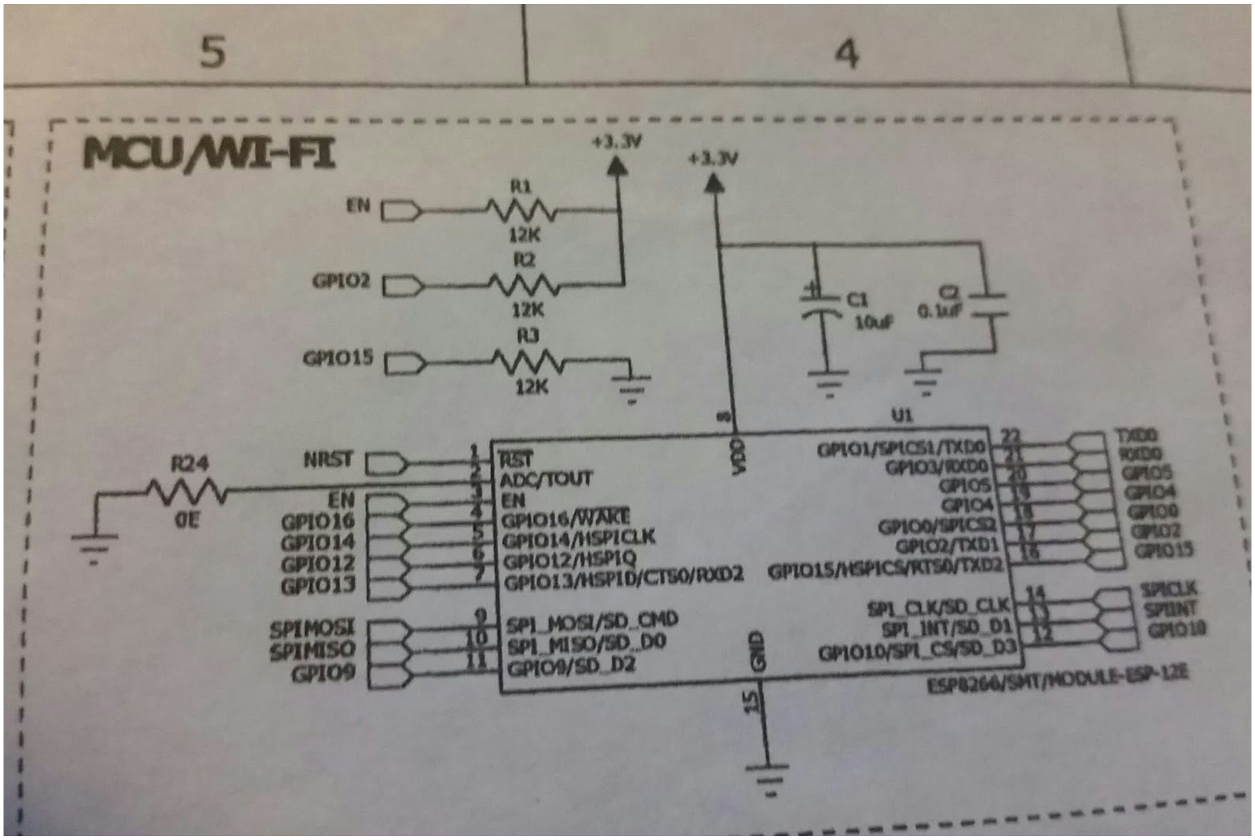
The application code consists of a pde file named mqtt_LED_wifi_encryption. It requires images located within the Application)mqtt_LED_wifi_encryption.zip file to run.

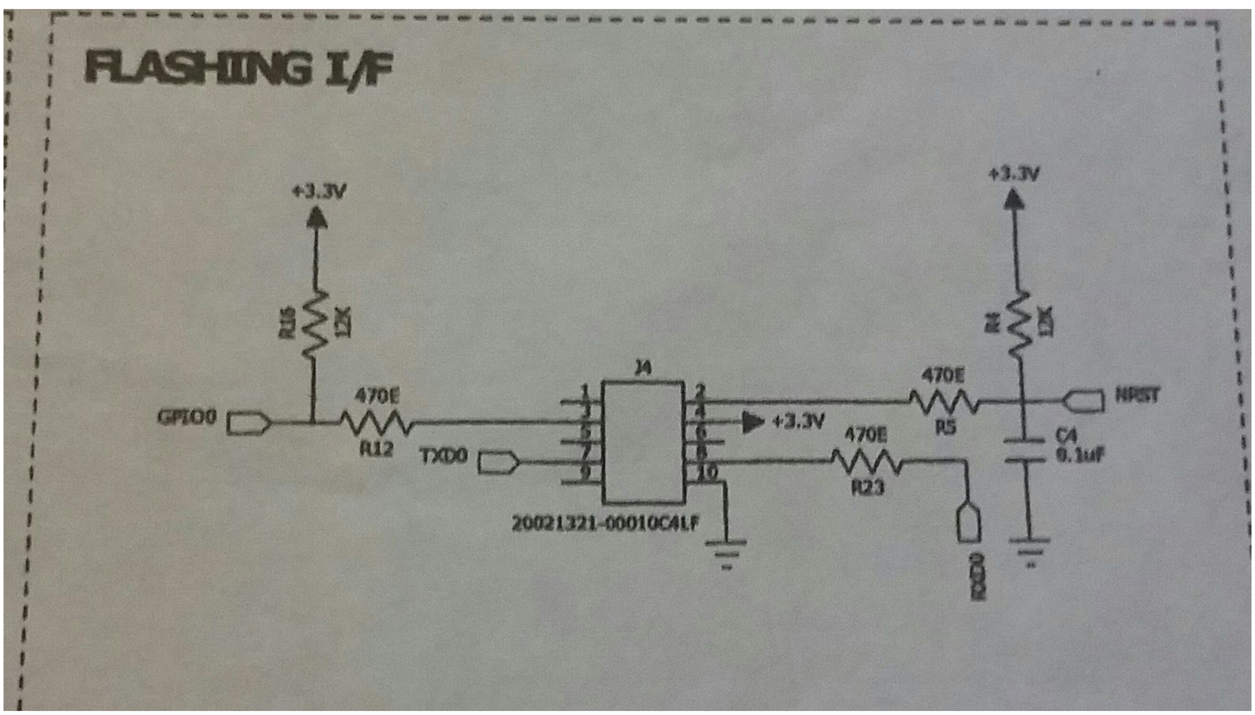
Appendix B

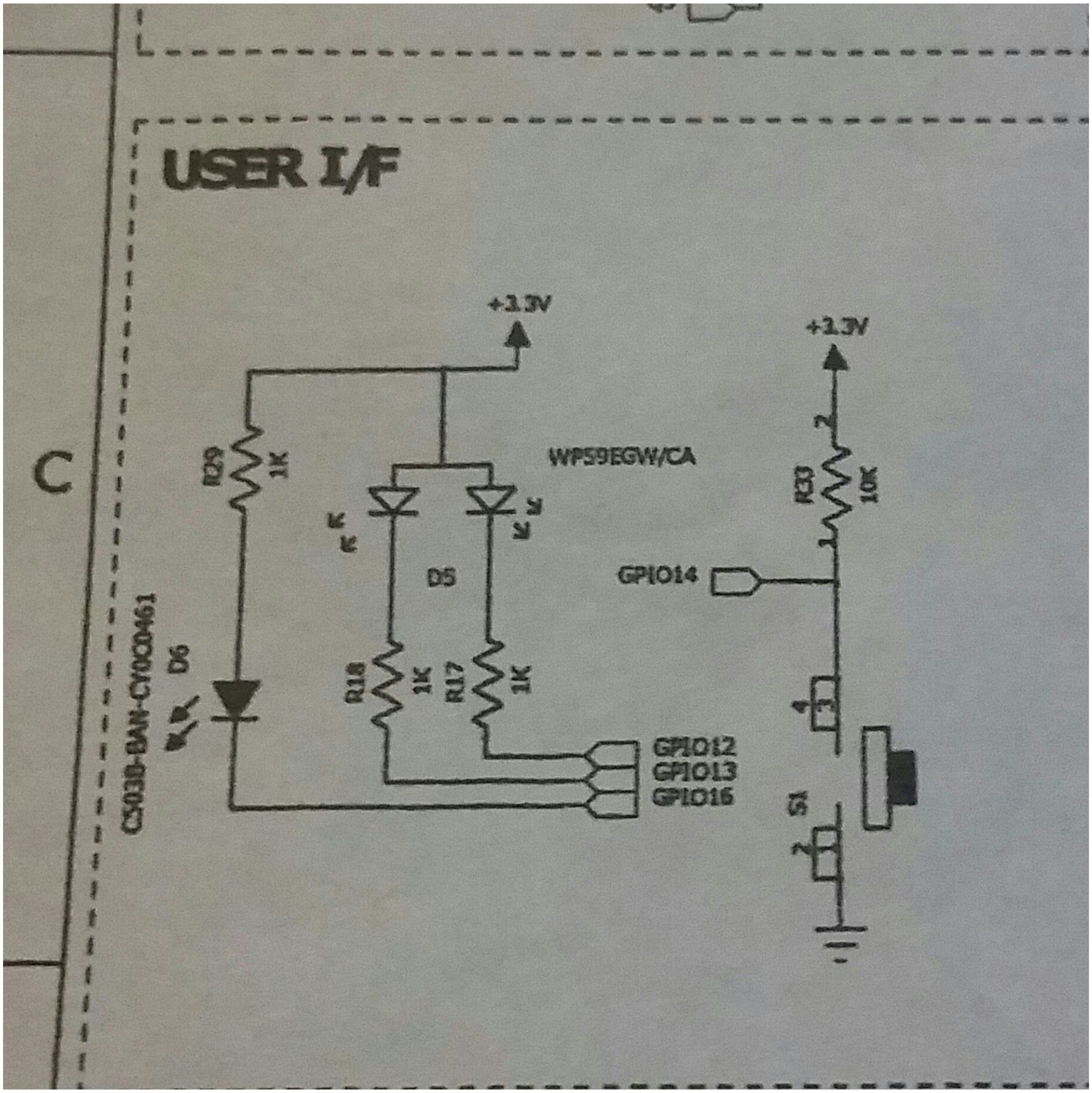
B.1 Schematics

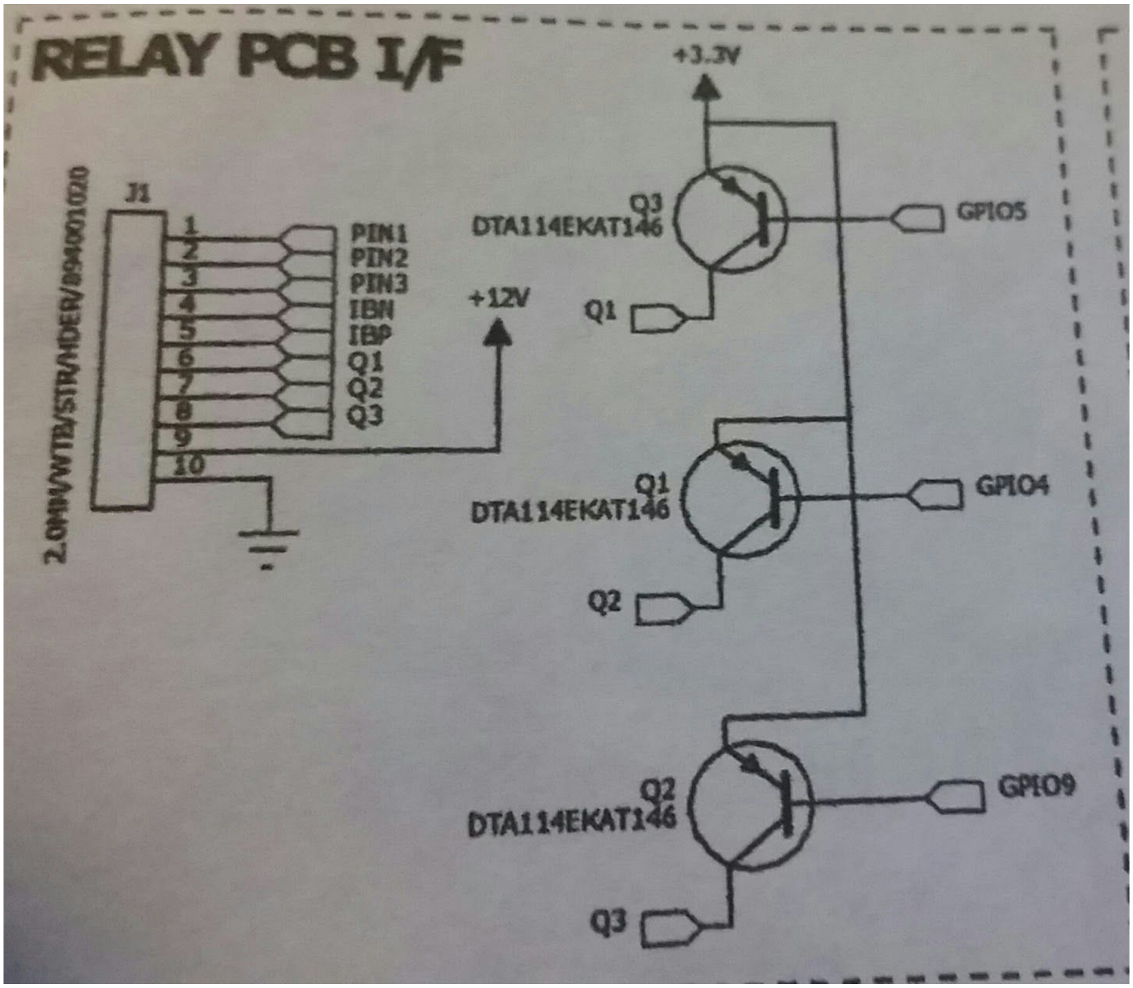





5
4
MCU/WI-FI
+3.3V
+3.3V
EN
R1
12K
GPIO2
R2
12K
GPIO15
R3
12K
C1
10uF
C2
0.1uF
U1
R24
0E
NRST
EN
GPIO16
GPIO14
GPIO12
GPIO13
SPIMOSI
SPIMISO
GPIO9
RST
ADC/TOUT
EN
GPIO16/WAKE
GPIO14/HSPICLK
GPIO12/HSPIQ
GPIO13/HSPID/CTS0/RXD2
SPI_MOSI/SD_CMD
SPI_MISO/SD_D0
GPIO9/SD_D2
VDD
GND
GPIO1/SPICS1/TXD0
GPIO3/RXD0
GPIO5
GPIO4
GPIO0/SPICS2
GPIO2/TXD1
GPIO15/HSPICS/RTS0/TXD2
SPI_CLK/SD_CLK
SPI_INT/SD_D1
GPIO10/SPI_CS/SD_D3
TXD0
RXD0
GPIO5
GPIO4
GPIO0
GPIO2
GPIO15
SPICLK
SPIINT
GPIO10
ESP8266/SMT/MODULE-ESP-12E


FLASHING I/F
+3.3V
R16
12K
GPIO0
470E
R12
TXD0
J4
20021321-00010C4LF
+3.3V
470E
R23
RXD0
+3.3V
R4
12K
470E
R5
NRST
C4
0.1uF


USER I/F
C
+3.3V
WP59EGW/CA
R29
1K
D5
R18
1K
R17
1K
C503B-BAN-CY0C0461
D6
GPIO12
GPIO13
GPIO16
GPIO14
+3.3V
R33
10K
S1



B.2 Layout

B.3 Parts used for Proof-of-Concept demonstration

Onset's HOBO Plug Load Data Logger -
http://www.onsetcomp.com/products/data-loggers/ux120-018

Appendix C

This is the link to the full length demo video for the IoTMLC. First the demo shows the physical frontal view of the controller with labelled parts. There is a quick view of the internal circuitry that composes the controller, before showing the connected loads and their wattage. The demo then proceeds to show the WiFi Status LED's response to entering access point mode before going through a walk through of the open access point connection through a computer. The walkthrough shows the access point broadcasting, a user connecting through predefined credentials, as well as the site through which the WiFi configuration settings are done. The demo goes on to go through each option (Configure Wifi with Scan, Configure Wifi No Scan, Info, and Reset). The video then demonstrates the Status LED states (Load 1 on only, Load 2 on only, Both Loads on, and Both Loads off) for each load and how the controller changes those states based on physical button push. Next the video shows the actual loads in each of those states. In addition, the demo goes through the application in the different states,the current measurements, and controls through the buttons on the application. Finally, the demo exhibits the complete setup of the application, the loads and the controller with control through the push button and the application.