Ryerson University
Department of Electrical, Computer & Biomedical Engineering
Faculty of Engineering & Architectural Science

XF 10: BATTERY FREE IOT SYSTEM FOR HOME AUTOMATION

by

Ryhan Adnan, Tenzin Dhargye, Hansel Kalathil, and Samayla Noor

Computer/Electrical Engineering Capstone Design Project

Ryerson University, 2023

Acknowledgements

First and foremost we have to thank our research supervisor (FLC), Xavier Fernando. Without their guidance and assistance, this project would not have been accomplished.

Completing this report would not have been possible without the efforts shown by the group members in this project. We would like to thank Ryhan Adnan, Tenzin Dhargye, Hansel Kalathil and Samayla Noor for their unwavering effort and their persistence to complete this project despite what hardships were going on during this semester.

Finally, we would like to thank our family and friends for their unwavering support. It would be an understatement to say that this semester was very stressful, but without your unconditional love and support, this project would not have been possible. Thank you.

Certification of Authorship

“ I/We certify that all author(s) of this document are submitting work that is our own and created with the best of our own knowledge. Any and all sources are acknowledged and disclosed in the document. Any sources used in the document such as ideas, data, and/or paraphrased paper are credited in the references. All sources are credited with accepted professional standards.”

Authors :

Name	Signature	Email
Ryhan Adnan	R.A	radnan@torontomu.ca
Tenzin Dhargye	T.D	tdhargye@torontomu.ca
Hansel Kalathil	H.K	hansel.kalathil@torontomu.ca
Samayla Noor	S.N	samayla.noor@ryerson.ca

Table of Contents

Abstract

Over the last decade home automation has gained great popularity as it increases the comfort and quality of life. The growth of the automation industry has led to greater advancements in human life. IoT (Internet of things) is an emerging new technology that describes the physical network of “things”, which are embedded with sensors, software and other technology for the purpose of exchanging data with devices over the internet. IoT has become one of the most important technologies in the 21st century. It has made controlling everyday objects such as baby monitors, thermostats, and lights highly efficient. Smart phones have become an integral part of our daily lives today, we are reliant on them for many of our tasks, from answering the phone to sending emails, and much more. Batteries that have to be replaced every now and then have a large environmental impact, therefore this project looks to design a system without the use of batteries to power the sensors involved. Since the sensors require power to constantly flow and there is no better way to do so without using some sort of storage system, we decided to implement supercapacitors. Supercapacitors have a much lower environmental impact as they have an infinite lifespan meaning they will not have to be replaced like their battery counterparts. To charge the supercapacitors we’ve decided to use solar energy so that in this way the whole system could be as ‘green’ as possible. Specifically solar energy was used to power up a PIR motion sensor, a BH1750 light sensor and a BME280 heat sensor. It was also able to control a DC motor to open, and close a garage door on demand. An ESP32 is used as the microcontroller to have wireless communication between the sensors and the device. Finally, all of this was monitored in real-time using ESP Home. All these sensors will be transmitting data through wifi to the software Home Assistant. To operate the software, a dedicated system needs to be implemented to run the software. As such we would require the use of a Raspberry Pi to act as the microcontroller system that can connect to a multi- network devices. The Home Assistant can allow the user to see real time sensor output and allow the user to create multiple automation that test either the functionality or performance of the component. The software can then connect to a smart device which can be controlled automatically for either lighting or garage door opening.

Introduction & Background

Home automation is a fairly new concept that has been gaining popularity in recent times. It consists of programming technologies through a central hub, which can route through different networks and give access to your home whether you are present or not. One of the benefits to home automation is being energy efficient and saving money. This can be done by setting up the hub to turn the system on or off automatically, as such a device like a thermostat can be automated to turn on energy-saving mode when no one is present within a certain area. Home automation also allows for the user to set certain parameters for the system to follow, allowing for convenience and home security. The lights can be programmed to turn off after a certain time when no presence is detected or a camera can relay a live feed when there is movement in or outside the home if the home owners are not home. Since a wide range of smart devices can be connected via software for home automation, the user can create an endless amount of home automation.

Now that we understand that there are many benefits associated with home automation, we should have greater knowledge of how it works. With IoT physical things can be connected through many different networks to other devices. It mainly encompasses smart devices with embedded processors, sensors, and communication to send and receive data. The devices connected to the hub share the data collected and then it is analyzed. This is mainly done with sensors. The sensors collect information from different environments that then enable devices to receive the information and act accordingly. Sensors are great, as there is no human intervention needed. It helps in collecting, sending, as well as acting on that information.

Although the idea of IoT has existed for quite a while, there have been many recent advancements in technology that have allowed it to become practical. The idea of adding sensors to simple objects was discussed throughout the 1980's but progress was very slow. The reason behind this was that the technology was just not ready yet. Chips were too big, bulky and communication between objects could not be established effectively. Soon RFID tags came into light and solved a part of the problem as they were low power chips able to connect wirelessly, along with the increasing availability of internet and wireless networking. The adoption of IPv6 which is the latest version of the Internet Protocol, along with some other factors really helped to scale IoT. Today there is access to low-cost, and low-power sensor technology. A host of network protocols allowing for sensors to easily connect to the cloud. Conversational AI has taken IoT to a higher level. Advancement in neural networks have brought natural language processing (NLP) to IoT devices such as Siri, and Alexa, making them appealing, affordable, and viable for home use.

Objectives

Although many advancements have been made in home automation, there is yet to be an IoT system specifically designed for home automation that uses energy harvesting processes. There are many home automation systems available in the market, so the goal will be to create a system that is competitive but can also harvest its own energy. Modern home automation systems also have many benefits such as safety features, convenience, comfort and energy saving technologies. All of these benefits will be taken into consideration when designing the IoT system. The goal will be to implement various different sensors and use them in unison to create a seamless and secure home automation system. All of these sensors will be used over Wifi as it is very simple and it exists in every home. Essentially, the objective of this design project is to create a battery free IoT system that successfully completes automotive tasks. These tasks are as followed:

- Controlling lighting using a PIR (Passive InfraRed) motion sensor and a BH1750
- Controlling and observing the humidity levels in the room using BME280
- Operating the Garage door that uses NFC tags to recognize whether or not the correct phone is scanning the tag or not
- Using the Home Assistant layout to create a user friendly hub/UI that can be used to control and observe the sensors

To summarize, this project will use the already existing simplicity of Home Automation and use it to create a battery free and easy to use system. If energy efficiency is taken into account with the ergonomic benefits of an IoT system, then the possibilities of home automation are endless.

Theory and Design

Garage Door:

In order to incorporate the garage door component with the rest of the project, a block diagram that consists of the components required was created. The plan that was created in the fall semester (ELE 70A) was seemingly very good but it was not compatible with the rest of the project. For that reason it was decided to be replaced by a much better and simpler design. The changes that were implemented have been presented below.

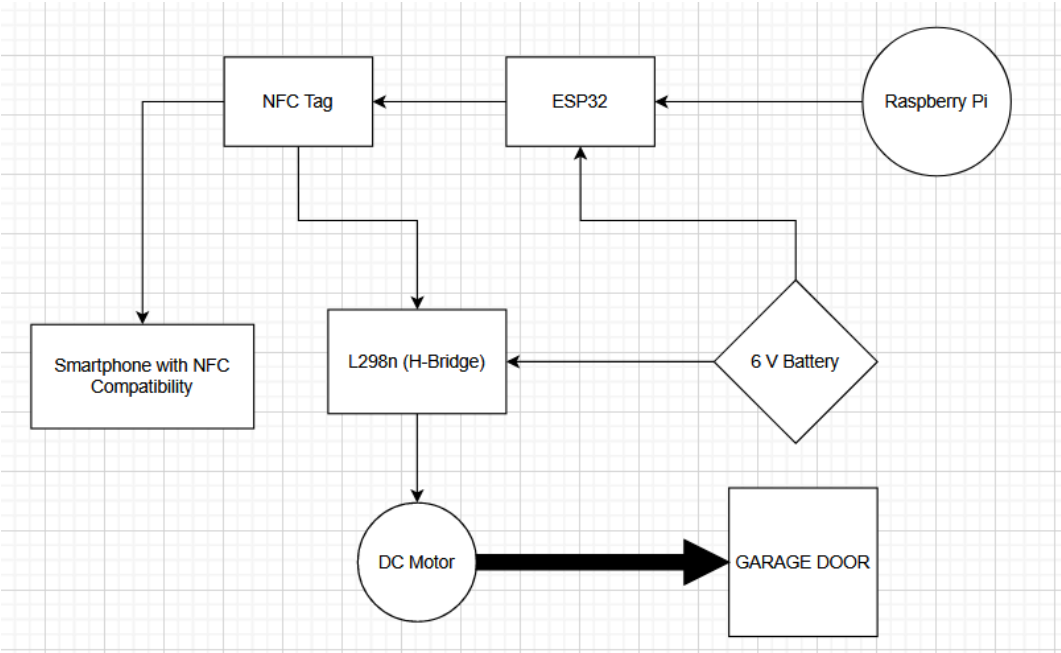

Figure 3.1: Block Diagram for the Garage Door Configuration

As seen from the diagram above, the main Raspberry Pi will now be connected to the ESP 32, which is compatible with Home Assistant. To briefly summarize, the ESP 32 will be used in conjunction with the Raspberry Pi to allow the L298n (H-Bridge) to communicate with the Home Assistant software. This will then allow us to create automations in the Home Assistant interface which will then let the user open and close the garage door. The L298n component will be powered by a 6 V battery and it will be used to control the direction of the motor. The NFC tags will be used to activate the H-Bridge and it will determine whether or not the motor will spin forward (open) or backwards (close). This will be dependent on which tag is scanned. There will be one tag outside the garage and one tag on the inside.

NFC Tags

Instead of having an NFC Tag reader, the user's smartphone can be used to read the tag and open the garage door. Only registered users can scan the tag and have it work. This means that it is very secure. This is because even if the user has somehow connected to the network, since the phone is not registered by the system, the garage door will not open. The range for the NFC Tag is approximately 4 inches. The NFC tags were integrated into the system by creating code in the Home Assistant software using the YAML language. The system will have two NFC tags, one inside the garage and one outside the garage. The NFC tag which is outside the garage, will open the garage door when it is scanned by the correct smartphone which has been registered to the home network. This NFC tag inside the garage will close the garage door when it is scanned. Since the user's smartphone will be combined with the Home Assistant software, there is no connection required and that saves even more energy as the NFC tags do not need to be charged or powered externally.

L298n Motor Driver (H-Bridge)

The L298n Motor Driver will be connected to the ESP 32 using its GPIO pins. This will be used to control the speed of the Motor using PWM (pulse width modulation). This motor driver will be powered by a 6 V battery as shown from the diagram above. The reason this component was chosen is because it will allow the motor to rotate forwards and backwards. This means that it will be possible to open and close the garage door. Along with the forwards and backwards rotation, the speed of the motors can also be controlled. Another reason why this component was chosen is because it allows the option of adding two DC motors. This means that if one is not enough, another one can be added.

DC Motor

Since the L298n Motor Driver will be powered by a 6 V battery, a DC motor that has been rated for 6V to 12 V was selected to be used. This motor will then be used to open and close the garage door. The reason this motor was chosen is because it is reliable and it is compatible with the L298n component.

Energy Harvesting:

In order to power the sensor devices without using batteries the simplest solution is to use solar cells. Since solar cells are small and cannot supply much power as well as being very dependent on light, a mediator between the solar cell and sensory device must be implemented. In this case a lithium-ion supercapacitor. Since we are now using a supercapacitor, the energy delivered to the load, in this case the sensors can be described by the formula

(1) $E = \frac{1}{2}CV^2$

where E is the energy stored in the capacitor, C is the capacitance and V is the voltage.

As seen below a very simple circuit can be found in FIGURE 3.2. This circuit will not work as the solar cell must be at a higher voltage in order to charge the capacitor, therefore we must implement a boost converter between the solar cells and the supercapacitor, an example of this is shown in FIGURE 3.3. Finally, we require another buck/boost converter in order to regulate the output of the capacitor to the sensors. To perform both boost and buck/boosting tasks, we will implement a chip called the AEMLIC which will keep a regulated voltage of 3.3V as long as the supercapacitor is kept above 2.5V. Once below 2.5V the output voltage will shut off and the circuit will only perform charging the supercapacitor. Therefore, we must modify equation (1) in order to account for the shutoff.

(2) $E = \frac{1}{2}C\left(V_i^{\,2} - V_f^{\,2}\right)$

Where V_i is the initial voltage and V_f is the cutoff voltage

Finally we can determine how long the circuit will operate for using the equation below

(3) $t = C(V_i - V_f) \times \frac{1}{I_{load}}$

where t is in seconds, I_{load} is the operational current of the sensors

Looking at FIGURE 3.4 we can see a fully realized diagram which will be used in the final system.

Solar Cell

For the solar cell we will use LL200 - 4.8-37 Solar Cell, the datasheet is provided in Figure 3.5 in the appendix. This solar cell is perfect for our applications as it has a maximum operational voltage of 4.2V

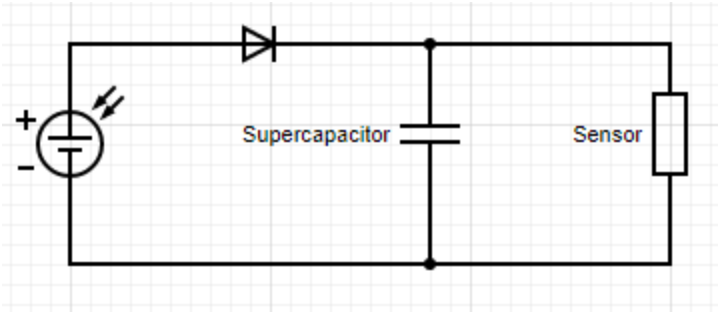

Figure 3.2: Simple Solar Harvesting Circuit.

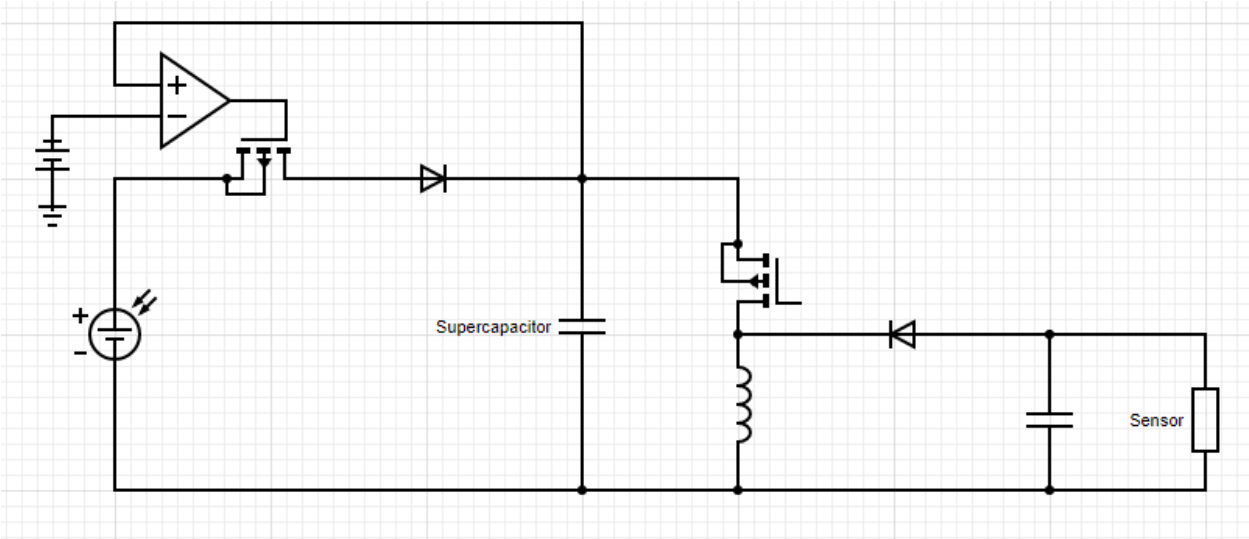

Figure 3.3: Updated Solar Harvesting Circuit.

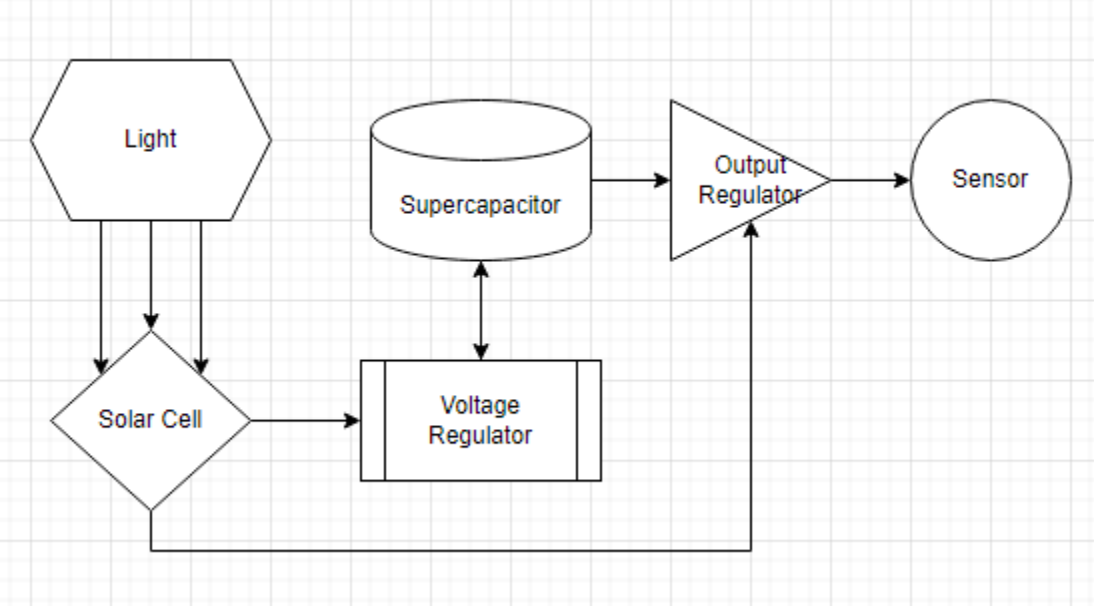

Figure 3.4: Solar Circuit Diagram

Supercapacitor

For the supercapacitor we will be using Vinatechs Lithium-ion supercapacitor which is rated at 250F, 3.8V. We decided to use a device with such high capacitance because the system is designed to operate overnight and therefore must have a high capacitance to be able to discharge overnight when the solar cells are not able to keep up with the demand. For the purpose of the demonstration, we will also use a 30F supercapacitor to more easily show the charging and discharging rates of the system.

Sensors:

PIR Motion Sensor

This sensor will receive infrared waves when someone is in the sensing range. Infrared waves are emitted by the human body which cause lights to turn on automatically. As long as someone is present in the sensing area the sensor switch will be on, and after a person leaves after a certain delay it will turn off which can be very energy-saving. There are many advantages to using a PIR motion sensor. The most obvious one being it is safe and energy saving. It is different from voice control as it does not require sound, and as it can be turned on by the mere presence of a person it avoids the situation of invalid power. There is also no problem with radiation for this sensor.

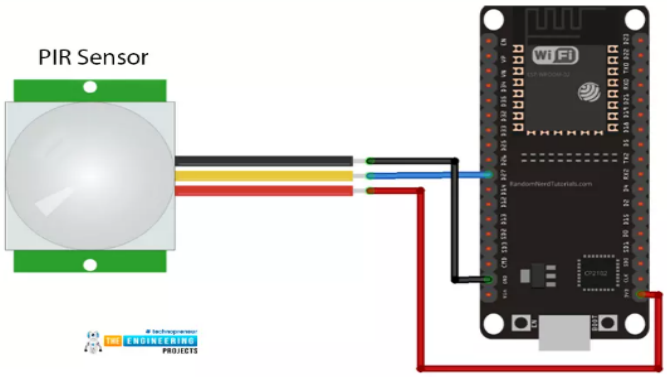

Figure 3.6 : PIR connection with ESP32. [10]

BME280

This is an integrated environment sensor by Bosch. It can measure humidity, pressure and temperature depending on the version. This is a very good indoor environmental sensor, and it supports both SPI (serial peripheral interface) and I2C(inter-integrated circuit) communication. The I2C protocol uses two lines to receive and send data. This includes a serial clock pin (SCL) and a serial data pin (SDA). I2C is popular and widely used due to the fact that it is so easily implemented in many electric designs that require communication between a master and multiple slave devices.

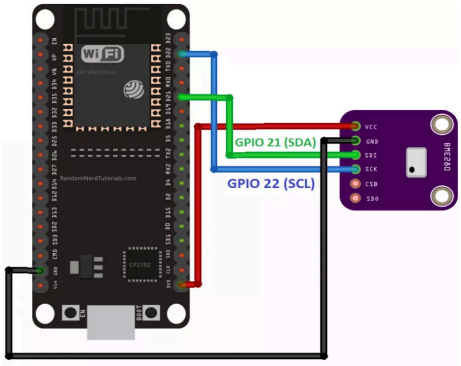

Figure 3.7: BME280 sensor connection with ESP32. [7]

BH1750

This is a 16-bit ambient light sensor which also has I2C communication and detects the amount of light in lux. Sensing the amount of light indoors can help to decide the time of day and whether lights need to be turned on or off. This sensor can be connected on the same bus as the BME280 sensor making the connection quite simple.

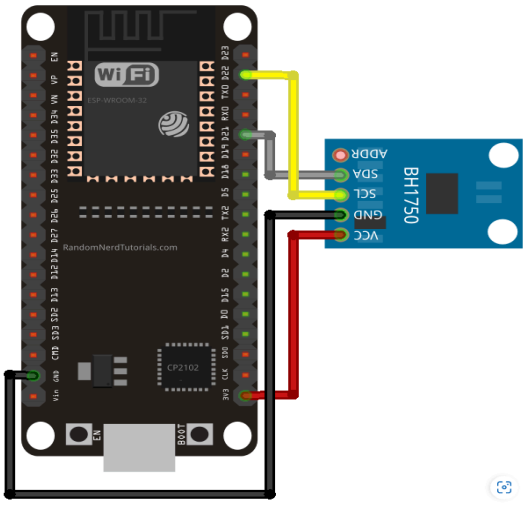

Figure 3.4: BH1750 sensor connection with ESP32. [5]

Software Software and Integration:

The software integration we chose was Home assistant. The system was picked due to the nature of the system being an open software which allows for the community to add additional add-ons that help improve the existing integration. The Home Assistant acts as the home controller making use of the Raspberry Pi to be the software's own dedicated operating system. When connecting the software with the sensors, the Esp32 is required to extract the data the sensors will be tracking. The Esp32 is a microcontroller that provides the user with compatible

sensors within its library, this dashboard is called ESPHome. The ESPHome can be installed into your home assistant repository, and the code can be obtained from the website esphome.io.

```
# Example configuration entry
sensor:
  - platform: bh1750
    name: "BH1750 Illuminance"
    address: 0x23
    update_interval: 60s
```

Figure 3.5: BH1750 Sensor Yaml Configuration [13]

In figure 3.5 we can see a sample of the code for the 16-bit ambient light sensor, this line of code helps operate the sensor and set certain update intervals to obtain data from the Esp32. In the Home assistant software, we can configure the yaml file for the esp32 and add all necessary codes for the different sensors. For the purposes of this project, we make sure of two sensors. A regular sensor which collects numeric data within a set update interval and a binary sensor which will obtain data that only need. The sensors after this point only require power in and to be connected to pins to start transmitting data through wifi or bluetooth. Later on we can integrate any devices using the same network to create automation with the sensors. The automation can be operated with triggers such as time or motion, this can in turn check a script that is written in the configuration, which in turn will trigger an action. The action can be many different activities but will typically operate a device to do a task.

Total Assembly

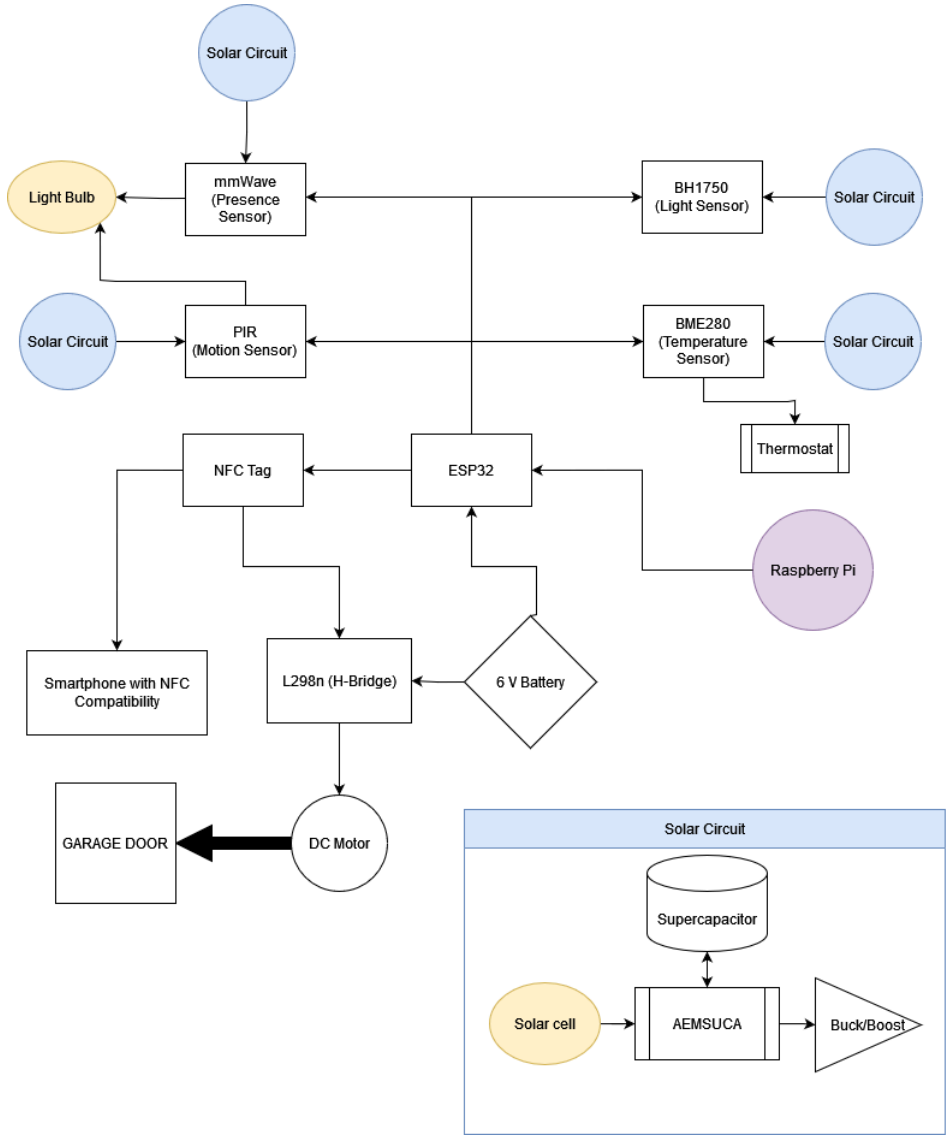

Figure 3.6: Block Diagram for the IoT System

As seen in **Figure 3.6** above, the main block diagram for the whole system can be seen. A Raspberry Pi 4 will be the dedicated system to operate the software called Home Assistant that will be used to control the system. The ESP32 is the central unit microcontroller that connects all the components to one another. As stated earlier, the main goal for this project is to ensure that the needs for home automation are met. This means that there must be a functioning temperature sensor (BME 280). There must be a functioning light sensor (BH1750). Also, in conjunction with one another, there must be a presence sensor (mmWave) and a motion sensor (PIR). Along

with this, there must be an automated garage door opening system that recognizes whether the correct vehicle is entering the garage or not. This will be done with the help of NFC tags.

These tasks/functions will all be completed using multiple different sensors. What makes this project special, is that the sensors will be powered using self sustaining energy harvesting devices. They will be powered using solar panels in connection with capacitors. The solar panels will produce power depending on the source of light, which will be sent to the AEMLIC which will regulate the voltage to be used in charging up the supercapacitor. While the supercapacitor is above 2.5V, the output of the buck/boost output voltage regulator will be enabled at 3.3V. To connect the sensors using the solar circuit we must simply connect the input pins of the sensor to the positive terminal of the buck/boost voltage regulator of the solar circuit. Next we will connect the negative terminal of the solar circuit's output to the ground of the sensor. The ground of the sensors must also be connected to the ground of the ESP32 as all the components must share a common ground.

As seen in the diagram above, there is a light sensor (BH1750), a mmWave sensor, a PIR sensor, and a temperature sensor (BME 280). The light sensor, mmWave sensor, PIR sensor, and heat sensor will all be used to control the lighting and heating settings in the home automation system. The goal will be to increase and decrease both the lighting and heating in the home depending on the situation or circumstances. Hence why the thermostat is directly connected to the heating portion. The light sensor will be used to determine how much light is present in the room and according to the sensor, the lighting in the home will adjust accordingly. The mmWave sensor will be used in conjunction with the PIR sensor to track movements to see whether someone is in the present in the room or not. Depending on the activity in the room, the light will either turn on or off. The heat sensor will then act accordingly as it will increase or decrease the temperature in the room as it will know if someone is present in the room or not. When these automation are combined with the sensors and devices, it will allow for the homeowner to save more energy and money. As most traditional home systems make sure of a system that operates on a schedule, the home automation will allow the system to only operate when there are individuals detected and can turn off devices when certain parameters are set.

Finally, for the garage door activation and functionality, the following scheme will be used. NFC tags will be placed on the outside of the garage door and on the inside. The user's phone will scan the tag using a phone that is recognized in the system with the software and it will open and close the garage door. If the correct phone is used, the L298n (H-Bridge) will cause the DC motor to rotate either forward or backwards. This will cause the garage door to open and close depending on what is necessary in that situation. The motor will be powered by a 6 V power supply since the motor that will be used is a 6 V DC motor. All of these sensors will be connected using a wifi network, this means that all of the sensors will be connected to the same network. This will ensure that the system is smooth and it is in unison with one another.

Alternative Designs

Garage Door:

Before the final design of the garage door was determined, other ideas were explored and almost used. The plan that was created in the fall semester (ELE 70A) was seemingly very good but it was not compatible with the rest of the project. For that reason it was decided to be replaced by a much better and simpler design. However, the old design must be explained and commented on. Essentially the premise of the previous design was to have the system be directly connected to the Raspberry Pi and have its respective OS (operating system) control the motor when the correct RFID tag is presented. The initial block diagram and schematic that was scrapped and replaced is shown below.

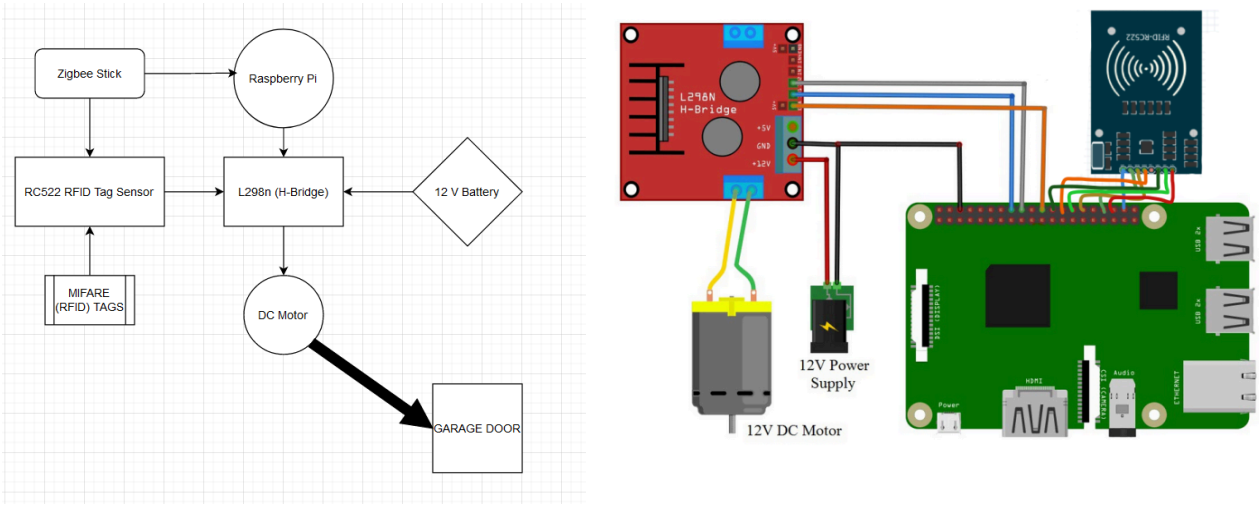

Figure 4.1: Old Configuration for the Garage Door

As seen by these diagrams, the goal was to have the RFID Sensor be used to identify if the correct vehicle is entering the garage or not.

However, the main setback here was that this system is not compatible with Home assistant, which is being used by the rest of the sensors that are present in this project. This meant that the whole design had to be scrapped to fit in with the rest of the system. Luckily, not much had to be changed. As stated above, the only thing that needed to be changed was the RFID Tag reader. This was due to the fact that the RFID Tag reader is not compatible with ESP 32. ESP 32 is a low-cost, low-power system on a chip microcontroller with integrated Wi-Fi compatibility. For this reason, the RFID Tag reader was completely removed from the project. This was replaced with the NFC tags which do not require a specific reader as the majority of the smartphones have NFC capabilities. This means that instead of being locked by a tag reader, anyone with access to the system will be able to enter and exit the garage. This is much better for security as well. This is because if the key fob for the RFID Tag reader is stolen or replicated,

anyone can enter the garage and safety is compromised. However, let's say one's phone was to be stolen or lost, the NFC tag can simply be turned off or replaced. This means that it is much more secure and easy to implement. All together, the current configuration for the garage door is much better and simple to use when compared to the old schematic that was initially going to be implemented.

Energy Harvesting:

Thermocouple: This device will be able to harness energy through a difference in temperature. Ideally this would've been used to produce power for the temperature and humidity sensor but as we can see from FIGURE 4.2 the thermocouple will not produce enough power at room temperatures, therefore it is not suitable for this project. Had the thermocouple been capable of producing enough power for each sensor then the overall circuit would be very similar to the one currently using solar cells. This can be seen below in Figure 4.3

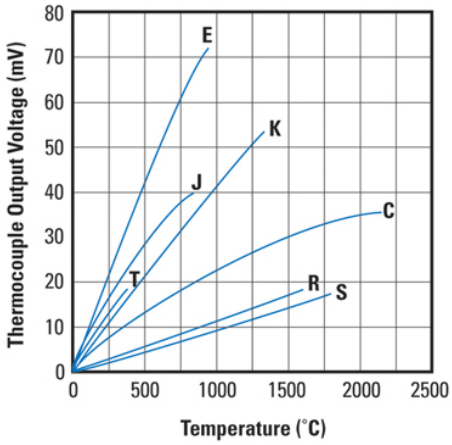

Figure 4.2: Thermocouple Temperature Vs. Voltage.

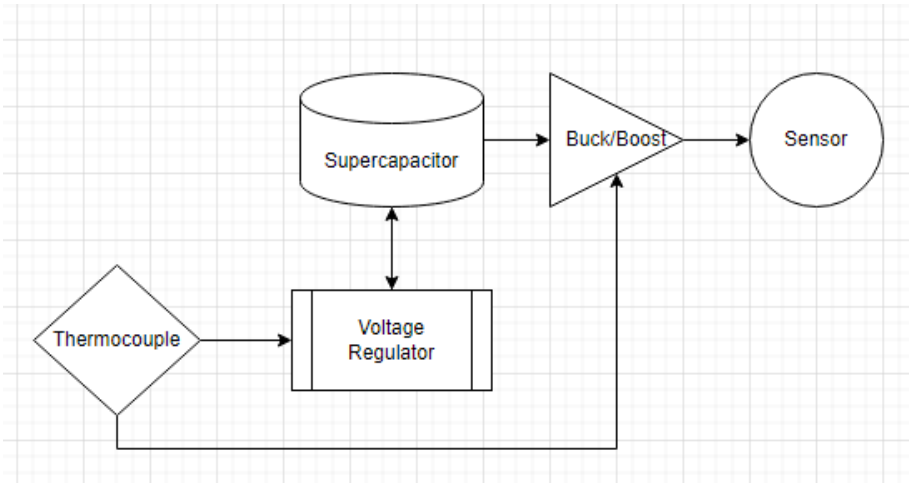

Figure 4.3: Thermocouple

Pressure Plates: A vehicle authentication system could implement this device but there are a few problems. If the vehicle is to be used to run over the plate, it must be able to withstand the weight of the vehicle. This would require very strong plates which are very expensive. Alternatively smaller plates could be used to activate the door opener but this would require the driver to exit the vehicle which is somewhat clunky.
Finally the last problem with this energy harvesting method is that the sensor will not always be on and therefore is not optimal in this project's design. To explain this; the power developed by the pressure plates will not be enough to keep supercapacitors charged enough to keep the sensor running until the next time the pressure plates are pressed down again. If the pressure plate would be able to keep the supercapacitor charged then the circuit diagram would look very similar to figure 4.3 where instead of a thermocouple there will be a pressure plate.

Sensors

There was a lot of research done into what type of sensors could be used for this project, and how it could all be connected together. After a lot of considerations the PIR, BME280, and BH1750 sensors were finalized as the key components. To connect these wirelessly we were aiming to use Zigbee initially. I got an SoC chip that was Zigbee compatible known as the MG22, however there were very few resources to provide guidance on how to assemble the connections as Zigbee is fairly new as well. After some group discussions we came to consensus that we would use Wi-fi to connect everything wirelessly. From this point the ESP32 was decided on to use as the microcontroller.

For the motion sensor a PIR sensor was decided for the basic requirements. The plan was to add a mmWave sensor along with the PIR so that subtle movements would be picked up as well. This was a good way to allow a person to be relatively still and still be detected, which the PIR sensor wouldn’t be able to do. Tasks such as typing, or reading a book wouldn’t cause the lights to turn off with a mmWave sensor. Although this seemed logical when planning it all out, when we finally went to implement it this semester with solar energy, it came to light that the mmWave was drawing too much current. Therefore this would not be a feasible solution.

In the process of the project the LD2410 sensor was also considered as an alternative option for a presence sensor. This was also a 5V device and had been used before in many projects. The problem that occurred with this sensor is that the pitch spacing was not of standard size. The part was bought from Amazon and did not come with the cable needed as the component was not of the standard 2.54 pitch. The earliest we could ship internationally was past the open house date which would be of no use. The hardware stores that were visited unfortunately also did not have the cable.

Integration and System Software:

As there are many different approaches that can be taken with the system integration. The main component that was needed was determining which home automation software we would integrate for the project. When looking into the market for software, programs such as Calaos, OpenHAB and Jeedom were considered due to the nature of its simplicity and multi-device support. Most of these software make use of different approaches to handle different networks but at certain times it can be very limiting to certain devices. The issue of being limited to certain networks and only allowing certain devices to be compatible with the software creates additional problems in the future. As such, the home automation software we choose to work with is Home Assistant. Home Assistant creates an open-source platform which allows for integration from many different networks such as Zigbee, Bluetooth, Wifi and much more. The software features plenty of add-ons which perform a multitude of different functions and the community creates helpful guides and forms that help new users. The software has a simple but customizable UI and the devices compatible with the software grow in quantity every day. Due to these reasons, Home Assistant was selected for the purposes of this design project.

After we have selected the software, we need an operating system to act as the brain and host the software. The two different hardware we looked at were the arduino and the Raspberry Pi. Since they both are boards that are compatible with Home Assistant, it came to performance needed for this project. Arduino is a microcontroller that can only handle simple tasks such as turning on LED or reading simple inputs like a button is triggered or light sensor. The Raspberry Pi on the other hand has its own opening system and can carry out more complex tasks such as monitoring multiple sensors and using the internet. In more simple terms an arduino is a microcontroller and a Raspberry Pi is a microcontroller. Due to the nature of the project we decided to make use of wifi and bluetooth capabilities of the Raspberry Pi for ease of use and multi network connection.

Originally we wanted to make the whole system sensor use a Zigbee network, however after some deliberation with the project members and permission from our FLC, we came to the conclusion of using any integration. The main reason for the change is the convenience it would provide the user as a typical household would use wifi and as such we aimed to use wifi more but with a combination of different integration.

Material/Component list

The table below shows the cost of all materials and components used in the project. Both unit cost and the quantity are shown below.

Table 5.1: Total Cost

Material/Component	Units	Cost per unit
Raspberry Pi 4 Model B 8GB Set	1	$419.99
AEMLIC	4	$29.90
Solar Cell	4	$18.92
Lithium Ion Capacitor	4	$9.03
L298n H-Bridge and DC Motors	2	$28.99
Wires	10	$0.49
NFC Tags	5	$59.99
BME280	1	$4.29
BH1750	1	$0.99
ESP32	1	$17.72
PIR	1	$3.98
Total Cost		$594.29

*All cost will be shown in CAD

Measurement and Testing Procedures

This section covers the measurement and testing procedures that were used throughout the completion of this project. These measurements include the voltages on the ESP32 GPIO pins. All of the measurements taken helped gauge how well this project was functioning.

For the board and microprocessor, it was confirmed that an ESP 32 was going to be used, thus a system that would work with ESP32 GPIO pins would have to be chosen. The GPIO pins on the ESP 32 provide around 3.3 V. The detailed diagram was shown in the ***Theory and Design*** section.

Garage Door

In order to spin the motor a proper H-Bridge and DC Motor had to be chosen. As stated earlier, for this project the L298n H-Bridge was chosen as it is able to handle anywhere from 6 to 12 V. This was paired with a DC Motor that is rated for 6 to 12 V.

Another interesting thing that was noted was that when the motor spins at a faster speed when it is counterclockwise when compared to clockwise. This is probably due to the fact that the motor was designed to spin counterclockwise as the L298n H-Bridge is commonly used for moving robotic parts such as wheels.

When coding the L298n H-Bridge component to the ESP32 using Home Assistant, it took alot of testing to find the correct pins that were responsible for pulse width modulation (PWM). This was responsible for controlling the bidirectionality of the motor as the project requires it to spin both clockwise and counterclockwise. Once the PWM pin was determined, the code was easy to implement and incorporated into the project. The code that was created used the YAML format. Using this code, two automations were created in the Home Assistant interface. The automations were also coded using the YAML format. These two automations were responsible for controlling the motor and making it go clockwise and counterclockwise. When Tag 1 is scanned using an NFC compatible phone, the DC Motor will rotate for 5 seconds clockwise. When Tag 2 is scanned, the DC Motor will rotate for 2 seconds counterclockwise. After a lot of testing, it was determined that the ideal time for the DC Motor to rotate should be for 5 seconds in the clockwise direction and for 2 seconds in the counterclockwise direction. As stated earlier, the reason that the DC Motor will rotate for less time in the counterclockwise direction, is because the L298n causes the motor to spin faster in the counterclockwise direction as the L298n was designed for robotic parts that are used to spin wheels.

Solar Harvesting

1. To test whether the circuit will be able to harvest energy into the supercapacitor we can simply disconnect the load. With only the solar cell and supercapacitor connected to the AEMLIC the capacitor will not discharge and will be charged by the solar cell. To test this we can take a voltmeter and measure across the capacitor over time. This will vary as the solar cell will produce different amounts of power based on the light level.

2. To test whether the circuit will be able to power sensory devices, connect the sensors to the regulated 3.3V output of the AEMLIC while at least the supercapacitor is connected but ideally with both the supercapacitor and solar cell. To measure the time it would take to discharge refer to equation (3).

System Integration

When operating the software there are several methods in measuring and testing the procedures, as to not overcomplicate things there are simple procedures that can be performed depending on the sensor being operated. The procedure can be as simple as doing a physical movement and checking the response of the system. For the PIR motion sensor and the mm-wave sensor, a motion such as waving my hand passes the sensor to see if the system triggers. Since both these components are motion, we only need a binary component to indicate true or false. The BME280 measures the temperature of the area, as such we can let it operate and confirm with a thermometer, if there is a difference the values can be altered. The BH1750 measures the lumens present in a room, the method of testing this system is to cover the sensor with a cloth and show the change in real time happening on the software. The L298n motor driver proved to be slightly different as the system required the use of automations to test if the component functioned properly.

Software Integration

The software integration required the need for the sensors to be tested individually for accuracy. After all the components were connected together, the sensors were tested on the Home Assistant overlay and the history of each component can be displayed for the user. The true test came in the form of automations as the sensors would be connected to an action which will automate a certain device. The L298n motor requires for a motor to spin forwards and backwards, this can only be achieved with automation. The automation for all components are relayed back to a certain device which can be triggered with either the sensor itself or a trigger. All sensors will relay from the esp32 to the software, the dashboard can update the user periodically of data and an additional test can be completed to create automation using the sensor data.

Performance Measurement Results

Detail the obtained simulation results.

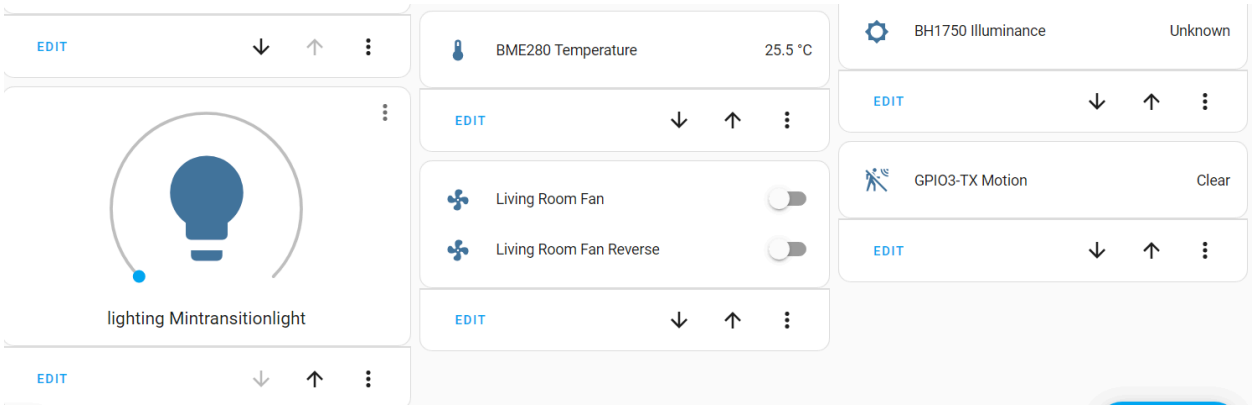

Figure 7.1 Software UI

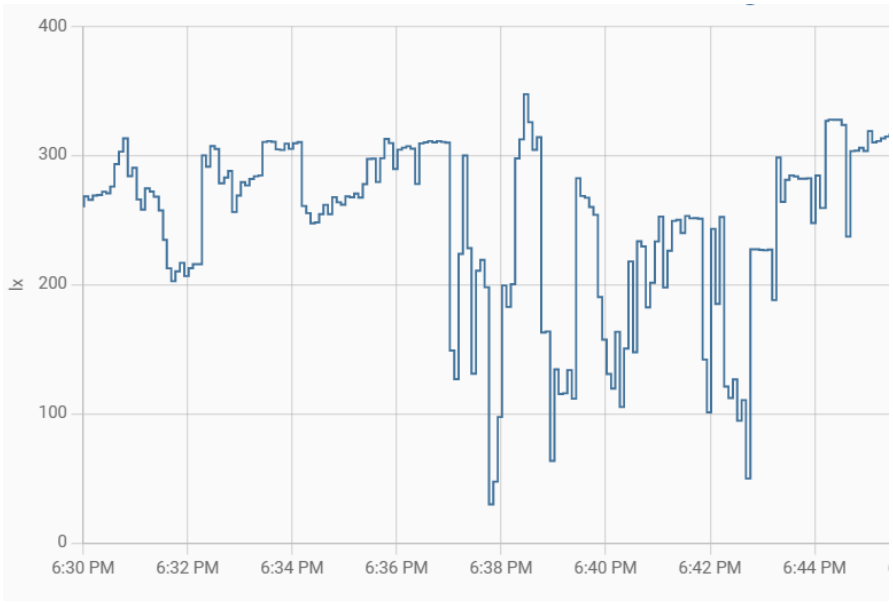

Figure 7.2. Lumens obtained from Sensor

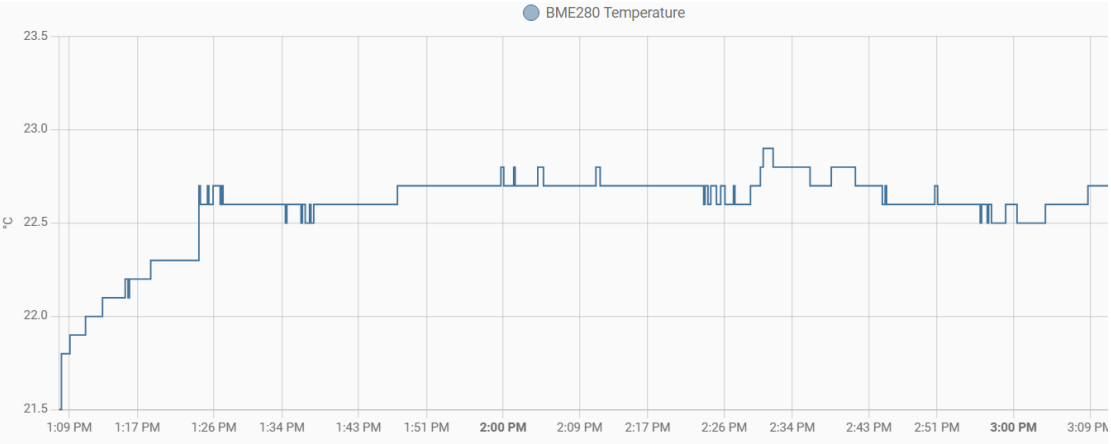

Figure 7.3. Temperature obtained from Sensor

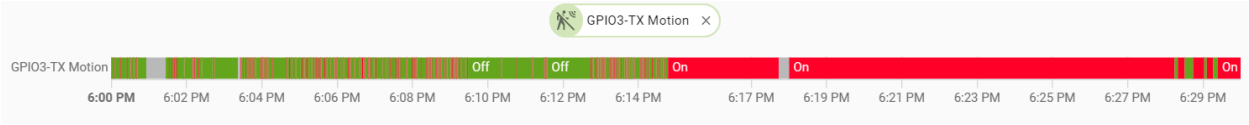

Figure 7.4 Motion obtained from PIR Binary Sensor

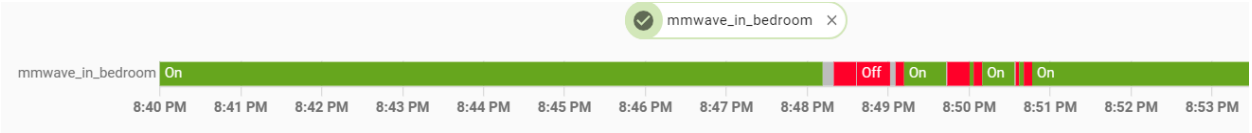

Figure 7.5 Motion obtained from Millimeter wave Binary Sensor

Solar Circuit Data:

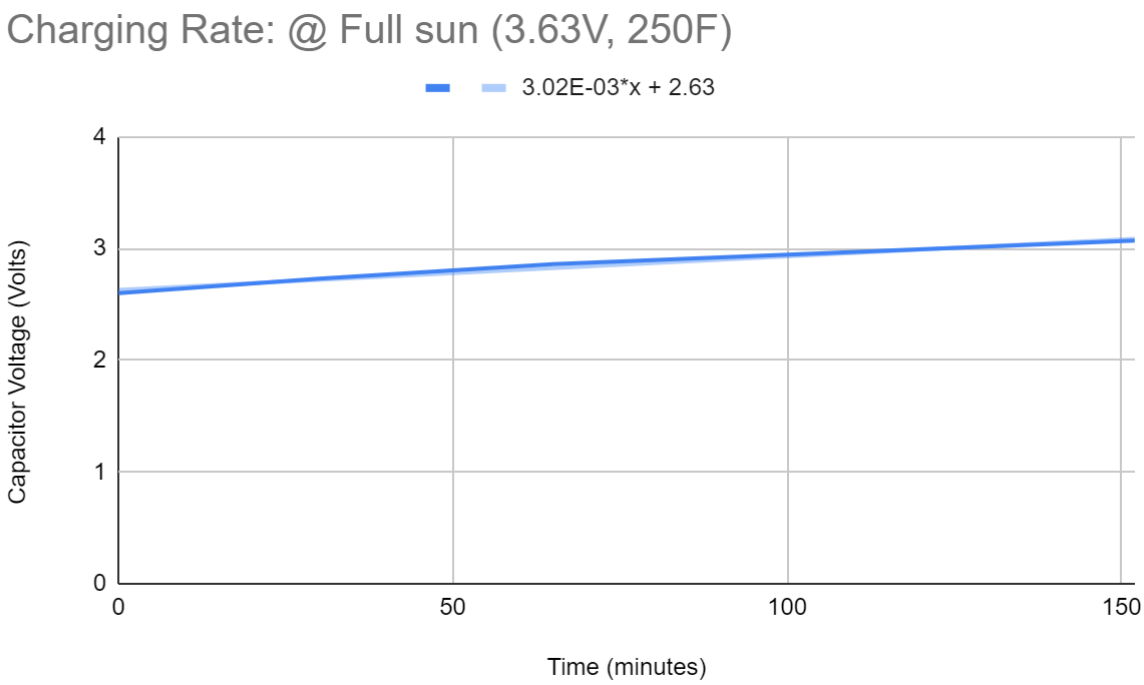

Figure 7.6 Charging 250F Capacitor (no load at full sun)

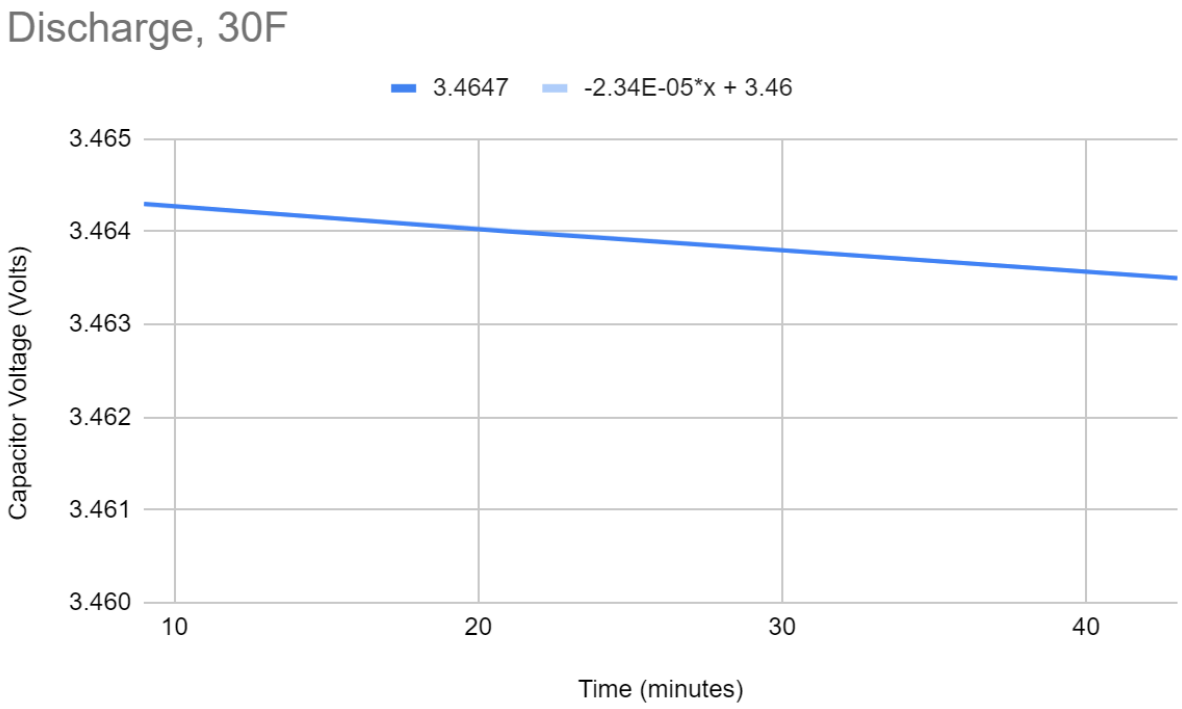

Figure 7.7 Discharging 30F Capacitor (no sun, load = light and heat)

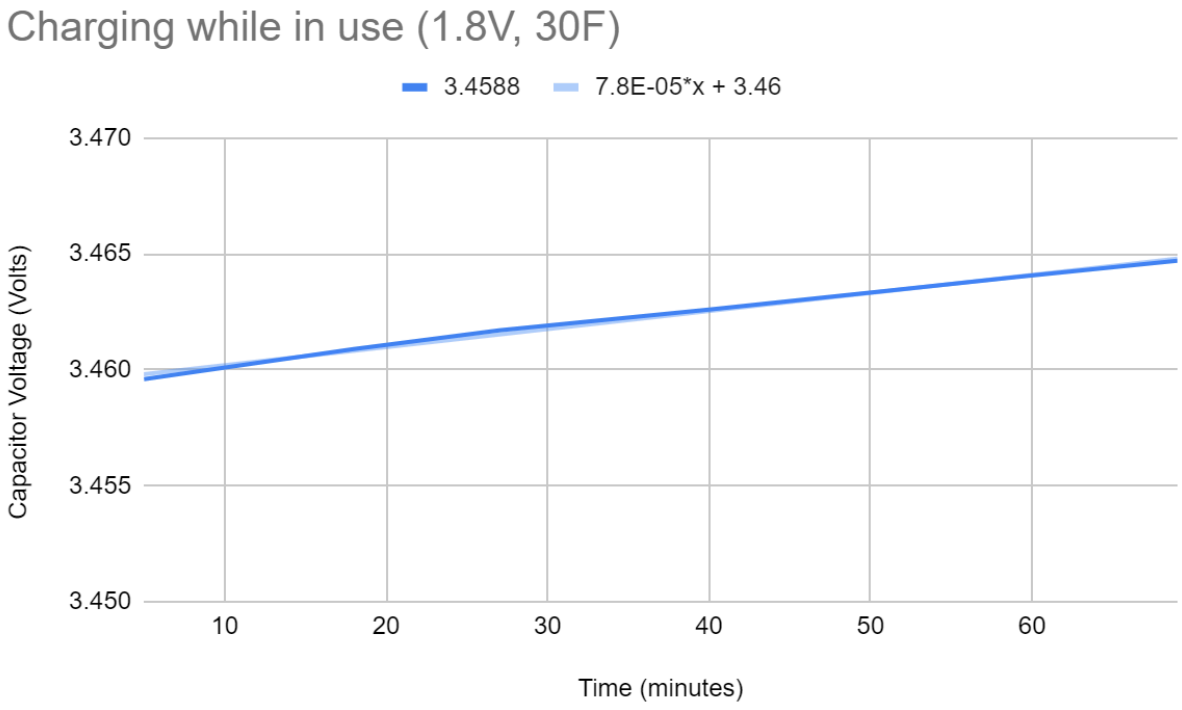

Figure 7.8 Charging a 30F Capacitor (indoor lighting, load = light and heat)

Analysis of Performance

From figure 7.7 we can see that the 30F supercapacitor discharges at a rate of 2.34×10^{-5} $[V/min]$, and from figure 7.8 we can see that the 30F supercapacitor charges at a rate of 7.8×10^{-5} $[V/min]$. In figure 7.7 we are running two sensors, which are the light and heat sensors, through one 30F supercapacitor while there is no solar panel connected; this emulates the scenario of very low to no light in which the system will still be able to operate. In figure 7.8 we run the same two sensors but this time the solar panel is connected to the circuit. During the testing the solar panel was receiving typical indoor light (approximately 300 lux) which measured across the solar panel as 1.8V. With not even full power being received through the solar panel the supercapacitor was able to provide power to the sensors while also being able to charge up.

In figure 7.6 we ran a test on the 250F supercapacitor as these will be better used in this project. Simply having more capacity is much better as if there is a long spell of low light the system will still be functional. Although the charge time is very slow, taking a little over two hours to go from 2.5V to just barely above 3V while the solar panel was providing near full power, the discharge time as shown in figure 7.7 shows a much smaller rate at a lower capacity.
From this we can conclude that the sensors will be able to work indefinitely and thus the objective of providing a battery free experience has been achieved.

In figure 7.1, we can observe the software UI that the user can interact with, this dashboard shows all the components that we can integrate with and the dashboard can also display devices we have connected to. This user interface can be customized to fit any needs and is it possible for all devices to be displayed on the dashboard to test individually.

From figure 7.2-7.5 we can observe the data obtained from the sensors, the data input from both the binary sensors and the sensors. The binary sensors only indicate true or false and can be used for more simple sensors such as motion triggered. The BH1750 on the other hand will operate in an interval preset by the user and can determine the light in the surrounding area, this requires a sensor. Figure 8.4 shows the required code for the L298n motor which is connected to a DC motor and enables the motor to spin forwards and backwards. The code displays the motor spinning forward after a trigger of the NFC unique tag after which it will spin for 500 milliseconds and stop. The purpose of the small time for open-house demo purposes as this would be extended to have a longer time for real life scenarios.

Conclusions

When all the components are pieced together, the system works seamlessly. The battery-free energy harvesting component of the overall system has been completed. This entails using a solar cell to charge a supercapacitor which will then power each of the required sensors. The initial design proposes that the heat sensor be powered through the heat in the room. Ultimately it was decided that it was unviable to power the sensor in this fashion therefore the heat and humidity sensor will use solar cells to power them. Powering the ESP32 using the solar circuit is spotty and not giving good results. Tests will be run regarding how long it takes to charge the capacitor, and how long it will take a capacitor to discharge without having a solar cell connected. Using these tests, it has been concluded that the ESP will be run using the grid, but all the sensors will be powered using the solar panels. All of the required components have been tested and soldered onto PCBs. A mock "house" has also been created so that it can look polished and presentable during the open-house. A mock garage has also been created as the current garage door is very primitive. The components for the garage door have also been soldered on PCBs with the main system. All together this project has been completed fully and all the shortcomings that were present from before have been solved and corrected.

References

[1] *AEMLIC v2 has 3.3V/400mA output: Details*. Hackaday.io. (n.d.). Retrieved April 8, 2023, from https://hackaday.io/project/178177-solar-harvesting-into-lithium-ion-capacitor/log/202149-aemlic-v2-has-33v400ma-output

[2] Dejan. (2022, February 18). *How I2C communication works? arduino and I2C tutorial*. How To Mechatronics. Retrieved April 8, 2023, from https://howtomechatronics.com/tutorials/arduino/how-i2c-communication-works-and-how-to-use-it-with-arduino/

[3] esp32tutorialslab. (2022, September 25). *BME280 with esp32 esp-IDF and display readings on OLED*. ESP32 ESP-IDF. Retrieved April 7, 2023, from https://esp32tutorials.com/bme280-esp32-esp-idf-oled/

[4] Ghosh, A., Zhang, C., Shi, S. Q., & Zhang, H. (2019). High‐Temperature Gas Sensors for Harsh Environment Applications: A Review. Clean : Soil, Air, Water, 47(8), 1800491-n/a

[5] Hansie, Sianturi, J. K., Bert, O. man, Santos, S., & Bert, old man. (2022, March 8). *ESP32 with BH1750 ambient light sensor*. Random Nerd Tutorials. Retrieved April 7, 2023, from https://randomnerdtutorials.com/esp32-bh1750-ambient-light-sensor/

[6] Kurniawan, A. (2019). Internet of Things projects with ESP32: build exciting and powerful IoT projects using the all-new Espressif ESP32. Packt Publishing

[7] Leslie, Aasestrand, P.-T., Tannenbaum, J., Paulo, Gary, Albin, T., Santos, S., Garber, D., Stockton, A., Denner, T., seta43, Joel, Horssen, I. van, gupta, S. kumar, & Mel. (2020, July 30). *ESP32 with BME280 using Arduino IDE (pressure, temperature, humidity)*. Random Nerd Tutorials. Retrieved April 7, 2023, from https://randomnerdtutorials.com/esp32-bme280-arduino-ide-pressure-temperature-humidity/

[8] Oner, V. O. (2021). Developing IoT projects with ESP32: automate your home or business with inexpensive wi-fi devices. Packt Publishing

[9] Shevchenko, G. V., Glubokov, N. A., Yupashevsky, A. V., & Kazmina, A. S. (2020). Air Flow Sensor Based on Environmental Sensor BME280. 432–435

[10] -Website Author Syed Zain Nasir syedzainnasir I am Syed Zain Nasir. (2022, February 8). *Motion detection with ESP32 & pir sensor*. The Engineering Projects. Retrieved April 8, 2023, from https://www.theengineeringprojects.com/2022/02/motion-detection-with-esp32-pir-sensor.html

[11] *What is the internet of things (IOT)?* What Is the Internet of Things (IoT)? | Oracle Canada. (n.d.). Retrieved April 8, 2023, from https://www.oracle.com/ca-en/internet-of-things/what-is-iot/#:~:text=What%20is%20IoT%3F,and%20systems%20over%20the%20internet.

[12] Yar, H., Imran, A. S., Khan, Z. A., Sajjad, M., & Kastrati, Z. (2021). Towards Smart Home Automation Using IoT-Enabled Edge-Computing Paradigm. Sensors (Basel, Switzerland), 21(14), 4932.

[13] *BH1750 ambient light sensor*. ESPHome. (n.d.). Retrieved April 24, 2023, from https://esphome.io/components/sensor/bh1750.html?highlight=bh1750

Appendices

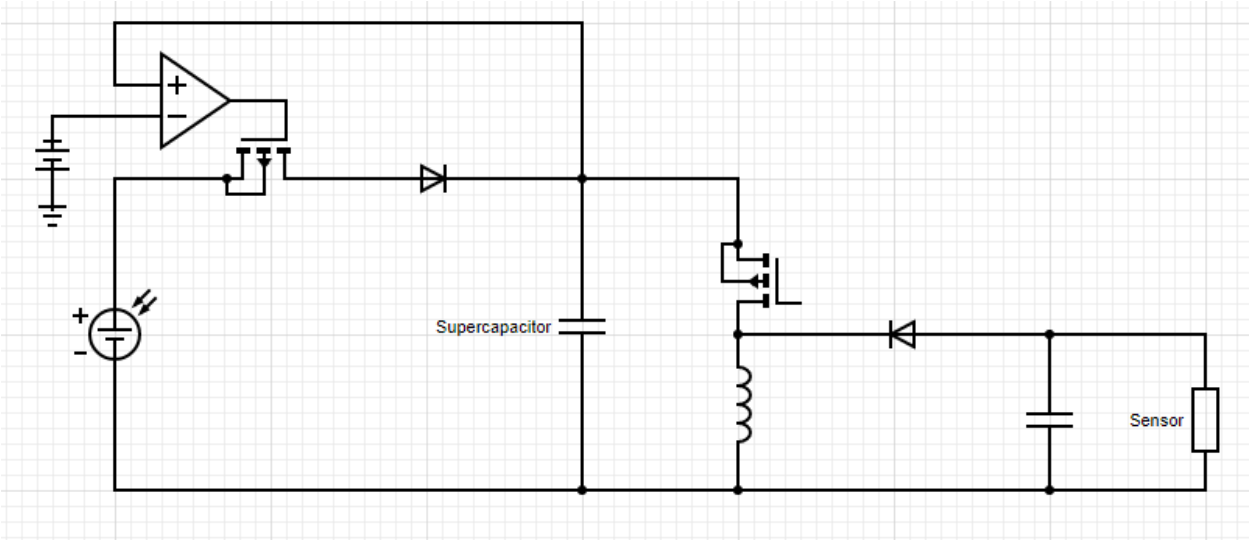

Figure 8.1: Updated Solar Harvesting Circuit.

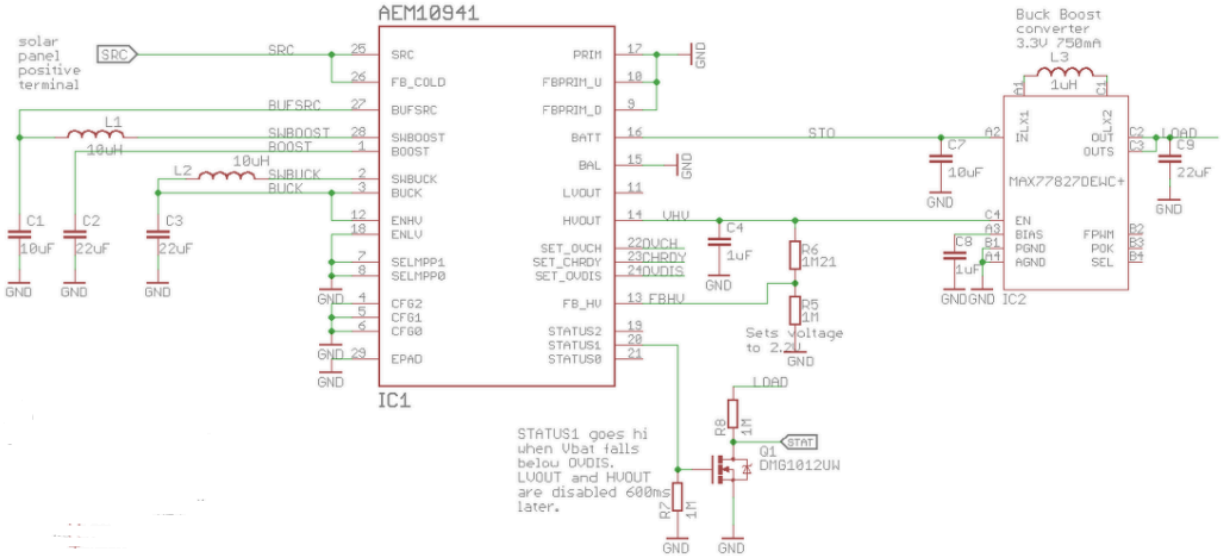

Figure 8.2: Updated Solar Harvesting Circuit ([1] AEMLIC).

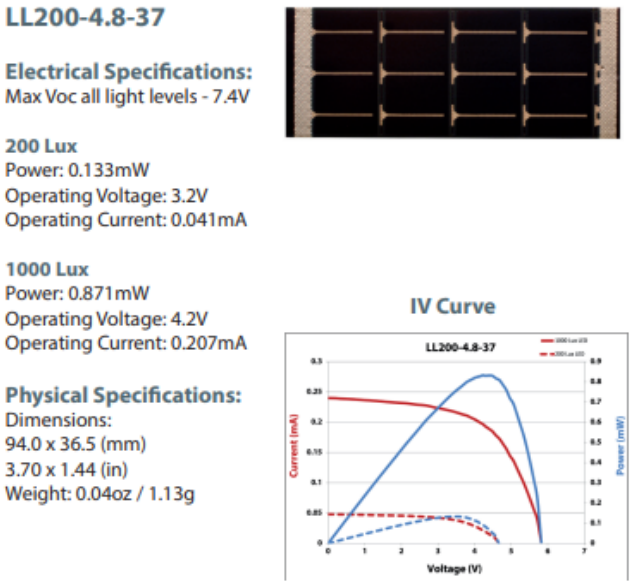

Figure 8.3: Solar Cell Datasheet

```yaml
alias: motor
description: ""
trigger:
  - platform: tag
    tag_id: 1dac42db-c30f-4b01-84f7-0ad6ed85fe64
condition: []
action:
  - service: fan.turn_on
    data: {}
    target:
      entity_id: fan.living_room_fan
  - delay:
      hours: 0
      minutes: 0
      seconds: 0
      milliseconds: 500
  - service: fan.turn_off
    target:
      entity_id: fan.living_room_fan
    data: {}
```

Figure 8.4 Motor Forward automation