Senior Design Final Draft

Dec 4, 2016

DEPARTMENT OF
ELECTRICAL & COMPUTER ENGINEERING

UNIVERSITY OF CENTRAL FLORIDA

Home Secured

Department of Electrical Engineering and Computer Science University of Central Florida

Dr. Samuel Richie & Dr. Lei Wei

Senior Design 2

GROUP 8

Name	Email	Degree
Phillip Dannelly	pdannell@hotmail.com	EE
Joshua Fry	jfry321@yahoo.com	EE
Tony Baran	Tony.wazid@gmail.com	CpE

A. TABLE OF CONTENTS

1) EXECUTIVE SUMMARY

Our home security system is meant to introduce a host of new features not normally found within alarm systems and to add a bit of convenience to the end user of the product. Our vision for the project is to offer a security system with no strings attached to the user and to allow the user to easily setup our system. Our design philosophy revolves around ease of use and affordability for the user without trying to nickel and dime them at all turns. Ideally, we are offering a product that can function as a sole item and only should need to be purchased once when it comes to fulfilling the need of the most common demographic.

The implementation for our system revolves around common systems easily available but generally aren't implemented together. This means we had to research various parts and figure out the viability to use them for our project, based on our constraints in order to meet our goals. The core of our system involves software and hardware based on the Raspberry Pi microcontroller technology coupled with an LCD. Users primary way to interface with our device is through the use of a keypad we created in order to cut down need for excess external items such as a smart phone and such. The only other way users can interface with our device is using an RFID tag for convenience of disarming/arming and keeping track of who touched the system. A basic wireless communication aspect was required to allow communication between our various systems which aren't all interconnected. Our system does have a power system attached to it which also required a lot of research in order to provide a safe system without damaging our item or posing risk to the public. This required extensive research into rectification technologies along with working with part vendors to help meet our exact needs. Transformer technology also paid a major role into ensuring we had a safe product that wasn't dealing with very big voltages.
Parts available for use in a printed circuit board were a decently big consideration for our home security system and implementation.

Our home alarm system is a solution targeting virtually all groups who have a home or are renting somewhere as it is easy to install and remove. Our only real constraint imposed on our system was a budget constraint and time constraint for implementation of our system. In terms of design decisions for our system, we had a decent amount of leeway as most systems on the market are greatly over our cost or try to have reoccurring charges. Our system has a possibility to appeal to masses as an alternative to the current most popular form of home security. On the risk side for our project to the end user there virtually is none as we don't keep track of any important information of the user and it isn't connected to any external networks.

2) PROJECT DESCRIPTION

The project aims to introduce a few features not generally found within most home security systems. Our home security system which we will design, build and test will be a standalone system that requires no extra products to function. The objectives of this senior design project will be focused on gaining experience with sensors, microcontrollers, programming, specifications, standards, power systems and just all around learning to handle projects as a team.

2.1) PROJECT MOTIVATION AND DESIGN CONSTRAINTS

Our home security system was picked primarily because it gives a very broad range of systems to work with as we apply our knowledge as Electrical Engineers in general. This is while still being able to add a little bit of innovation to a common commercial product that is found within many of our homes. Our only main constraint that limits us is time, money and power limitations in general.

2.2) OBJECTIVES

The following is a breakdown of all objectives and goals we intend to meet for our project. This starts with our overall views and expands downwards into our actual goals for each item.

2.2.1) USABILITY

Our number one absolute important feature for our system is the ability for virtually anyone to be able to use it and understand it. This means most of the system is to be setup in a way where there isn't too much hassle in dealing with complicated options and such.
The philosophy when it comes to usability of our product is that simplicity is king. Our target age range in terms of operating the most basic feature of our system which is our RFID implementation would be so that it is simple enough for even a 6-year-old to utilize it. For the full range of features we would be aiming for it to be easy enough for even a tween to use it, which would also cut down on some of the hassle and issues some of the older generation of people tend to run into with these kinds of products.

2.2.2) EASE OF SETUP

Our second real objective for our system that follows our usability philosophy is that it is simple to setup. This means that the vast majority of our instructions for how to set and forget the product should essentially be the equivalent of a quick start guide that many other products contain. We are aiming to keep the basic setup of each feature to be easily summed up within five steps or less. If possible, one of our objectives is to make it where someone with a decent grasp of electronics could setup the system without a single instruction read about the system and without confusion about how to utilize it from there on out. This is because it is extremely common nowadays for many people to actually not read instructions as we currently live in a world where most things are plug in play. When items don't fall into a plug and play philosophy like we are going for, many consumers tend to get unhappy especially with lengthy reading. In many cases this means they may actually rate the product worse or return the item in question.

2.2.3) AFFORDABILITY

Our last real goal before we go into the extras added to our consumer product versus ones on the market, is that the item is affordable. One of the reasons this is such a big goal for us is we want to be able to target a much wider amount of consumers with our product and innovations. This is really imperative to the success of a consumer product such as this as the cost definitely plays into how people view a product and the company. It not only plays into people's view on a product, but the market for this type of item has a lot of competition in general. If an item is too expensive and never receives the exposure needed, it essentially fails to help innovate the market for the product. Ideally our product would fall to the lower middle side of cost in general for this type of item.

2.2.4) GRAPHICAL USER INTERFACE

Our first goal in terms of new features is a graphical user interface in which the home user can interact with. This is the basis for most of our other features to be added and to keep the system simple for the user while still adding more features. Our ideal goal for the graphical user interface is to have a simple menu that will be navigated using our push button keypad. The menu will use colors to help keep the selection process simple for the user and display text for options. In traditional alarm systems there are generally complicated processes for setting things such as passcodes. With our system it will only require selecting a password option on the screen for example. This means the user will also be able to do things such as view their inputs visually rather than temporary memorization. For people growing up in today's society with smart phones and even touch screen ordering from restaurants, this means they will feel right at home due to the graphical user interface. From knowing if the system is armed to seeing logs of who disarmed and armed the system, most of this isn't possible without this addition.

2.2.5) SOUND

Sound is an important goal for us to implement for a variety of reasons. The first major reason we have this as a goal for our project is to help alert the user. This is because if the user is only mildly paying attention or not in visual range of the system, it becomes impossible to give them important notices they need. We want the user to know if someone is accessing their door or simply breaking in so they have a mild advantage to the situation. The secondary function of it is that sound can be a powerful deterrent to an intruder for example as they are less likely to stay around if they know a person or system is on to them. The very last reason this is an important goal for us is that sound just adds to the presentation to the system for most people. Whether it is a refreshing jingle, warning beeps or just alerting people to activity in their homes, we feel sound will play a vital role to making the product appear as professional as possible.

2.2.6) MOTION SENSOR

Our overall objective when it comes to our system is to provide at least a basic amount coverage to the user. We don't want them to feel as if they aren't receiving the best possible coverage or at least an acceptable amount of coverage within the home. The vast majority of systems offer two types of motion sensors in general, which are ones that operate at a distance and ones that can be attached to windows. We plan to offer the

same selection as these other systems in order to meet the current standards. Ideally we would have one sensor that can just be mounted anywhere the consumer likes. We don't want them to be overly complicated to setup in order to follow our philosophy of ease of use for the consumer. In order to maintain this objective, we plan to make them so they don't require any special setup. Ideally this means you can just take them out of the box, insert batteries and put them where you want and your all set. This is where our actual difference will come within our sensors versus other systems on the market. This however, means we will have to trade off some of our accuracy as to which sensor was tripped as a tradeoff. That does make it far easier for a consumer to expand their system to their liking as they can just order more sensors without issue and keeping in line with our goal of ease of use.

2.2.7) TILT SENSOR

Our tilt sensor is actually a brand new type of sensor not commonly found with other systems currently on the market. Our objective for this type of sensor is not only to be reactive in the event of a break in overall, but to actually be a preventative measure which may alert the owner of our system to a break in before it even occurs. The basis for this type of sensor and how we want it to function is simple. We want it to be attached to a door handle and trigger an alert no matter what our alarm status is set to. This means if someone tries your door handle to see if your door is unlocked, you will know about it. This gives you time to see if there is an intruder scouting out your location or even if an intruder is about to follow up and break into your home. Virtually everyone has glass in their front door or even a peephole to see who is at the door so this will be applicable to everyone who owns an exterior door essentially. Now in order to maintain our ease of use philosophy with this type of item, we plan to have it possible to just remove from a box and attach to the underside of a door handle with ease. Like our other sensors, our objective is to make adding more of them require virtually no effort if you want to expand how many are active with the system. However, this item in particular is meant specifically so exterior doors, so we don't plan to have all handle types to be supported.

2.2.8) RFID

Our objective in the case of our RFID system, is that it is the pinnacle of our ease of use model when it comes to our security system. This is the feature we want to be the most easily useable system when it comes to our home security system. We have the goal for RFID system to be simple enough for a child to use in the end. Our view for this system is that it will be a simple swipe to activate and deactivate using our RFID system. There should be no setup to use this system by default, other than having the base of our system plugged in like normal. This means that no physical setup on the base itself should be required to accomplish our goal. It will however still offer some customizability of it that isn't required to function. We do want to give users the options to set some of the data on the RFID for identification purposes through our system or through a commercially available application for the smart phone in the end. The data we have plans to display utilizing the RFID will go to the system logs alone in order to denote who was accessing the system.

2.2.9) BATTERY BACKUP

One of the biggest worries for the system is that a momentary power flicker would cause loss of power within the system or cause the system to reset. The biggest objective is that the system acts like it has a small battery to protect against momentary lapse in home power. This means that essentially we want for our system to have at a minimum a capacitive battery to keep the power going in moments of very temporary power loss. Ideally we have the objective that the capacitive battery should only last about thirty seconds or so in event of total power loss to protect against flickers. The secondary objective is for an extra battery backup system that can be added if wanted. This is in order to save some money for the end user and be available as sort of an add-on if the user deems it necessary. This isn't a core planned objective for the system however, but it is one we still plan to at least leave for future implementation if wanted. Ideally this secondary system would last far longer the our primary one. However, the system will add a tiny bit of complication when it comes to utilizing it for the user. We believe, we can still make it follow our ease of use model though by still making it relatively easy for the user to setup.

2.2.10) SYSTEM LOGS

System logs are a feature we have planned that isn't a standard item in today's alarm systems in general. Our objective for it, is that we plan for it to be a system in which will have multiple utilities to it. We want this system to be one that primarily is driven off the systems armed and disarmed status while keeping a record of the time it occurred. This means that the owner of the system will be able to utilize it for multiple purposes such as knowing if their kids or significant other disarmed the system for any reason at all. The secondary effect of this is that tampering with the armed and disarmed status of the system becomes far harder to do. We also aim for this system to be the primary extra feature of our RFID system. We want the RFID chip which is programmable to be able to store data that will interface with the system log to record things such as the exact person's name who disarmed the system for the user. This means that there would be minimal confusion about who had accessed the system during certain times of the day.

2.2.11) PUSH BUTTONS/KEYPAD

In order for our security system to be fully functional and to meet our goal of being user friendly, we have a major objective that our keypad is simple and straightforward for the user. This system has to not only be user friendly to follow our goals but also use as little power as possible. Like with most security system panels, our design will be following suit to the current standards laid out by the majority of the market. Our main objective is to have an input device that will enable the user to navigate the menus on the graphical user interface of the LCD with ease. Ideally, there shouldn't be a single button missing that a user may require in order to operate the full host of features for our system. The pushbuttons should let the user enter passcodes, arm or disarm the system, or silence the system for when a breach is detected. The keypad is an extremely big objective for our system, since it is so essential to its function as a whole. In order to complete our goals, we need to develop a keypad that will not only let user navigate to the different menus on the screen (settings, arm, disarm etc.) but to allow our users to enter virtually any password that our system allows. Another objective that we will be working towards is to design the input device to use very little power. This is an extra goal for us in order to assist with

making our system friendly not only to the user, but also to the environment when not in use. As a team we feel that if we keep the total power consumption low, then we will have completed our personal goal for this part of the project.

2.2.12) DOOR CHIMES

The main objective of our door chimes is to alert the user when a door has been opened when the system is not armed. When the system is armed and a door has been breached, then the security system will trigger an alarm to scare off any intruders. As a whole, we feel that if we don't include an added security feature to the entrances, then our design would not only be obsolete, but also easily bypassed. Reason being is because if an intruder were to breach an entrance, then their only concern would be avoiding our motion detectors.

Secondly, we will need to develop a door chime system that will use power only when a door is open. This means we need to include a certain type of mechanical switch that will close as soon as a door is open. Completing this goal should be fairly straight forward since it is just a circuit with a simple switch. Never the less, it should be an efficient design so that the main panel uses as little power as possible with respect to our peripheral devices that we are implementing. Also, the door chime switches will only be used on the outside doors of the user's house. It would be redundant and annoying to implement them on the inside since a chime would sound every time a door is opened.

Secondly, we want to design a door system that will only use one port on the Raspberry Pi GPIO pins. Again, this is a fairly straight forward process which we are confident enough to complete rather quickly. By completing this goal, we will be able to attach, in theory, an infinite amount of doors to a single GPIO pin which will use the same amount of power that a single door switch will use.

2.2.13) USB CHARGING

Our goal when it comes to USB charging is to add additional convenience to user. This is a feature that isn't exactly some new amazing idea for public consumption, however, it is new when featured to a security system. One of the reasons I believe this is an actual good addition to the system is because of the location the alarm system will be sitting. The plan is to have it sitting on a table although, wall mounting is an easy option for us. This is an ideal location because traditional wall chargers require an entire separate purchase along with being in less than ideal locations such as in wall sockets that require bending down. This means that the end user has to deal with long USB cables and have them fall off tables and such. The secondary reason it's a decent convenience item for the end user is it really doesn't add any cost to our design along with the fact that everyone is already familiar with these type of items. Part of the reason it doesn't cost extra for us is because phones and most devices have gotten so advanced they can determine what amount of Amperage is needed and limit it itself along with being the same voltage as our main power source.

2.2.14) TYPES OF AVAILABLE MOUNTING

One of our objectives for our project is to allow two types of mountings available for the base device which is wall mounted or just sitting on a surface. Part of this is giving consumers more choice in terms of utilizing our item and it allows the consumer to select

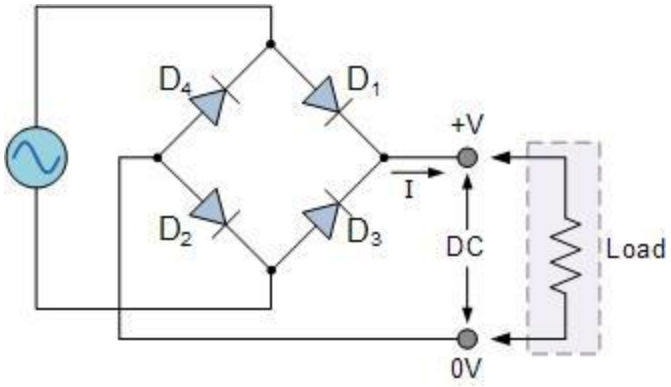
the type of surface they would like the item to sit on. This adds to the variety of ways the consumer can not only utilize our product but slightly increases the market for our item. This is mainly because not everyone may have a spare table to just sit our product on in general so they can still use it even if it's not the ideal method for our setup. The only real objective to be considered with this is that the device should look decent on a flat surface or a wall surface.

3) RESEARCH

3.1) REFERENCE DESIGN

3.1.1) FULL WAVE RECTIFIER CIRCUIT

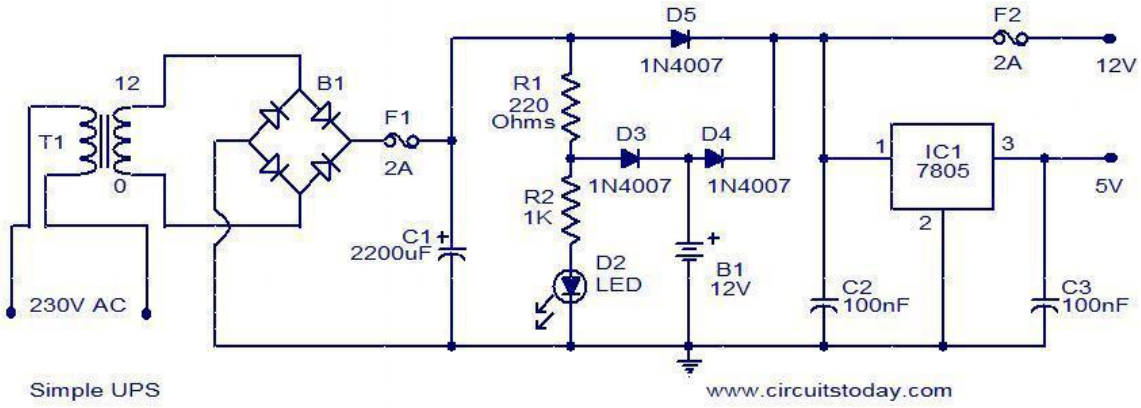
The following is a reference design for a commonly found full wave rectifier circuit in which four diodes are used to rectify an AC wave.



*Diode bridge rectifier reference design by www.electronics-tutorial.ws [1]

3.1.2) CONTINUOUS UPS CIRCUIT

The following is an entire UPS circuit involving a European AC wall plug. When power is lost the diodes allow passing of current and voltage from a battery.



*Continuous UPS Circuit by http://www.circuitstoday.com/ [2]

3.1.3) RFID READER CONFIGURATION

The reference schematic below contains all the component that the PN532 needs to function with the appropriate values that is designed with. This diagram is from the datasheet for the reader.

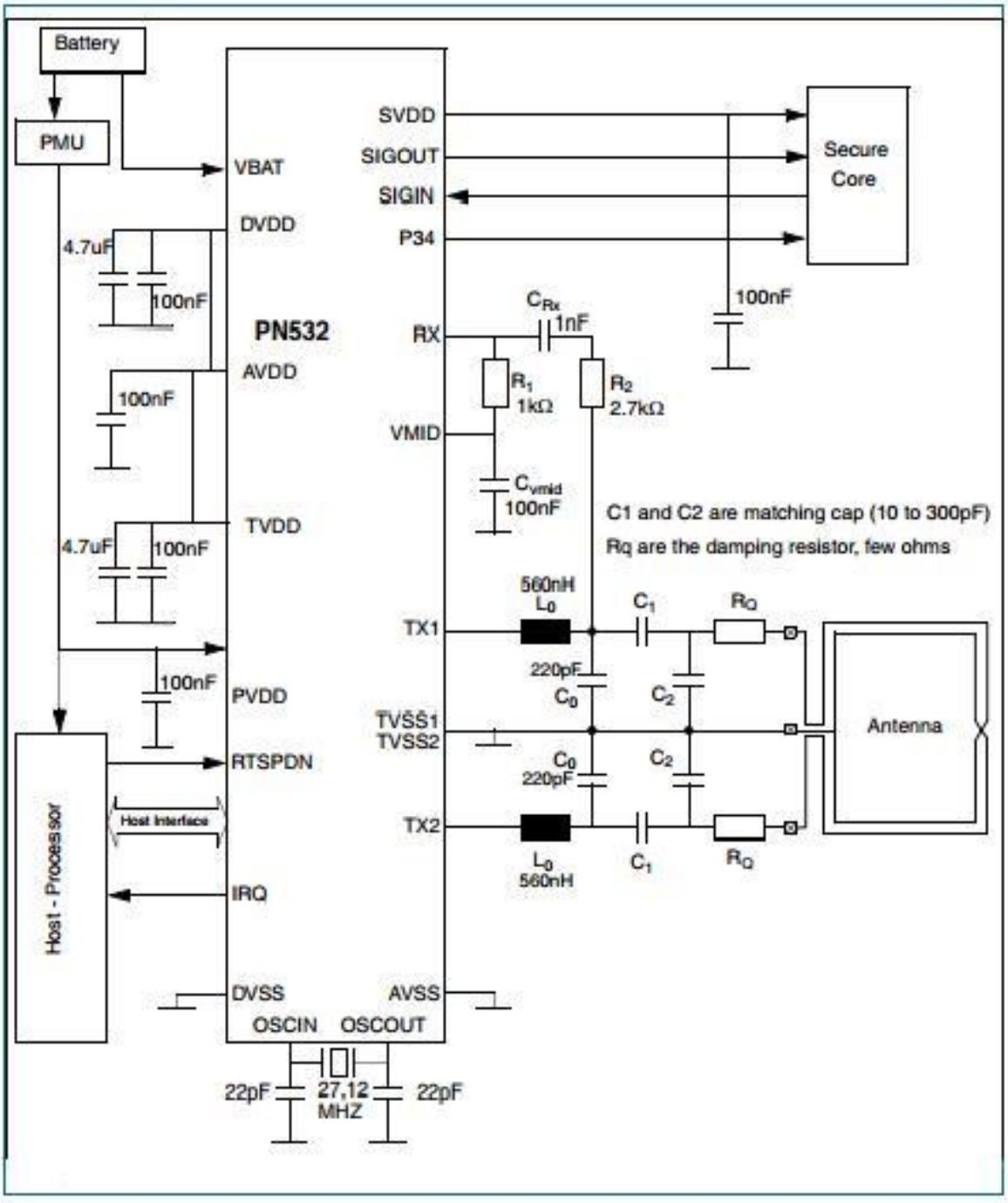


Figure 4.4.4.3 Reference schematic for the PN532 RFID Reader with the suggested component values. Credit to Phillips Semiconductors.

3.2) RELEVENT TECHNOLOGY

3.2.1) RFID

A RFID is a Radio-frequency identification that uses electromagnetic fields that will automatically identify and track tags that are attached to a certain object. The way this works is there is a two-way radio transmitter called interrogators and they send a signal to the tag and reads the response. There are two types of Radio-frequency identification which are as follows, a passive RFID don't contain a power supply and uses the nearby RFID radio waves. An active RFID contains a power source such as a battery and can operate from further ranges. These tags can store up to certain amount of data and have a microchip and an antenna. When the data is stored within the RFID's microchip it is waiting to be read, from there the antenna receives an electromagnetic energy from the RFID's interrogators antenna and thus confirms the unique tag. If it is an active tag, then the internal battery or power will create an electromagnetic field that will be sent to the reader and the reader picks up the signal and interprets the data. Each tag contains a set of unique binary numbers and it represents a certain number of bits depending on the model type that is chosen. Another important aspect of radio-frequency identification depends on if the chip is if there are read/write, read-only, or write once, read multiple times. Read/write RFID chips have a preprogrammed serial number that can't be written over, but obtain additional blocks of data that used to store information. Read-only tags can only send data and communicate with the factory programmed identification number. Write once, read multiple time tags allow the user to reprogram the information stored on the tag for their desired application, but maybe limited if the tag is only able to be written to one time. Below shows the range how far they are able to communicate.

Type	Range	Read/Write
Passive	1cm-1m	Read only
Active	1-30m	Read only
Active	1-30m	Write once/Read multiple
Active	1-30m	Write multiple
Active	0-1m	Read only

Table 3.2.1.1 This table shows the different combination of passive and active radio-frequency identification tags and readers with their varying ranges and data limits.

Passive tags are more inexpensive as compared to active tags, but are limited to certain restrictions such as the read/write ability. Depending on the type of active tag that is purchased some send out special signals to the receiver so it can identify the transmission. The tag can also communicate with the reader if it sends out a transmission as well. With a passive reader it can only transmit the data that is stored on it when it is in close proximity to a reader than can trigger the antenna circuit that is built into it. Passive tags operate at different frequency levels depending on the tags that are purchase. Low frequency operates between 125-134 KHz and have a short read range that is about 1-10 centimeters. High frequency and Near-field communication (NFC) operates at 13.56 MHz and has a read range of about 1 centimeter up to 1 meter. Ultra-High Frequency operates between 865

MHz – 960 MHz and has a long read range which is about an average distance of 5-6 meters, but depending on the tag it could reach up to 30 meters. With an active tag they operate at 433 MHz and 915 MHz and have a longer read range than a passive RFID. There is also radio-frequency identification that integrate both passive and active tags. The reader transmits a signal to active the passive tag of the RFID, then the tag gathers the signal by using the active portion integrated and transmit the unique identification it contains back to the reader. But this type of tag has a lower range and requires more energy from the reader to transmit to active the passive tag. The standardization of RFID's is set by multiple organizations such as the EPCglobal, the International Electrotechnical Commission (IEC), and the International Standards Organization (ISO). The standards are set limiting the operating frequency of the designed RFID and their intended functions.

For the purpose of this project we will utilizing a radio-frequency identification that contains an active chip. We will be implementing the MFRC522 to read the RFID that operates at 13.56 MHz. The operating distance is up to 50 millimeters. The reader supports ISO/IEC 14443 A/MIFARE and NTAG standards. It can host multiple interfaces such as Serial Peripheral Interface (SPI), Serial UART, and I^2C-bus interface. This module is ideal due to the fact of its low-operating current which is between 13-26 mA. Its idle current is 10-13mA when it is not in use. When it's asleep its less than 80 μA and it reaches a max current output of 30 mA. The frequency that the module support is 13.56 MHz. The operating frequency of this card classifies it as a High frequency and Near-field communication (NFC). The card type that will be used with the reader will be a mifare1 S50 keychain. This card contains a small RFID chip that has a modulator/demodulator, rectifier, clock generator, power on reset and a voltage regulator and an antenna, and will be powered by the reader when it is in close proximity. This card can be written and store up to 1 kilobyte of data in writable EEPROM, which are divided into 16 sectors with 4 blocks and 16 bytes in each block. The last block of each of the sectors contain a trailer that has two secret keys, which have access conditions for each block. It can also handle over 100,000 rewrites. There is a 4-byte unique identification that is already preprogrammed into the chip for authenticity. This is important because the user will be able to disarm and arm their alarm system with this card. The tag is light weight and won't be an inconvenience to the user because it only weighs 4.3 grams. The range of the tag is approximately 100 millimeters. This is ideal because the user must be close to the system to arm or disarm. The card can be read in less than 100 milliseconds. It has a data transfer rate of 106 kbits/s and has anti-collision which means several cards can be in the same proximity and can be distinguished by the unique identification that it is programmed with and will be selected depending on the transaction that is taking place. This RFID has a data retention of 10 years. The security embedded in this device consists of a three pass authentication based on (ISO/IEC DIS 9798-2) standard and after successful authentication the memory operations are encrypted. Values for the RFID are stored in a special format and can be increased or decreased.

Advantages and disadvantages

The advantages for passive RFID's are they last longer, it doesn't depend on an internal power source, the tags are inexpensive and are small, and the tags are more resistant to physical damage or unjust environments. They can operate in situation, temperature not being a factor. It allows for real-time updates. The disadvantages are the

communication depends on the antenna size and the shape, the range is limited and the user must be in close proximity to the receiver. It can't be read through metal or liquid. Anyone with a basic RFID scanner can be able to access the information from the tag. There is a possibility that the RFID can be hacked and a virus infection is probable. It is very to clone an RFID once the person intercepts the information needed. The advantages of active RFID's are they can be read from long distances; they have a larger memory capability to be able to store more information. The disadvantages of active RFID's are the tags are costly, the tags are larger in size, and when the battery dies the tag must be replaced.

3.2.2) WI-FI

Wi-Fi enables certain electronic devices that are capable to connect to a wireless local area network. Wi-Fi uses two frequency ranges that it broadcast in and they are as follows, the 2.4 gigahertz which is part of the ultra-high frequency range and 5 gigahertzes which part of the super high frequency range. Institute of Electrical and Electronics Engineers' (IEEE) sets the 802.11 standards. The standard is a set of media access control (MAC) and the physical layers and the foundation of how to implement them in a wireless local area network. The MAC sublayer provides addressing and channel access which makes it possible multiple nodes to communicate within multiple access. Wi-Fi has various encryption types such as WEP, WPA, and WPA2. Wired Equivalent Privacy (WEP) is a security encryption for IEEE 802.11. It uses a special algorithm that turns your password into a complex hexadecimal string. WEP is not an ideal because it is not difficult for an intruder to join your network. Using it will also reduce the performance output of your router. WPA was a much stronger encryption compared to WEP. It employed a Temporal Key Integrity Protocol (TKIP) in which it has a per packet key. This means that it generates a new 128-bit key for each packed produced and prevents certain attacks that existed in WEP. Next came WPA2 which is the most secure out of the 3. It implemented Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP) in which it ensured only authorized personnel could access the information. It has access control with layer management and provides the user with authentication that is unique and genuine. Wi-Fi has different bands such as the g/n/ac which are capable of faster speeds as you move up. The 2.4 gigahertz utilizes the channels 1-11. The 5 gigahertz band utilizes the channels 36-48. Wi-Fi works by allowing a device to transmit and receive information through radio waves. An access point such as a router is the device that propagates the wireless signals for devices to obtain the transmission. The device must be equipped with a wireless network interface controller to be able to connect to an access point. The signal is then transmitted to a wireless router by an antenna. The router must decode the incoming packets and sends and receives data that it obtains from the internet or any other local area network. Once the data is received it will be able to transmit that data back to the device that is requesting it.

The Wi-Fi alliance was formed in 1999 in which it organized multiple member organizations that establishes and enforces the standards for devices to be able to communicate with one another and have backwards compatibility of devices that are Wi-Fi enabled. Wi-Fi devices have great advantages over technology such as Bluetooth due to the fact they can transmit further, but that also comes at a cost of increased power. At first the 802.11a band could transfer up to 54 megabits per second in the 5GHz band it uses orthogonal frequency-division multiplexing. In which it splits the radio signal into multiple

sub signals before it reaches the destination which helps reduce inference. 802.11b was introduced later because of the low cost it was very popular. It works on the 2.4 GHz frequency band and can handle up to 11 megabits per second and uses complementary code keying to help increase speeds. Then came the 802.11g which could transfer 54 megabits per second in the 2.4GHz band it uses orthogonal frequency-division which is the same as 802.11a. The 802.11n build upon the previous bands and there was an added transmitter and receiver antennas which allowed for greater speeds it is backward compatible with all the previous bands. It can achieve speeds of about 140 megabits per second and transmit up to four different streams of data. Then came the 802.11ac which builds from previous standards, and is backwards compatible as well. It works on the 5 GHz frequency band and it is prone to less inference as oppose to its predecessors. It reaches a maximum of 450 megabits per second, it allows for up to multiple of 8 streams and has a very high throughput. Present day Wi-Fi routers are relatively inexpensive and will have 802.11 g/n for interoperability which have a transfer rate of 300-900 megabits per second depending on the router. The encryption has also improved from WEP to WPA2 which is a lot stronger security to make sure your network is secured from intruders.

For the purpose of this project Wi-Fi will be considered an option for the use of communication between the different sensors to the raspberry pi. The use of Wi-Fi is ideal because of the security features it has to offer. WPA2 is highly encrypted so the user would not have to worry about an intruder gaining access to their system. As long as there is an access point in the user environment the sensors will be able to communicate with the raspberry pi via an added wireless network card. Using Wi-Fi for communication would be beneficial because of the long range. It will be able to cover a larger area for the intended environment and improved protection. The Edimax EW-7811Un will be added to the Raspberry Pi for Wi-Fi capability. It implements IEEE standards of 802.11b, 802.11g, 802.11n and has a transfer speed up to 150 megabits per second. It operates at a frequency between 2.4-2.48 GHz. Using the 2.4 gigahertz network is important because it has a longer range and can penetrate certain infrastructures. Depending on where the sensors are located this is important because it can cover a larger area of the intended environment. It consists of WEP 64/128, WPA, and WPA2. WPA2 is the ideal security feature that will be used because of the high security and it is the less vulnerable out of the three.

Advantages and Disadvantages

Advantages of the 2.4 gigahertz network has a longer range and it supports older devices. It is better suited to transmit data over a long range as it can penetrate walls and other solid objects. The disadvantages are that there are few channel options and only three of them are non-overlapping, it also deals with more interference from microwaves and cordless phones which decreases the speed of the wireless network. It is also very congested and that can result in dropped connections or slow data throughput. The advantages of the 5 gigahertz network are it has 23 non-overlapping channels and there is far less interference so the speed of the wireless network should not be compromised. It can transmit higher amounts of data and it is less congested. The disadvantages of the 5 gigahertz network are not all technology is capable of handling that network and it has a shorter range. The advantages of Wi-Fi are users are able to access any network resources as long as they are in a primary environment that provides it, it is capable of handling a lot of clients at one time. The disadvantages of the 5 gigahertz network are that the availability

is not everywhere and the range is sufficient enough for a home, but not sufficient enough for larger areas depending on coverage and the infrastructure sometimes signals are lost due to walls and other components. Not many devices have the capability of supporting 5 gigahertz frequency band because it is a lot costlier to implement.

3.2.3) BLUETOOTH

This method rapidly switches a carrier from the vast amount of frequency channels available and uses a pseudorandom sequence that both the transmitter and receiver know. A spread spectrum has many advantages such as they are resistant to narrowband interference. They are difficult to intercept because it would be difficult for an intruder to intercept the transmission if the pseudorandom sequence is not known. The spread spectrum adds minimal noise to the narrow frequency communication, which in turn makes the bandwidth more efficient. microchips. Bluetooth Depending on the type of Bluetooth technology implemented there are various classes that are available and with those classes they have different power bands and range permitted with them. Since the device uses a radio broadcast communication systems, they don't need to be in the line of sight of each other, but need a quasi-optical wireless path. A class 1 Bluetooth integrated chip outputs a maximum of 100 megawatts and an of about 100 meters, class 2 outputs 2.5 megawatts, but has a range of 10 meters. For the purpose of this project class 1 would be ideal because of the range that it covers. The Bluetooth special interest group handles the standardization and helps ensure interoperability. The Bluetooth protocol stack contains the architecture of the core protocols, cable replacement protocols, telephony control protocols, and adopted protocols. All Bluetooth stacks must contain LMP, L2CAP, and SDP. Link Management Protocol (LMP) is integrated and used for the set-up and control of the radio link between the two devices. The Logical Link Control and Adaption Protocol (L2CAP) has multiple logical connections between two devices, but uses different higher level protocols to segment and reassemble the on air packets. The maximum transmission unit that needs to be supported is 48 bytes, but 672 bytes as default. L2CAP has two modes retransmission and flow control and they can be configured in either isochronous data or reliable data per channel. The Service Discovery Protocol (SDP) it allows a device to detect if there are any devices in the perimeter that can work with it. The services use a Universally Unique Identifier (UUID) to communicate with each other to be able to connect with one another. Radio Frequency Communications (RFCOMM) is a cable replacement protocol that generates a virtual serial data stream. Bluetooth Network Encapsulation Protocol (BNEP) transfers one protocol stack to another using the L2CAP channel. It transmits IP packets in a personal area networking profile. Adopted protocols are implemented by other organizations that set the standards for Bluetooth. Point-to-point (PPP) is an internet standard protocol that is associated with this and it is used for transferring IP datagrams which is a packet-switched network and transmit data over a point-to-point link. Object exchange protocol (OBEX) which provides an exchange of objects. Wireless Application Environment/Wireless Application (WAE/WAP) WAE is a framework for wireless devices and WAP is an open standard for information services. Bluetooth has many versions and depending on which version is implemented the maximum speed and range varies as well. Standard devices right now have version 4.0+ implemented and it has a transfer rate of about 25 megabits per second and a range of 200 feet. Bluetooth v4.0 offers classic Bluetooth, Bluetooth high speed and Bluetooth low energy protocols. Bluetooth high speed foundation is based on Wi-Fi. And Classic Bluetooth consists of legacy

Bluetooth protocols. Bluetooth low energy has a different protocol stack from the ones mentioned above. There are two different modes that are implemented, single-mode has only a low energy protocol stack. In dual mode, Bluetooth smart functionality is integrated with Classic Bluetooth controller.

3.2.4) LED

A light-emitting diode (LED) is a two-lead semiconductor light source in which it contains a p-n junction, when it is activated it emits a light. P-N junctions are able to convert absorbed light energy into an electric current. This effect emits light when there is some type of electrical energy being applied. When the electrons travel from the N-region and recombine with the existing holes in the P-region that creates some type of dissipation which emits energy in the form of heat and light this effect is called electroluminescence. The color of light is determined by the band gap of the semiconductor. Light-emitting diodes are often small and can be measured to be less than $1mm^2$.

Blue LEDs were created using gallium nitride on a sapphire substrate. The discovery of high powered blue LEDs led to the development of white LEDs. White LEDs can now produce over 300 lumens per watt of electricity and can last up to 100,000 hours. The LED has a chip embedded that contains semiconducting material used for doping with certain impurities to create a special junction known as the p-n junction. Current flows from either the anode side to the cathode. When electron is flowing and it meets a hole then it will fall into a lower energy and releases energy in the form of light. Depending on the color the LED releases it will fall into a specific wavelength of light that determines its color which depends on the band gap energy of the materials that it contains in the p-n junction. Typical LEDs are designed to operate between 30-60 mW. They are low power consumption technology. Light emitting diodes are known for their high luminous efficacy. There have been LEDs that have been created that can consume 24 mW at 20 mA. When LEDs operate at higher electric currents this creates higher heat levels which will have an effect on the LEDs life span. High brightness light-emitting diodes have an operating standard at 350 mA.

LED COLOR	Standard Brightness			High Brightness		
	Ipk (NM)	Iv (mcd)	Viewing Angle	Ipk (NM)	Iv (mcd)	Viewing Angle
RED	635	120	35	635	900	30
ORANGE	605	90	30	609	1300	30
AMBER	583	100	35	592	1300	30
YELLOW	570	160	30	--	--	--
GREEN	565	140	24	520	1200	45

Table 3.2.4.1 This table compares the standard and high brightness LEDs and shows what is the most optimal viewing angles and brightness depending on the color.

We use the table above to make the decision on what color and type of LED would be the most ideal for this project. For the purpose of this project we will be implementing a red and green light-emitting diode. The red LED represents when the system will be armed or if there is an intrusion taken place. The green will represent that the system is disarmed and is stable. Although the LED power consumption is very low we will still have to take it into consideration for the calculations because the microcontroller we are using has some restraints on how much power it is able to output without being overloaded.

Advantages and Disadvantages

The advantages of using light-emitting diodes are their long lifespan which could be up to a 100,000 hours of visible light being generated. Light-emitting diodes are very low power device. They can operate in a wide range of temperatures from -40 to 185 degrees Fahrenheit and with the humidity below 65%. They can light up very quickly and can achieve full brightness within microseconds. The disadvantages of using light-emitting diodes are it has a short illumination range. The heat generated by the LED must be absorbed to prevent over-heating or damaging to the chip. Light-emitting diode is ideal for this project because it will notify the user of when the system is armed or disarmed.

3.2.5) LCD

A liquid-crystal display (LCD) is an electronic visual display that uses the light modulating properties of liquid crystals. LCD's are able to display arbitrary images or fixed images with low information content. They are made up of large number of small pixels which gives them the ability to display the images. Each pixel of the LCD consists of a layer of molecules that are between two transparent electrodes and two polarizing filters. There are different types of LCD panels one being the cold cathode fluorescent lamp (CCFL) which is what is placed on opposite sides of the display and a diffuser sends light out evenly to produce an image. Another panel is the EL-WLED which uses white light-emitting diodes around the edge of the screen, this in turns uses a light diffuser to spread the light across the screen to produce the image. A WLED array is an LCD that contains a full array of white LED's placed behind a diffuser which then placed behind a panel. This display gives the LED's the ability to dim the display in darker areas thus increasing the contrast ratio. The last panel is an RGB-LED in which it has a full array of LED's, but the LED's are RGB. This display increases a wider color of gamut's.

When it comes to liquid-crystal displays there are two types, passive matrix and active matrix. With passive matrix it uses super-twisted nematic which require less power and are cheaper to manufacture. Active-matrix technology which is widely used today has multiple types of displays. Twisted nematic displays contain liquid crystals that are twisting and untwisting allowing a certain amount of light through to display the image. In-plane switching (IPS) aligns the liquid crystals in a plane. There is an electric field that is applied across from the glass which helps the liquid crystals switch and produce an image.

When deciding on the type of liquid-crystal display to purchase there are a few key factors to consider. Resolution of the LCD and the viewing range of it. Determining the size of LCD that is suitable for intended use. The timing performance of the LCD depending on the pixel response of how quickly the pixel can change its brightness from one level to another. The refresh rate of the LCD which determines how quickly the display can refresh the data it is displaying for the most accurate real time information. The

brightness and contrast ratio of the display to see how the performance would be in a dark room versus a fully lit room.

Advantages and Disadvantages

The advantages of liquid-crystal displays are they are very compact and light and have a low power consumptions depending on the panel you choose. There is very little heat that is emitted. There is no refresh rate flicker because they normally refresh at 200Hz or more. LCD has many native display ports such as DVI or HDMI. The disadvantages of LCD are there might be some uneven backlighting or limited viewing angle. There is a fixed bit depth and can only display a certain amount of colors. There is a possibility of dead or stuck pixels.

3.2.6) MOTION SENSOR

A motion sensor is a device that detects moving objects, normally it is looking for an individual. It is normally integrated with security systems; it monitors any type of motion that is in the general area that it is observing. There are different types of motion detectors available on the market and may contain different components depending on the consumer's preference. For instance, an electronic motion sensor contains optical, microwave, or acoustic sensors and possibly a transmitter for illumination. But passive sensors are only capable of sensing objects that are moving.

In the industry today there are several sensor technologies being used. The passive infrared sensor (PIR) checks for the skin temperature of the moving object. This is done by detecting black body radiation at a mid-infrared wavelength. There is no energy that is being radiated from the sensor. The microwave technology inside of the motion sensor detects a moving object by sending out a continuous wave of microwave radiation. Thus the detector checks for any phase shift that is either moving towards or away from the sensor itself. Ultrasonic integrated sensors send out an ultrasonic wave and works similar to the microwave sensors except it is at a higher frequency. There is one fault to using this type of sensor because it is very sensitive to movement in areas that it isn't supposed to cover and can trigger the sensor. Tomographic motion sensors have a wide range of operations because they can detect through walls or any type of obstruction. It uses radio waves to detect any motion as they pass from node to node of a mesh network.

We had a couple different type of motion sensor modules that we had in mind to implement in our project. The first of many was the active infrared motion sensor, or shortly put AIR. The way an AIR works is it transmits an infrared signal to a receiver which will continuously be transmitted until the signal is blocked by a physical object. When this occurs the alarm system will sound. Because the way the transmitter and receiver are set up, they are less likely to give false positive readings because the two components are stationary. The downside of these types of motion sensors is that they are easily detected. A simple camera phone will enable you to see where the infrared beams are coming from. This would not be an efficient design for us since we are trying to make our security system have the least amount of possible ways to be breached. Assuming an intruder discovered our AIR motion sensor, then they could transmit a beam to the receiver. That way the security system will always be receiving an uninterrupted signal from the sources. Because an AIR motion sensor can be easily bypassed, then we will not be including it in our design.

Even though the device doesn't give a lot of false positives, it appears to be too easily manipulated by intruders if they were to discover the transmitter's location.

The next type of motion sensor that we considered implementing in our design was an optic based motion sensor. How this type of motion sensor operates is by recording footage and then checking over and over again when the image changes. In our opinion, this seems like it could be a very tedious process. We would need a lot of storage just for the stored video. Additionally, we feel like the coding would be extremely complex to implement. In order for us to make it work we would have to write a code that will literally check each pixel of the frame and compare it to the last. This would be a very tedious process for not only us, but for our microcontroller. We feel like the amount of processing power that this would take for just a motion sensor would be unnecessary. Not only unnecessary but it would be inefficient. We are trying to make our security system use as little power as possible, therefore this device would be straining our power limits for a simple sensor.

A very basic type of motion sensor is actually a mechanical switch. An excellent example we can use here as an analogy is a trip wire. The trip wire will always be set, until something crosses it. This is comparable to the AIR motion sensor. They both are dependent on a signal the will set the alarm when the signal is tripped. Another example of a mechanical motion sensors are pressure plates. These work the same as switches. As soon as someone steps on the pressure plate, then it will complete the circuit, thus sounding off the alarm. The advantage of this design is that it's extremely cheap and easy to design. The downside to this device is that an intruder could step over the pressure plate. That is also the same disadvantage as the trip wire. If an intruder were to notice these two components, then they could easily just step over it and then our whole security system would be compromised. The lack of coverage that the mechanical motion detectors provide for our security system is the reason why we will not be including it into our design. The only way for a mechanical motion sensor to be one hundred percent effective is by designing a system that will literally cover every square inch of the interior of the house. This would not only be tedious, but also a waste of time.

Another type of motion sensor that we have researched is an ultrasonic sensor. The way this works is it sends out a very high frequency sound signal that will emit from the speaker, travel until it hits an object, and then bounces back into the receiver. Based off how long the signal takes to transmit and receive is how far away the object is from the sensor. This device is originally used for tracking the distance of an object. We can easily implement this technology into our design to detect when there is motion. A summation of how it works is by detecting when the range of the object that it points at changes. Once it does, this means motion is being detected and this will enable the system to sound the alarm.

Finally, the Passive Infrared motion sensor will be the device we are introducing to our design. Our research has concluded that they are the most commonly used motion sensors in the industry today. A complication with this type of motion sensor is that they are prone to false positives because of environmental changes. Additionally, they can be bypassed by an intruder by simply wearing a heat insulated full body suit. This will render the device useless because it will not be able to detect any body heat. We feel like this is a

low risk to take since the everyday common criminal is not going to have a great deal of knowledge on motion sensors to even consider wearing this type of suit.

Since we are going to implement a device that uses a PIR motion sensor, then we need to figure out which device we will be using in our design that will meet our goals and requirements for this project. After extensive research, we have concluded that the HC-SR501 motion sensor will be most suitable for this project. This module has three connecting pins which are Vin, ground, and out. It also contains a pyroelectric sensor that will generate energy when heat is detected. This means that it will basically turn on when someone walks in front of the sensor, thus making it extremely power efficient. The module also has a cover implemented into the design that will sense energy over a 120-degree range.

The device includes two potentiometers. One will enable us to adjust the sensitivity of the sensor itself, and the other potentiometer will adjust the delay time. The sensitivity adjustment nob will let us set the distance of which the sensor can detect a moving object that gives off heat. The maximum range for this sensor is around twenty-two feet. The minimum distance it can detect is around nine feet. Because the minimum distance is at such a long range, we will need to determine an optimum placement for this device such that an intruder will not be able to slip by undetected.

Additionally, this device contains two different types of trigger modes, non-repeatable mode, and repeatable mode. The non-repeatable mode works by making the output pin high when an object is detected and then switching back to a logical low value when the delay timer is over. The other mode will keep the output level high at all times until an object is detected which it will then turn the output logic level to low. To help us better visually understand the module, a schematic diagram of this device connected to the Raspberry Pi will be included in chapter four section 4.1.

Because we live in Florida and it is naturally hot, the distance at which it will detect is slightly shorter since the ambient temperature will be around the same level as the human body. To compensate for this, we will be only detecting motion on the inside where the ambient temperature is considerably lower than the human body temperature. Additionally, this type of sensor is sensitive to light. Because of that, the sensor will be operating at its best when the sun is down or in a room that has little light pollution. Finally, the sensor module detects movement from left to right, or right to left and not top to bottom or bottom to top. This means we need to install the module parallel to the ground to ensure an accurate detection. The failure to detect movement from top to bottom shouldn't be too much of an issue because of natural human movement vector which is parallel to the ground.

Advantages and Disadvantages

The advantages of using motion detectors are if it is connected to an alarm system and the system is unaware of a breach from a window or door then most likely the motion detector will sense the intrusion. They are a key component to an alarm system due to the fact that it covers a wide range depending on the sensor in use. The disadvantage of using motion sensors are the possibility of the sensor being triggered by a small pet or some other movement which would make it inaccurate.

3.2.7) TILT SENSOR

A tilt sensor is a component that is capable of detecting if an object is being tilted based on two axes for reference. There two types of tilt switches. Mercury switches which aren't used in practice as often because of the hazardous material, uses a drop of mercury in a tube and when the orientation the sensor is changed a bead rolls down the tube, which causes the contacts to touch thus turning off the switch. The ball-in-cage switch houses a metal ball with two or more contacts. When the switch is tilted at a certain orientation the ball shorts the two electrical contacts thus turning off the switch.

Advantages and disadvantages

The biggest advantage of tilt sensors is their multiple purpose use. You can use tilt for a variety of things; for example, to:

Measure slope angle of satellite
antenna Monitor boom angle of cranes
Estimate height of feature that uses vertical distance like tree, building
etc Measure steepness of a ski slope
Use a warning system on external surface to indicate when tilt is too much
Measure alignment of 2-dimensional plane tilt angles
Indicate pitch and rolls of aircraft or vehicles
Measure look angle of satellite antenna towards a satellite
Measure the angle of drilling

The only big disadvantage of tilt sensors is that some types may be too expensive.

3.2.8) SPI COMMUNICATION

The Serial Peripheral Interface (SPI) was created by Motorola as a means of communication between devices. It sends data to microcontrollers and small devices such as shift registers, sensors, and SD cards. It uses a separate clock and data lines and has select lines to choose which device to send the data to. It utilizes a master-slave architecture in which there is a single master that is connected to multiple slave devices. It has a four wire serial bus connection and specifies four logic signals which are as follows:

SCLK: Serial Clock (output from master).
MOSI: Master Output, Slave Input (output from master).
MISO: Master Input, Slave Output (output from slave).
SS: Slave Select (active low, output from master)

SCLK will control the data flow, MOSI will send the data from the master to the slave, MISO sends data from the slave to the master, and SS is active low and it is used to select which slave device is going to be selected. For SPI to communicate the master device must configure the clock to be compatible with the slave device that is desired. The master device will send a 0 to the slave select to make the device activate. Once the slave is activated, the master will be able to send data by toggling SCLK so that data will over the MOSI and MISO lines. SPI utilizes full duplexing in which it is able to allow the master and slave devices to send and receive data simultaneously. The master will continue to toggle SCLK until all of the data is transmitted and once it is done it will deselect the slave device.

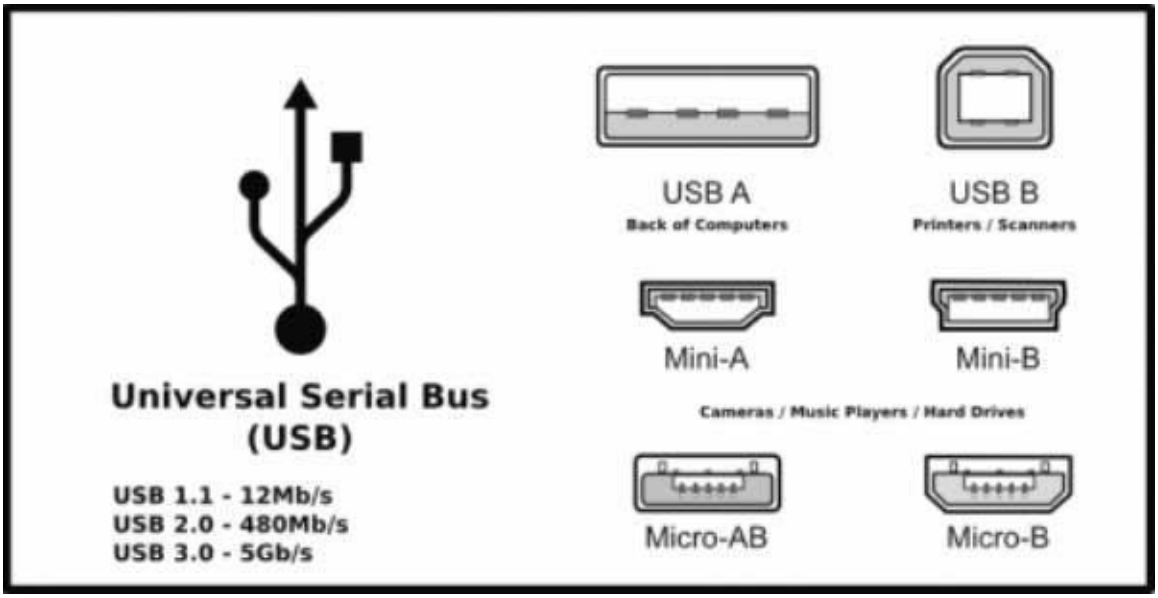
Advantages and Disadvantages

The Advantages of using Serial Peripheral Interface (SPI) is it has full duplex communication in which the data can be sent and received simultaneously. It does not have a fix limit and the message size can be arbitrary. Slaves user the master's clock so there is no need for precision oscillators and transceivers are not needed. The disadvantage of using Serial Peripheral Interface (SPI) is it requires more pins on the microcontroller. It only can only support one master device and there is no error-checking available. There is no hardware slave acknowledgement.

3.2.9) USB

USB in general is the modern day standard for data transfer between all types of devices. Currently, no matter what kind of device your using such as a portable media player, network adapter, printers, mice or even a keyboard, theirs a high chance it contains this type of connection. It is short for Universal Serial Bus and was a type of connection to simplify transfer of data between devices. It wasn't solely created for the end user to have control of, however, in modern day applications it easily fits into this as one of most common uses. The USB has gone through many variations over the years, although, not much has changed in general when it comes to how it functions. Generally, the newer version of USBs tends to support a greater or higher speed data output and a greater
Amperage. Part of this isn't for the sole function of data outputs, but also to allow devices to charge quicker in general. The main types of USB variations that have been in use or currently are in use are the USB 1.0, 2.0, 3.0, 3.1 and a USB-C. Although, they all appear to be the same by observing them, they are greatly different. The industry standard for the first USB appeared around the 1990's and was the product of a multiple company effort and since then has been continued to been worked on. The main reason for its creation was to standardize the communication between devices as to many types of standards were being created. For the most part, most of these connection types functioned off the same exact concept but had different layouts that caused compatibility issues. Overall, there are a few type of end connectors layouts such as the Mini, Mini-B and Micro size which are still in use. However, the Micro connection is quickly becoming the most popular type in use. Originally, there was only a USB A type connection for both ends on devices. This was first used and adopted during USB 1.0. It was widely implemented into the first Legacy-free PCs. It wasn't until the introduction of USB 2.0 that the Mini-A and Mini-B came into creation and use. USB Type C was created around the time of USB 3.0 and was intended to be a universe plug that would last infinitely into the future. USB Type C unlike its other counterparts contains 24 pins and actually will pass different amounts of current depending on the configuration that it's plugged into. Below is a representation of these connections and their profile shown in image 3.2.9.1.

Image 3.2.9.1: Commonly used USB plug types.



Picture by itarticle.net [4]

The amount of transferrable data isn't limited by the configuration, but by the actual device sending the data through the cable. Each type of connection also has a USB device associated with it that will determine the actual speed that data can be transferred over. In essence, the cable type has little bearing on the actual efficiency but will be determined by the device sending and receiving the data. The limit of data transfer can be found below.

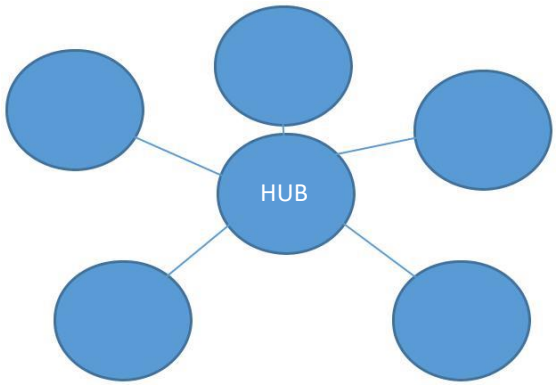
NAME	MAX DATA RATE	RELEASE DATE
USB 1.0	Between 1.5 Mbit/s and 12Mbit/s	January 1996
USB 2.0	480 Mbit/s	April 2000
USB 3.0	5 Gbit/s	November 2008
USB 3.1	10 Gbit/s	July 2013

One of the key differences between a USB and many other configuration of data busses is that the USB can send and receive data at the same time. Many other types of devices such as an Ethernet cable is actually incapable of this resulting in a higher download/upload speed but only one action at a time which introduces latency not found within a USB type device. Later on, the focus on USB not only became one of data transfer, but also the ability to power devices as needed. With newer and more power hungry devices, it led to a need for a higher ability to transfer power to devices. This led not only to the changing of the USB data transfer rate, but an evolution in terms of power available for devices. This meant that not all devices would charge at nearly the same speed with all types of USB. However, they all had to be designed with the idea that the consumer may plug in less than ideal types resulting in less power available. The result was most of the devices that utilize a proper

standard will also attempt to avoid damaging anything plugged in using the standard. The support for USB power delivery hasn't quite stayed nearly the same when it comes to standards and companies planning around their use. Some companies use standards based around disabling the data portion of the device and tying things such as resistors in their place and some ignored this all together. Also some devices are smart enough to accept wide ranges of acceptable currents and know how to self-limit the amount needed. The current standards when it comes to power charging without special company considerations are as follows.

NAME	DATE	POWER SPEC
USB BATTERY CHARGING 1.0	03/2007	5V, 1.5A
USB POWER DELIVERY REVISION 1.0 VERSION 1.0	07/2012	20V, 5A
USB TYPE-C 1.0	08/2014	5V, 3A
USB POWER DELIVERY REVISION 2.0 VERSION 1.0	08/2014	20V, 5A
USB TYPE-C 1.1	04/2015	5V, 3A
USB POWER DELIVERY REVISION 2.0 VERSION 1.1	05/2015	20V, 5A
USB POWER DELIVERY REVISION 2.0 VERSION 1.2	03/2016	20V, 5A

On the design side of an actual USB, it tends to be asymmetrical. The USB port itself uses a Star Topology in which many devices connect to a central hub. A Star Topology itself is layout common in many applications where an actual star pattern is formed all connected to the middle as follows.



This actually allows for great numbers of devices to not only be plugged into a USB hub, but also additional USB layers added on top. There is an upper limit of around 127 devices that can be supported. Each pin is actually a pipeline on the device. There is actually a much higher amount of supported pins, however, those applications are generally for specialty setups. USBs actually have class codes depending on use that effect how it treats each device and identified exactly what is being interpreted. This is one of the reasons that USBs allow for plug and play as a very common feature and why they are so commonly used among devices. Along with this hardware, there were a few other types of configurations introduced with USB hardware when it came to charging. USB charging became in itself its own specialty with the rise of newer devices. One of the major changes when it came to USB charging was the inability to actually transmit data in some of the configurations. This means USB charging have become their own special standards of USB with its own ideals known as a charging port. One thing to note is the power difference between them and tradition USB use varies greatly. One way in which they function different is they support a much higher Amperage and sometimes ignore the slower ramping up of Amperage. The upper limit when it comes to these devices tend to run around 5 Amps. However, the physical USB cables aren't nearly quality tested up to this limit in all cases as they tend to be tested only to 1.5A to meet current standards. One risk of this is the voltage difference across ends of cables may be far greater than normal while still needing to fall within specifications not to damage USB devices. One other charging standard is known as the PoweredUSB which adds 4 pins that support different voltages and Amperages that is common used.

3.2.10) BATTERY TECHNOLOGY

Within today's society batteries play a very big role in terms of types available on the market and types out there that can be utilized depending on need. Originally there was very little standardization within the market involving batteries. However, over the years this changed greatly and batteries standard to become very standardized. The basic premise for batteries in general is they are essentially a container with various chemicals and materials which can hold a charge and output electricity as needed to a system or application.

The types of composition on the inside of a battery help to shape the designation they fall into along with shape and stored voltage. One common designation for batteries currently on the market was that there are dry and wet cell batteries. One of these generally contained a liquid while the other weren't necessarily a fluid. The common designation in terms of shape falls into three types, cylindrical, rectangular and button cells. The current standards in terms of U.S applications fall under the ANSI C18 series of standards.

Originally, non-rechargeable batteries were the most traditional type available out there available on the market with the most familiar kinds of being A, AA, AA, C and D batteries being the standard types. However, rechargeable batteries are quickly becoming the common type in a host of nontraditional shapes.

Some of the common safety concerns when working with a battery system is that you shouldn't have jewelry on at all when working with a battery or any system as they tend to conduct and can pose a major risk to the user. Another major safety concern

according to research if dealing with a larger battery was to use safety goggles around certain types of batteries such as Sulfuric Acid batteries or Hydrogen Gas batteries as they pose extra risks from damage of being a risk to eyes and skin. On smaller batteries good practice should still be observed when dealing with batteries in general that contain a very high power such as disconnecting ground wires to make passing electricity less of a risk.

There are also two types of batteries that fall into the lead acid category which are starting battery and deep cycle batteries in most applications. The deep cycle one generally will pass an energy slower to a system as they aren't designed for that purpose and cannot be interchangeably used. However, Dual Purpose batteries are becoming far more common in use that meet both depending on system. Batteries such as Wet Cell, Gel Cell and Absorbed Glass Mat batteries fall under the type of batteries known as lead acid types. This is further divided into batteries that can be serviced or maintenance free. The only difference between these is in the design of allowing a user to maintain a battery or not in general. However, most consumer level batteries used in consumer applications will generally fall under maintenance free as most users will lack the required knowledge in order to deal with these type of batteries. Batteries also come with a few common times used to measure various stats such as capacity and such. The common terms found in standards for batteries given to rate them is CCA, CA, RC and AH. These terms stand for Cold cranking amps, cranking amps, reverse capacity and amp hour respectively. The CCA is a measurement of amps that a battery can deliver at 0 degrees F for 30 seconds while not dropping below a designated amperage. While the CA is measured at about 32 degrees F. CA replaced what used to be known as Marine Cranking Amps or MCA. RC represents an important stat for larger capacities known as Reverse Capacity in which tells the user how long a battery fully charged will discharge 25A until the battery drops below a 10.5V threshold. The most common measurement found with batteries denoting how long they will physically last is the Amp Hour or AH which is how long a battery lasts with deep cycles. This is the only measurement other than voltage most consumers will be familiar with in general.

Below is a list of terms to help make a summary of all these terms for convenience.

Table 3.12.10.1: Common Battery Terms

TERM	DESCRIPTION
CCA	**Amps a battery can output at 0 degrees F and not drop below a certain voltage.**
CA	**Amps measured at 32 degrees F and not drop below a certain voltage**
AH	**Amp hour or how long a battery will last under use. The most common term other than voltage for a battery.**
RC	**Number of minutes a battery will output 25 amps and drop below a certain voltage. A common term in larger batteries.**

When trying to test a battery the recommended method to be used is specific gravity coupled with the voltage of a battery. This is done by using a temperate compensating hydrometer and measure the D.C Voltmeter. A more convenient tool to accomplish this id

one using a load tester which is specifically made for this application. Other common tools to test a battery would be a quality load tester which will indicate if a battery has been punctured.

Another important note when it comes to batteries is the life performance of a battery. It is common for on average only 30% of batteries to reach a 48-month mark. The most common reason for batteries to reach end of life early is due to sulfur buildings up on plating of batteries. Common things that accelerate the rate of these battery failures are weather related, stored without being used, using wrong batteries, undercharging a battery, extreme temperate, air exposure and parasitic drain. Parasitic drain is the result of various systems that run even when an item isn't an operation. This is the what causes many batteries to empty far quicker than expected for the end user. A common example of this is various clocks and such within smart phones that must run even when not in use, although, low power mode helps to alleviate a lot of their use. Ideally, rechargeable batteries are to be kept at their maximum levels in order to maintain them for long term use. This is why in systems such in cars they tend to try and keep them as charged as possible.

Below is a list of the most common types still of batteries currently in use and their capacities along with commonly found phone types. These generally fall under dry cell batteries used for portable or personal applications.

Table 3.12.10.2: Common Household Battery and Phone Batteries

TYPE	CAPACITY (MAH)	VOLTAGE (V)
AA (NON-RECHARGEABLE)	2700	1.5
AAA (NON-RECHARGEABLE)	1200	1.5
C (NON-RECHARGEABLE)	8000	1.5
D (NON-RECHARGEABLE)	12000	1.5
9V	565	9
GALAXY S5	2800	3.85
GALAXY S6	2550	3.85
IPHONE 5	1507	3.8
IPHONE 6	1915	3.82

3.2.11) MICROCONTROLLER

The Raspberry Pi 3 model B is the latest of the Raspberry Pi brand, it came out earlier this year to replace the Raspberry Pi 2 model B. The Raspberry Pi 3 model B comes with a lot of great features that the previous version was lacking of in addition the features that were already on the Raspberry Pi 2 model B. The Raspberry Pi 3 model B comes with a 1.2 GHz 64-bit quad-core ARMv8 CPU, 802.11n Wireless LAN, Bluetooth 4.1 and BLE (Bluetooth Low Energy) in addition to all the features that were in the Raspberry Pi 2 model B. The Raspberry Pi 3 model B is compatible with the both the Raspberry Pi 1 and the Raspberry Pi 2; it has the same form factor as the previous version of the Raspberry Pi. The Raspberry Pi 3 model B separates itself from the other competitors when we compare the benchmarks of some the other microcontroller out there we can clearly notice that there's a great difference that make the Pi 3 model B is by far the best out there right now. When comparing the Sysbench CPU, the Pi 3 has the lowest between all the Raspberry Pi's version. This table gives an idea of the Sysbench CPU:

Raspberry (version)	Single-Threaded	Multi-Threaded
Model B+	510.8 secs	N/A
Model A+	502.4 secs	N/A
Zero	349.4 secs	N/A
Pi 2	293.1 secs	76.3 secs
Pi 3	182.2 secs	49 secs

SysBench table.

When it comes to SysBench CPU the lower the better; by far the Pi 3 has the fastest Sysbench between all the Raspberry Pi's version. Like the Pi 2 it offers support for multi-threaded operations. The SysBench shows that the Raspberry Pi's brand came far; from single-threaded to multi-threaded as well as the single-threaded design being greatly improved in performance.

The previous Raspberry Pi version, except for the Pi 2 which uses four USB ports, use a simple connection where you have to connect everything physically to it because they only featured a pair of USB 2.0 inputs. The Raspberry Pi 3 not only take care of this issue but takes it to another level; it is Bluetooth compatible which is very needed at this day and age. It allows users to connect peripherals wirelessly. The SoC boost the performance of the Raspberry Pi 3 over the Pi 2 which was more performant than the Pi 1. Even though both the Pi 2 and the Pi 3 use quad-core processors, the Pi 3 has a quartet of Cortex-A53 CPU that is clocked at 1.2 GHz where the Pi 2 features a ARM Cortex-A7 clocked at 900 MHz and also the graphics processor of the Pi 3 is faster than the Pi 2 with 400 MHz for the Pi 3 compare to 250 MHz for the Pi 2.

The Raspberry Pi's family or brand uses GPIO pins that are generally used with python which carry to a bottlenecked CPU. When we take into account the Python GPIO where

the higher the frequency the better the CPU works, the Pi 3 almost double the frequency of the closest competitor which the Pi 2 as we can see in the table below:

Raspberry (version)	Python GPIO
Model B+	59.7 kHz
Model A+	62.9 kHz
Zero	88.5 kHz
Pi 2	197.2 kHz
Pi 3	344.4 kHz

Python GPIO table.

The Raspberry Pi 3 solves the issue of pausing and lagging when performing simple task like browsing the internet or managing documents that existed in the previous versions of the Raspberry Pi brand. The Raspberry Pi 3 has improved about 50- 60 percent from the Previous version; it becomes more reliable to use. Both the Pi 2 and the Pi 3 were tested using Sunspider JavaScript benchmark where the lower the number the better the performance; the Raspberry Pi 3 took 2873.8 millisecond to complete the benchmark and The Raspberry Pi 2 took 4614.8 millisecond to complete the benchmark.

Another aspect of the benchmarking that was tested for the Raspberry brand was the Whetstone. The Whetstone was developed by B. A. Wichman in the 1970s; it was developed to actually measure a computer's speed. It focuses on floating-point performance. Even though it's an old method, it offers a pretty good look into the peak floating-point of a processor. Also they use the Dhrystone to measure integer or whole number performance. The Dhrystone was developed by Reinhold P. Weicker in the 1980s; they are very capable of comparing the performance of different chips. For both Whetstone and the Dhrystone, the higher the number the better the performance. In both cases we notice that the Raspberry Pi 3 almost double the closest competitor which is the Pi 2. The Whetstone is measured in MWIPS (Million Whetstone Instruction Per Second) and the Dhrystone in MIPS (Million Instruction Per Second) Below is table that gives us a glance at Whetstone and the Dhrystone:

Raspberry (version)	Whetstone	Dhrystone
Model B+	232.6 MWIPS	847.1 MIPS
Model A+	236.9 MWIPS	863.2 MIPS
Zero	340.5 MWIPS	1237.3 MIPS
Pi 2	437.2 MWIPS	1671.6 MIPS

Pi 3	711.4 MWIPS	2458.1 MIPS

Whetstone, Dhrystone table

Generally, improvement in performance comes with sacrifices. When comparing the Raspberry Pi 3 with the other version it consumes the most power; but that is what you get for having the best performance in the family; that extra performance you get means that it spends quite some times in idle mode. The Pi Model A+ has the maximum battery life followed by the Pi Zero as we can see in the table below:

Raspberry (version)	Power Draw (Idle)	Power Draw (Load)
Model B+	0.25	0.31
Model A+	0.11	0.17
Zero	0.1	0.25
Pi 2	0.26	0.42
Pi 3	0.31	0.58

Power Draw table.

In conclusion, the Raspberry Pi 3 has proven to be the best out there within its family; except for the power consumption it eclipses all the other Raspberry Pi versions. With its performance it is by far the best in its class; the closest competitor which is the Raspberry Pi 2 is far behind to be even considered as an alternative.

3.2.12) INTERRUPTS AND POLLIN

To get required information from an external system or device, the processor uses a method called interrupt.

An interrupt is a signal sent to the processor by either hardware or software letting the processor know that an event requires immediate attention. An interrupt signals the processor that it needs to stop executing the current program and allow it to execute a special code. For example, an Ethernet device driver would interrupt whenever it receives an Ethernet packet from the network. The Linux kernel needs to be able to deliver the interrupt from the hardware device to the correct device driver. This is achieved by the device driver registering its usage of the interrupt with the kernel. It registers the address of an interrupt handling routine and the interrupt number that it wishes to own. A Common example is pressing on the key on the keyboard, which causes to the keyboard to send interrupt to the microcontroller to read the information of the pressed key.

The program which is associated with the interrupt is called the interrupt service routine (ISR) or interrupt handler. For every interrupt, there is a fixed location in memory that holds the address of its ISR. The group of memory locations set aside to hold the addresses of ISRs is called the interrupt vector table. You don’t have to know exact locations of these vectors. Compiler does this for you. So basically when timing is important for

microcontroller to react or when it should detect signal from outside world that occurs relatively rear but lasts for very short interval than interrupt is a better solution.
An alternative to interrupts is the method known as polling. In the **Polling method**, the microcontroller must access by himself the device and ask for the information it needs for processing. In fact, we see that in the Polling method the external devices are not independent systems; they depend on the microcontroller, and only the micro is entitled to obtain access to the information it needs.
The main drawback of this method when writing program is waste of time of microcontroller, which needs to wait and check whether the new information has arrived. The microcontroller continuously monitors the status of a given device. When the condition is met, it performs the device. After that, it moves on to monitor the next device until everyone is serviced. The microcontroller checks all devices in a round robin fashion. Polling is like sitting and checking your phone to find out if someone calls you as opposed to interrupts where you have to wait for it to ring. It's better to use interrupts if the processor is loaded with tasks and the response time is not really critical; polling is better to use when the processor needs to response to an event right away.

For our project we will be using polling because the microprocessor (Raspberry pi 3) we are using doesn't come with interrupts, so we are forced to use the polling method. The polling method makes the coding simpler because you don't have to setup interrupt handling but it can be slower because it's always checking on the device which will make spend less time on other essential tasks.

3.3) COMPONENT DECISIONS

3.3.1) MICRO-CONTROLLER

Our microcontroller is the most important part of our device and functions as our hub to everything else. Essentially since it is the heart of our device it will be implemented in a way that everything revolves around it. Our goal for its function is to communicate with all parts of our device and to essentially know what is going on at all times along with driving things such as our LED's. Our secondary objectives also is that it will serve as the storage place for all our information that may be recalled later.

For our project we are using Raspberry Pi 3 which is the generation of the Raspberry Pi amily, Model B, 1 GB RAM. We decided to go with this one because it is one of the most popular one nowadays and it also fits our design consideration. According to Raspberrypi.org, it is the recommended one for school project use. It supports all the features that were available in the previous versions and also has backward compatibility. Also it fits well within our budget. It also comes with an HDMI cable, a microSD card with a card reader, the mother board with: a Broadcom CPU (BCM2837) 4x ARM Cortex VideoCore IV, a 40-Pin GPIO header (populated), a Wifi antenna, a DSI display connector, a MicroSD card slot, a 2.5/5V MicroUSB, a HDMI Port, a CSI Camera Connector, a 4-Pole 3.5 mm Audio & Composite Video, Ethernet Port, 2 USB Port, Bluetooth 4.1 Classic, and a 1 GB RAM LPDDR2 (900 MHz). It also comes with a cover case to protect the motherboard.

It features a wireless radio that is very small with a Broadcom BCM43438 chip that provides 2.4 GHz 802.11nn wireless LAN, Bluetooth Low Energy, and Bluetooth 4.1 Classic radio support. It is smartly built onto the board so it can be implemented at a lower

cost. As opposed to the more used qualified module approach, the only feature that is not used is a disconnected FM radio receiver.

It also comes with a built antenna, so there's no need to connect an external one. The radios are connected to the antenna and built into the board so the device's size can be kept minimum. The antenna should be able to easily pick up wireless LAN and Bluetooth signals from our required distance.

The Broadcom BCM2837 SoC (system-on-chip) comes with four high-performance ARM Cortex-A53 processing cores running at 1.2GHz with 32kB Level 1 and 512kB level 2 cache memory, also it comes with a VideoCore IV graphics processor linking to a 1GB LPDDR2 memory module on the back of the board.

Like all the Pi series microcontrollers, the Raspberry Pi 3 features the 40-pin GPIO (general-purpose input-output) header. The only thing that's changed is that the UART is exposed to a switch on the GPIO's pin, which is handled internally by the operating system.

The same SMSC LAN9514 chip that were used in the Raspberry and Raspberry Pi 2 is the same chip that is featured in the Raspberry Pi 3, but this time with 10/100 Ethernet connectivity and four USB ports connected to the board. The chip is connected to the SoC via a single USB channel, it acts as a USB-to-Ethernet adaptor and USB hub.

3.3.2) TRANSMITTER/RECIEVER/TRANSPONDER

For the purpose of this project we must decide on the type of transmitter and receiver required for the sensors to communicate with the Raspberry Pi. The transmitter that will be used is the TX433. This transmitter works with a 433MHz receiver and can communicate well with the Raspberry Pi. They only work with communicating data one-way so two pairs required, but must be set at different frequencies to be able to work. The transmitter can operate up to a range of 90m and has a data rate of 4800bps and requires a 5V supply voltage. The minimum required current is 9mA and the max is 40mA. The transmitting velocity is less than 10 kilobytes per second. The receiver that will be used is the RX433 module and it operates at a frequency of 433MHz. The range that the receiver can handle is about 100m in open air and the power required is 4.5-5.5 V and the data rate is 4800 bps. The working current required is less than 5.5mA max and the device has a bandwidth of 2MHz. The transmitting velocity for the receiver is less than 9.6 kilobytes per second. The transmitter and receiver is ideal for this project because of the power consumption required is not extensive on the system and is manageable. The frequency required for the transmitter and receiver to communicate is ideal because it does not interfere with any of the other components that require frequency as communication such as the NFC tag. The distance that the transmitter and receiver covers is enough for the intended use of the environment that it will be implemented in. The cost of both components are fairly cheap compared to other components that were being considered. The modulation for both of the modules is amplitude-shift keying in which the digital data is represented as variations in the amplitude of a carrier wave.

3.3.3) LED

As for LEDs we're only using two colors, red and green, five millimeters LEDs from microtivity; the 5 mm is cheap and also comes with resistors. They are through hole LEDs (with two legs). The red one comes with ¼ watt resistor for connections, emits focused, bright red light and forward between 1.9 to 2.0 Volts. The green one forward between 2.0 to 2.1 Volts. We decide to go with the two legged one because of its simplicity to set up:

one negative lead, one positive lead which is the longest. It is also the most used in everyday life and compatible with various circuit boards.
We have many advantages in using LEDs instead of others, such as: efficiency; LEDs can emit more wattage than incandescent light bulb and also its shape and size don't affect its lightning fixtures like fluorescent light bulbs. Color; LEDs don't need color filter to emit an intended color as opposed to traditional lightning method. Size; LEDs can be as small as 2mm even smaller and they are easy to attached to printed board circuits. Warmup time; LEDs light up within very little time, some will become fully bright under a microsecond and the ones used in communications system can be even faster. Focus; you can design the solid package of the LED to focus its light as opposed to incandescent and fluorescent which require external reflector to acquire light and send it to use. Lifetime; LEDs usually live longer, approximately 50,000 to 100,000 hours, as opposed to incandescent, approximately 750 to 2,000 hours and fluorescent, approximately 10,000 hours according to Digitallumens.com. Dimming; LEDs can be dimmed very easily either by pulse-width modulation or lowering the forward current. Cycling; LEDs are the perfect choice for on and off cycling as opposed to incandescent and fluorescent which fail faster when cycled often. Shock resistant; LEDs are solid-state components, so they are difficult to damage when they get shock externally as opposed to incandescent and fluorescent which are very fragile. Cool light; LEDs, as opposed to most other light sources, radiate little heat in form Infrared that can cause damage to sensitive object or fabrics. Slow failure time; LEDs usually fail over time by dimming instead of sudden failure like incandescent bulbs.
There are some disadvantages like prices, temperature dependence, voltage sensitivity, etc but they are outdone by the advantages.

3.3.4) LCD

Our LCD is a 1506 LCD module for the Raspberry Pi with a 32 MHz speed. It supports all the Raspberry Pi versions currently on the market. The NeoSec TFT is a small LCD display that pushes onto your Raspberry Pi (Model A/B) via the GPIO pins. The screen makes use of nearly all available space above the Pi, allowing a decent resolution. It comes packaged in a small clip-top box with everything inside.

The screen features touch screen functionality alongside a little directional touch pad attached via a belt cable. I've not seen this kind of thing in the market yet so this was a fresh approach and comes in handy if you need precise mouse control (or you don't want finger marks on your screen!). The touchscreen itself does also come with a stylus, another nice little addition to the package.

The GPIO is accessible via the extra PCB tab below the screen, allowing you to connect any kind of header you want (or none at all). It looks as though this could be cut/snapped off if required, as there are a number of drill holes creating the break for you.
It's subtle and out of the way: A buzzer is mounted to the rear of the screen which makes a sound every time you touch. I found this quite annoying, however, NeoSec have told me that on the latest model this is optional, in case you don't want the 'beeps'.

Quality

The screen itself is nice and bright, with rich blacks. The resolution allows the font to be a nice size allowing the user read on the screen without issue. I love the size of this screen, and the way it covers the Pi completely. There's also very little blank space

on the screen itself. It does feel a little more delicate than some other screens I have seen previously. This is probably down to the fact that there's no PCB area around it. There's also nowhere to fit nylon screws or similar to help keep things steady – but I did a DIY job on this which isn't difficult.

I guess it's hard to keep us happy – we all want the biggest screen on our Pi, but to achieve that you need to remove the PCB area. This is a tough balance to strike.

Competitor Comparison

There's an obvious competitor that you can't help but compare to when you see other small Raspberry Pi screens…so this review will focus on the pros/cons of the NeoSec 3.5″ TFT compared to the 2.8″ PiTFT from Adafruit.

NeoSec vs AdaFruit

Although these screens are different in features and size, they're suitable for comparison in terms of "Raspberry Pi supported touchscreen".

Price

I'm mentioning price up front as I think it's important to consider this whilst reviewing each screen. There's not a lot in it price-wise.

This NeoSec screen package comes in at $41 – that's the screen, touchpad, pen and DVD. This is also pre-assembled. A basic package with just the screen is $25.

The AdaFruit screen rolls up at $34.95 – including the screen only (no buttons). You also have to assemble this screen, including soldering the main GPIO connector and taping down the screen element.

Verdict: NeoSec wins this one. Considering the extras, you get with it, I personally think it's a better deal for a 'screen on Pi' solution. (and the basic $25 package is clearly much cheaper)

Out of the Box

The PiTFT requires assembly, including GPIO and button soldering, and taping the screen to the PCB. That tape isn't very sticky at all so you'll most likely need to get your own – I used No Nails tape. The NeoSec screen comes pre-assembled and ready to go. No assembly required.

Screen Size

The PiTFT rocks up at 2.8″ at 320×240 resolution – using the extra space around it for the PCB which provides holes for fitting support screws.

The NeoSec weighs in at a more comfortable 3.5″ and a clearer 320x480 resolution. The 3.5″ screen covers more of the Pi, which I think looks much smarter. That extra screen space does come at a price, which is that it has a slightly more delicate feel to it and has no mounting holes for support screws like the PiTFT. The font on the NeoSec screen seems smaller yet clearer, allowing more on screen, but there may be a way to match this on the PiTFT that I haven't discovered yet.

Fit/Quality

The PiTFT has holes around it to use nylon screws as a screen support. It also has PCB area around the screen acting as a bit of protection. The PCB covers the entire underside of the screen, ensuring no light comes out of the back.

The NeoSec screen doesn't have any support holes, and has no PCB area around the screen (but it's a bigger screen, which is more important in my eyes). I can't see the NeoSec doing well on a Model A without that Ethernet port holding it up. The NeoSec's PCB doesn't cover the rear of the screen either, so light comes out on to your Pi.

GPIO

The PiTFT has an optional upside-down connector to attach a belt to breakout to a breadboard. I don't like the whole belt thing, it feels a bit too 90's computing for me and the upside down back to front thing makes it hard to do something different, like add a regular GPIO header.

It's hidden away though, which is nice and tidy if you're not using it and I'm pretty sure I'm the only person who hasn't purchased a Cobbler belt breakout so maybe don't listen to me!

The NeoSec is a little more traditional with the GPIO, and simply gives you a mirror of the GPIO next to the screen. This is good if you want a simple prototyping access, but perhaps not as ideal if you just want a screen, as it does stick out. It looks as though it can be removed as drill holes indicate an easy option to cut or snap it off.

Features

The PiTFT comes ready to fit 4 tactile buttons to, however these need to be purchased and fitted separately. The blue PCB of the Adafruit board is attractive when compared to traditional colours.

The NeoSec comes packed with a touchpad, touchscreen pen and DVD software. The PCB can't really be seen, but it is the standard green colour.

Support

The PiTFT benefits from the massive following and fan base that Adafruit command. Their forums are full of information, and generally a lot of people buy their products, so most people have had the issue and written about it on blogs and forums.

The down side of a large company like this is that getting 1:1 email support quickly is unlikely due to the sheer number of requests they must receive (although I didn't try this option, purely on the assumption it would take too long).

The NeoSec screen doesn't have that massive following that AdaFruit does, so finding information already out there can be difficult. Fortunately, NeoSec counteract this by providing excellent personal support by email and also regularly on the Raspberry Pi forum.

3.3.5) TRANSFORMER

Our transformer selection was based off of our Voltage Regulator requirements. We needed to step down a AC wall plug to something our circuitry could actually handle in our power system. Unfortunately, creating components ourselves for this showed that it could extremely dangerous for us or those we would be presenting to. So I opted for an off the shelf solution instead of creating one ourselves. Our voltage needed to still be decently high within the range of 6V to 38V in order to meet our needs. One thing that needed to be denoted with this is that when we rectify our AC voltage it would raise our Vrms to about ~40% higher then put in. Unfortunately, very few AC->AC transformers existed on the market as a consumer product. The way this was worked around was that I had pulled up parts from a HVAC seller as HVAC appeared to be the most common industry to deal with AC->AC conversion for their circuitry. This led to needing a determination between Class I and Class II transformers. The difference is Class I transformers can get wet while Class II transformers are dry use only. So we opted for a Class II transformers around the center of our range, but with a good safety margin to meet our needs. What was available as an easy to get off the shelf solution was an MGT-1220 transformer. The stats for the

transformer was a 120VAC 60Hz input and an output of 12VAC 60HZ 20VA. Being a 12VAC and 20 VA this means our output Amperage should be about 1.67A. This means this particular transformer falls well within our range when rectified for our circuit. There isn't too much faith this item would be in stock in the future for our particular seller, however, since the part is standard use in the industry, many alternatives should exist for this type of item.

3.3.6) DIODES

Our diode choice was mainly based on if it could handle the Voltage/Current we had to send through our system. Based on our transformer we needed a diode capable of dealing with roughly 16 Volts and a current of 1.6A. In order to have a good safety margin we went far higher than the required stats. It appeared though due to the low cost of diodes we had a massive amount of flexibility in our choices for them. In the end we made our choice based off the cost per unit and the maximums for our diode and came out with using the 1N5408 Diode. It supports a max average current of 3A and a voltage of 1000V which was far greater then needed. But, it appeared to be one of the lowest costing diodes for us. There was also a lot of faith that the item would remain in stock as it has been sold for many years for this item.

3.3.7) DC-DC CONVERTER (VOLTAGE REGULATOR)

The main consideration for our DC->DC voltage regulator was the output current and voltage. The primary consideration for our regulator was that it would power a microcontroller as this is our primary system we are worried about powering. This part determined what the entire rest of the power system would require in terms of stats and was actually our first chosen part. The Raspberry Pi has a maximum power requirement that was 2.5A and 5V for its input. It also must be very close to these values or else fuses with trip and such on the board if 6V is fed destroying the board. For this reason, we decided against creating our own regulator as it needed to be very exact for our use. The original search led to looking for circuits that were classified as the TL08XX. Since we needed a 5V output the determined circuit type was to be a TL085 type. However, then it was required to narrow them down by their output amperage for our circuit to meet the 2.5A range. This lead to only about two choices in circuits in terms of companies that manufactured these. We went with the D24V25F5 circuit by Pololu as it not only met these output requirements, but allowed for a generous input voltage to our circuit. It also appeared as if they would have these items in stock in case to many of the chips were destroyed during testing as they had 1000's listed available.

3.3.8) CAPACITORS

There are only about 2-3 capacitors needed within our circuit. Our main consideration is our smoothing capacitor which is required to keep our ripple as low as possible. We use the equation C=I/(F*Vr) to solve what we need for our circuit. Ideally we will be using a 9000 Microfarad capacitor for our circuit. This will most likely be quite a few capacitors in parallel in order to reach our required capacitance. We in the end used two 4700 Microfarad capacitors instead of many smaller ones in case issues would arise from putting many in parallel. However, according to data we learned about afterwards, it would appear multiple smaller ones is the industry standard currently for items that aren't

a resistive load but instead items that have smaller duty cycles. Our follow up capacitor that will exist on the outside of our Buck regular isn't nearly as important, but follows good practice guidelines when dealing with a system. This one will ideally be very small ceramic capacitor that is under 1 Microfarad. It is just to filter out some excess noise for us while ceramic capacitors are ideal for this.

3.3.9) RFID/NFC

Deciding on a module to implement to read and write radio-frequency identification tags wasn't easy. As stated in the research portion the implementation of the MFRC522 module was ideal, but there were some problems that arise. As a result, we have opted to use the PN532 RFID/NFC Breakout and Shield. This module would be more suited for the goal of the overall project because it was made specifically to work with the Raspberry Pi. The chip that is embedded in the PN532 is very diverse because it is implemented in other platforms such as mobile phones. The board dimensions are as follows 2" (51mm) x 4.7" (117.7mm) and 0.425" (1.1mm) thick. It uses a I^2C 7-bit address 0x48 scheme. If enabled, it is also able to enter low-power mode when not in use. The chip offers RFID and Near field communication (NFC) which is used in devices to communicate with cell phones bi-directionally or make payment processing. It gives us a wider range on how we can innovate our system and include other features. This module supports multiple NFC/RFID tags type 1 thru 4. There are multiple types of communication offered with this board such as UART that can transmit at any baud rate, I^2C, and Serial Peripheral Interface (SPI). It is supported by a libnfc library which makes implementation ideal because of the many options available to program the radio-frequency identification tag. When plugging in the FTDI cable and using the FTDI serial port to communicate this opens up the option of using Near field communication dev with any operating system such as Windows, Mac, or
Linux. There's an onboard Light emitting diode that will be used to notify the user if the reader is working and it has a 3.3v regulator built in. The antenna implemented on the board is a 13.56 MHz strip line antenna that is approximately 0.1 inches, and also 4050 level shifter chip.

3.3.10) MOTION SENSORS

The HC-SR501 PIR motion sensor is the one we choose to use in our system because it is based on infrared technology and automatic control module; it also uses Germany imported LHI778 probe design, high sensibility and reliability. With a voltage range between 5V and 20V, a power consumption of 65mA, a Transistor-Transistor Logic output of 3.3V high and 0V low, a lock time of .2 second, a sensing range of less than 120 degrees within 7 meters, temperature between -15 degree to 70 degrees and 23mm diameter. Some of its features are: automatic induction to enter the sensing range if the output is high, photosensitive control that can be set day or light intensity without induction, temperature compensation that can be used to compensate performance, induction block time (2.5s default block time), wide operating voltage range (DC 4.5V-20V default), micropower consumption with static current less than 50 microamps, output high signals. HC-SR501 uses differential detection with two pyroelectric infrared sensors. By taking difference of the values, the average temperature from the field of view of sensor is removed and thereby reducing false positives.

Interfacing HC-SR501 with Raspberry Pi is easy because the output of sensor is Pi friendly ie. 3.3V and it can be powered from the 5V rail of Pi. The PIR sensor, HC-SR501 consist of 3 pins: Vcc – 4.5V to 20V, Input power, OUTPUT – TTL output of sensor 0V, 3.3V, GND – Ground

This motion sensor will work perfectly with our system as it has most of the features we're looking for

3.3.11) TRANSISTORS

This project required a transistor to enable the LED. The decision factor that played a role in this was due to the fact that the GPIO pins on the Raspberry Pi can only use up to 50 mA and we did not want to overload it. That's the reason a transistor was added. Specifically, the 2N2222 transistor because we are familiar with it since we have used it in numerous classes and we have knowledge on the required turn on voltage and when it will enter saturation. We also know the range of the beta required for our calculations. The cost of the transistor is very cheap as oppose to other components. The transistor is a NPN bipolar junction transistor and it is used for low power and medium voltage and can operate at high speeds. With the use of this transistor we can cut down the power consumption of the LED to about 26µA. This makes the transistor an ideal component that should be implemented in this system to help us save current usage which works with the power required for this system that is based in our calculations.

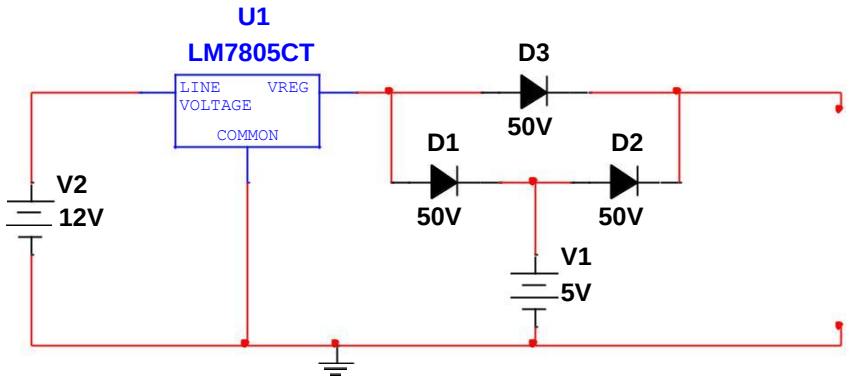
3.3.12) PIEZO BUZZER

When it comes to an alarm system sound is a key aspect because it notifies the user when there is a breach in their system due to motion being detected or even if there is an intrusion in from one of the doors or windows. The component decided on to warn the user of any intrusion is the piezo buzzer. The reason for this was because the cost of the buzzer was not much. It met all the requirements needed for this project. It plays a clear sound, its light weight, there are no contacts so there is no noise and it is highly reliable, and it has a low power consumption. It is very versatile and easy to implement into our system. It has a sound pressure of about 70 dB. It operates on 6-18VDC and it carries 10mA of current at 12V. It has a tame buzzer tone at 2.8 KHz. The buzzer is able to be coded to where it will be able to have multiple tones which is ideal because we will need different tones for the motion, arm and disarm, and the tilt sensors. This will give the user an easy way to distinguish what the system is trying to alert them about.

3.4) POSSIBLE DESIGN IMPLEMENTATIONS

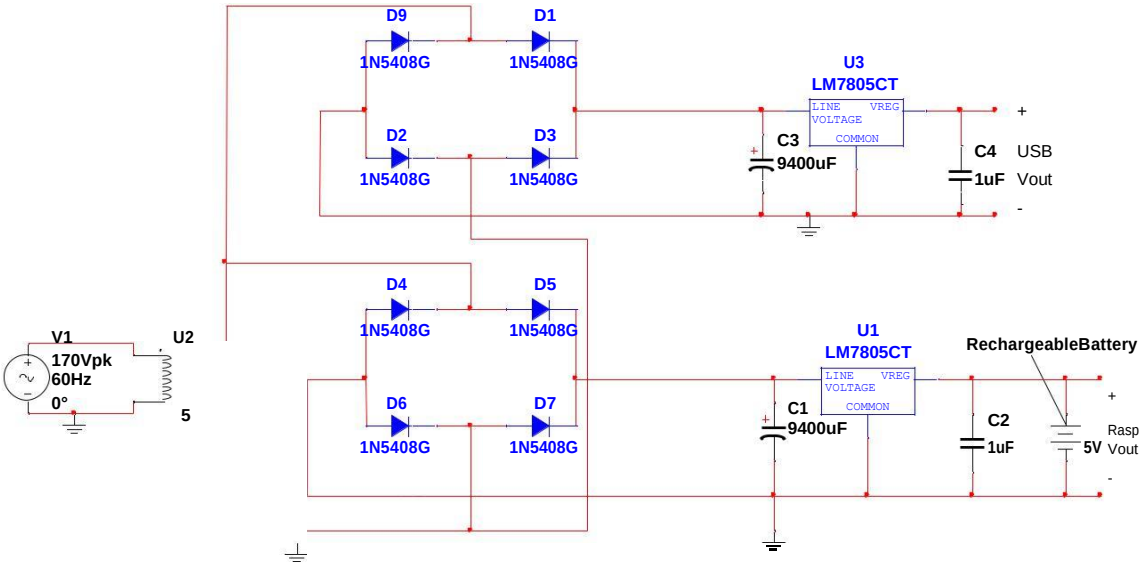
3.4.1) CONTINUOUS UPS

Below is a possible implementation of the most basic sort of continuous UPS battery backup system one can make. It uses diodes to pass voltage that measures the LM7805 linear regulator on loss of power.



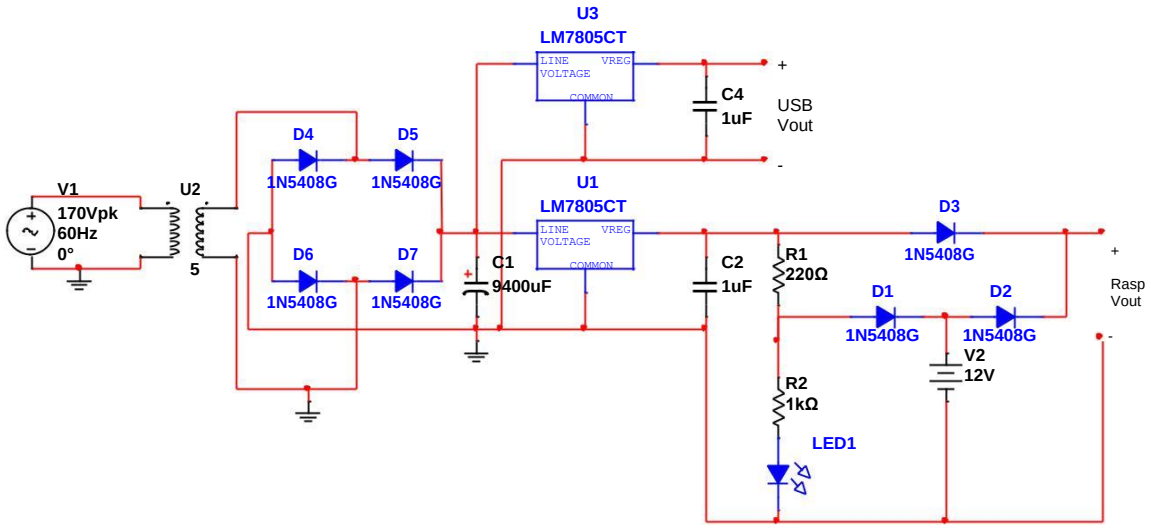
3.4.2) AC TO DC CONVERTER DUO OUTPUT

Below was an early prototype and possible design implementation of an earlier version of our circuit that was supposed to have a battery backup. A more streamlined efficient design was decided upon in the end that cut down on use of diodes in general.



*Design was created using NI Multisim.

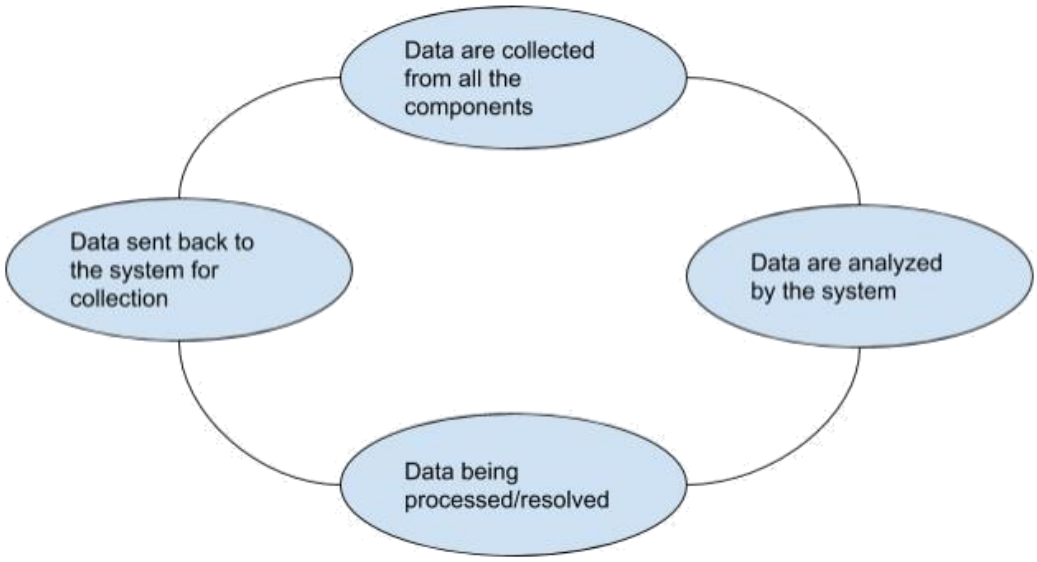
Below was a more refined version of the previous image which although would work, added complication into the circuit in a way that wasn't necessarily needed. We eventually opted for one that cut down on complication in the end to allow assisting with others.



*Design was created using NI Multisim.

3.4.3) DATA HANDLING

Our objective is to save data provided by the system so that we can use them as statistic to improve the system. When collecting data from the system, we will take into account every single detail. Data will be used to keep track of the system operation/execution on a daily basis We will setup a way for the system to mark all the errors/failure during execution so that we can go back, trace and fix the issue. A data handling cycle will be developed to make the process easier. The system memory will be used to keep data on a daily basis then send feedback



Collection of data - the system will collect all data such as: when someone log into the system, number of time the system turns on/off, number of time doors are open, number of time the alarm goes and reason for going off, which sensor causes the alarm to go off, number of time a pin is used to enable the system. After collecting data, the system will move to analyzation of the data.

Analyzing data - in the process of analyzing the data, the will synthesize the raw data into summary information, including calculations of average and spread, and present the information in tables, graphs and charts. Data will be interpreted so they can be categorized for processing.

Processing data - after analyzing the data, we are in the process of selecting which data to keep and which not to keep. All the data kept for errors/issues will undergo the resolution process. Data that are not errors/issues will be saved in memory for record purposes and will be use as statistic for the system.

Sending Data back to the system - after all the went through collection, analyzation and processing, the resolved data will be sent back in the system for recollection and so the cycle will keep going.

Some issues that we can consider in handling the data is to ensure integrity of the data which include the following

The type of data that are handled and the impact on the system

Procedures that can describe the duration the data should be kept, who should hadle it, when and how

Responsibilities and privileges which is how the data are being divided among handlers, when to do that and why

The content of the data and the storage capacity, the reliability of the data, its longevity, the requirements of the storage, the effectiveness and how to upgrade the system

4) HARDWARE DESIGN

4.1) STANDARDS AND DESIGN CONSTRAINTS

4.1.1) PUSH BUTTON HARDWARE CONSTRAINTS

The main constraint that we have come across as a team when designing the pushbuttons was the placement of the resistors. If we placed the current limiting resistors in the output instead of the input, then this could be extremely catastrophic for the microcontroller. Therefore, we had to be extremely careful with our calculations and our resistor locations. If we made one mistake when setting what current our outputs were at, then this could damage the Raspberry. Additionally, if we also place the resistors in the wrong place, then the outputs on the microcontroller would negate the inputs where the rows are reading from, and flow into each other. If this happened, then we would potentially cause irreversible damage to our microcontroller. This would cause us to have to order another Raspberry Pi which would put us about a week behind schedule when testing our equipment. Because we are compressed with time, we can't afford to be behind schedule.

We have already had to adjust our design multiple times before testing it in order to be confident with our schematic that when a button is pressed our microcontroller will not be harmed or damaged. The main issue we had in this area was the current limiting resistor placement. If we decided to place the current limiting resistors on the inputs, then when two buttons are pressed it could consequently short out the two GPIO outputs. This would be the equivalent of creating a short on a battery. Referring to Ohms Law, when there is no resistance from a battery, then in theory the battery will want to supply an infinite amount of current. This is the same for the output pins on the GPIO. If we short out two output pins, one supplying 3.3 volts, and the other supplying 0 volts, then since there is no current limiting resistor then the circuit would be faulty and would damage the raspberry.

Not only is the placement of the resistors an issue, but the value of these resistors are something that we had to be careful about as well. For this example, let's assume we decided to connect the current limiting resistors on the four inputs. Now let's assume that two outputs are not shorted. Instead two rows were shorted that are connected to a single output that is supplying 3.3 volts. If we set the four resistors to be too low, then this could cause the output current to be extremely high. The single output would not only be using an inefficient amount of power, but if the current exceeded the maximum amount that the GPIO pins are allowed then this would be very risky, and potentially catastrophic to the Raspberry. For example, if we decided to use a 100-ohm resistor, then the current supplied by the single output would be 33mA. This calculation is assuming only a single button is pressed. If two buttons were pressed, thus shorting out two of the input pins, then the current would effectively double. The data sheet for the raspberry states that the maximum output current for the combined GPIO pins is 50mA. Since the current would be 66 mA when two inputs are shorted, then we would basically need to buy a new Raspberry pi.

We have come to the conclusion that in order to protect against the outputs being shorted to each other, we should attach the current limiting resistors to them. We have also decided to not include a current limiting resistor to the inputs. The simple reason for this decision is because if we chose a 1k ohm resistor on the outputs, then that will be enough

resistance to reduce the current. Because only a single output will be supplying voltage at a time, if two outputs are shorted together, then the output supplying the voltage will negate the output not supplying voltage and flow through the current limiting resistor and finally through the short that is connected to the input.

4.1.2) RASPBERRY PI CHARGING STANADARDS

The Raspberry Pi has a unique set of acceptable power inputs as it doesn't share common Voltage and Amperage levels that most devices require. The most notable thing is that the Pi is powered by using a Micro-USB. However, the current draw from the device is far too high to be powered by hooking the device into traditional USB plugs on the PC. This means that the only effective way to power the device with heavy use is by plugging it into the wall itself. One of the notable less documented features about the devices power system is it has a Polyfuse to disallow the device from having too much current put into the input and damaging it. A Polyfuse is essentially a fuse that can be reused and reset over and over again the case of overcurrent. It automatically will reset itself so it isn't needed to be reset manually by the user. The fuse essentially will reset its state when the overcurrent is removed from the item. Now for the voltage of the device, all models require a 5V input into them in order to function. The voltage must be very exact or else it may damage the device in general. The tolerance according to sources I could find fall within a +-5% voltage range. This means the device will get damaged if it goes out of the boundary between 4.75V and 5.25V. According to any source I can locate this is also a common industry standard with many other applications out there. This applies to all versions of the Raspberry PI microcontrollers currently on the market. The next real difference comes in the form of the allowed amount of current per device. Every single device has a different measurement when it comes to allowed current. They appear to have an upper limit when it comes to this portion of the device, however, no lower bounds. The reason this doesn't appear the damage the circuitry when disconnected from the device is because of on board backflow protection within the device. This means essentially we only need to consider the upper limit of Amperage. The Polyfuses of each device is rated at the recommended PSU current capacity limit given by Raspberry PI's website itself. The device we will be planning around in particular to the Raspberry Pi 3B. A table of the collected standards can be viewed below.

Product	*Voltage Required*	*Voltage Tolerance +-5%*	*PSU current capacity recommended*	*Bare board current draw.*	*USB peripheral power drawn*
Raspberry Pi A	5V	Between 4.75V and 5.25V	700mA	200mA	500mA
Raspberry Pi B	5V	Between 4.75V and 5.25V	1.2A	500mA	500mA

Raspberry Pi A+	5V	Between 4.75V and 5.25V	700mA	180mA	500mA
Raspberry Pi B+	5V	Between 4.75V and 5.25V	1.8A	330mA	600mA/1.2A Switchable
Raspberry Pi 2 Model B	5V	Between 4.75V and 5.25V	1.8A		600mA/1.2A Switchable
Raspberry Pi 3 Model B	5V	Between 4.75V and 5.25V	2.5A	400mA	1.2A

4.1.3) USB STANDARDS

The charging standards for USB's have changed throughout the years. There are also many types of USB charging standards now on the market. The four most common USB types out there appear to be USB 1.0, 2.0, 3.0, 3.1 and a USB-C. For our project we will be ideally using a combination of USB charging standards, USB 2.0 to Micro and company specific standards.

Our wiring layout will reflect the current charging standard in which there will be no connection to Data+ and Data-. This means we will only be utilizing Vbus and Ground for our standard. These wires are indicated by a Red and Black wire respectively.

Wire	**Color**
Vbus	**Red**
Data+	None
Data-	None
Ground	**Black**

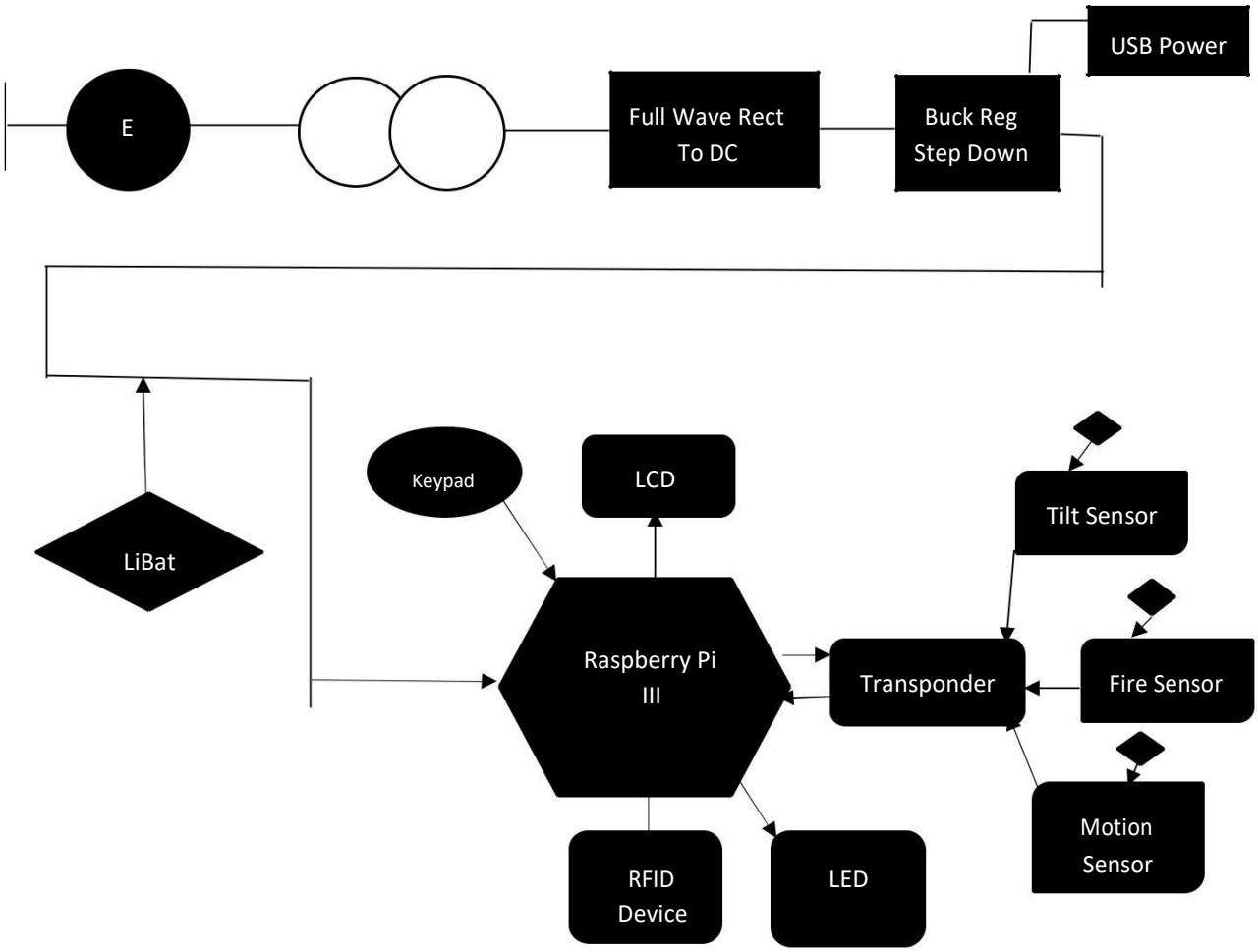
First thing for our standard is that our project requires a voltage of around 5V. According to specifications our Voltage tolerance should fall within +-5%. However, this will vary USB port type as they have progressively become more tolerant. The chart below demonstrates the current tolerances needed.

Type	**Ideal Voltage**	**Upper Limit**	**Lower Limit**
USB 1.0	5V	5.25V	4.45V
USB 2.0	5V	5.25V	4.4V
USB 3.0	5V	5.25V	4.0V

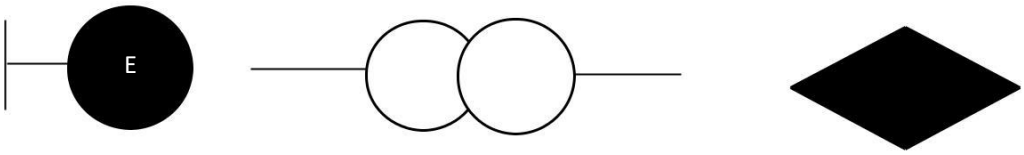
The standard we will be using in terms of current and power output is based off of the needs of the Raspberry Pi 3. The PI requires a maximum amperage of 2.5A. This means we only have 4 options for standards that were available to us. However, the PI is also easily driver by a Micro-USB. So we opted for using the Power Delivery Micro-Format. Its exact specs can be seen below with others for comparison.

Specification	Current(A)	Power(W)
Low-Power Device	100mA	.5
Low-Power Superspeed	150mA	.75
High-Power Device	500mA	2.5
High-Power Superspeed	900mA	4.5
Battery Charging 1.2	5A	25
Type-C	1.5A	7.5
Type-C	3A	15
Power Delivery Micro-Format	3A	60
Power Delivery Standard or Type-C	5A	100

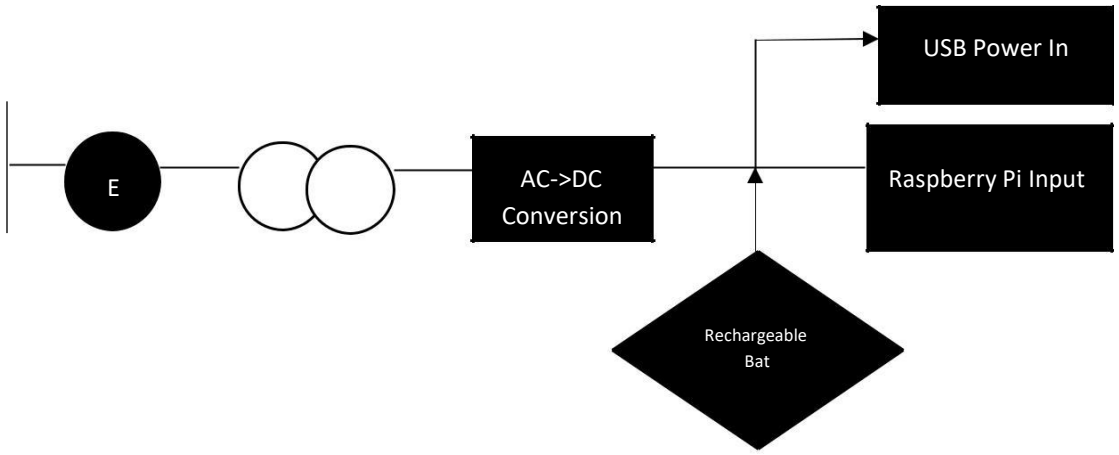
4.2) OVERVIEW FLOWCHART OF HARDWARE



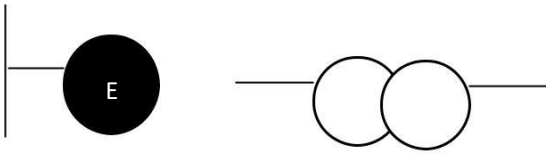
Wall Outlet Transformer Battery



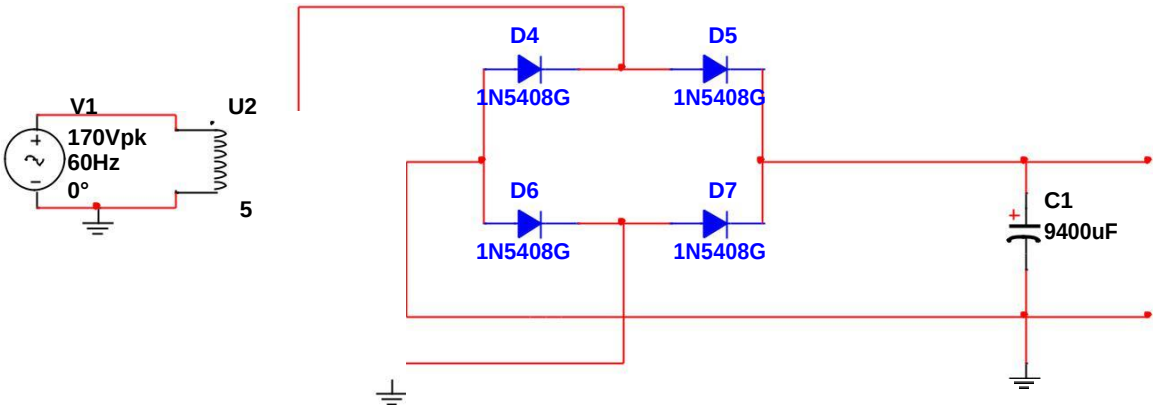
4.3) BATTERY AND POWER SYSTEM



Wall Outlet Transformer



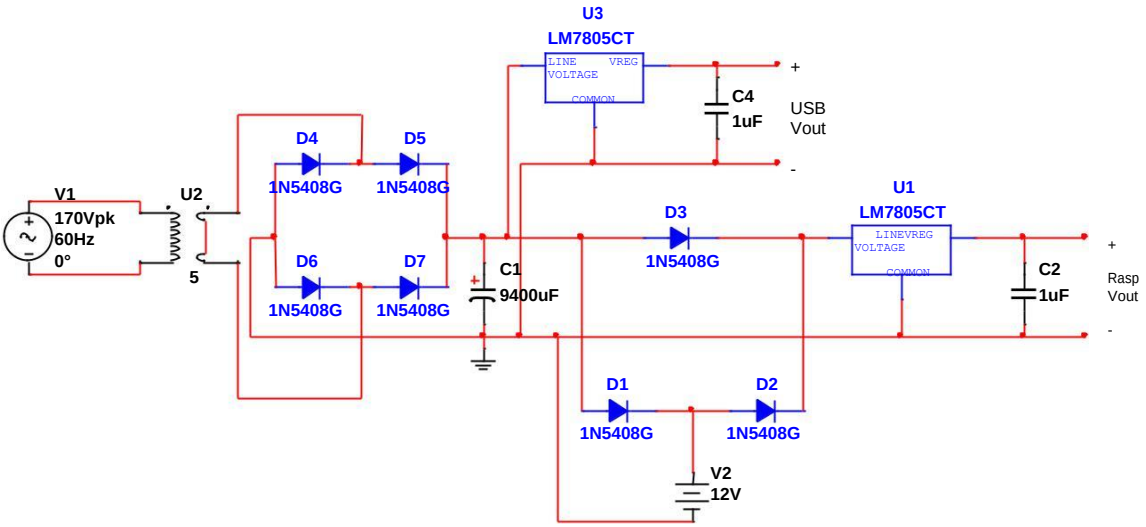
4.3.1) DESIGNED AC TO DC CONVERTER



*Design was created using NI Multisim.

Our AC to DC converter pretty much is the most basic standard on the market currently. This is one of the most common designs out there at the moment when it comes to conversion. Overall, it functions off the principles of full wave rectification using four diodes that alternate at various times in order to rectify our sine waves that come from an AC wall socket. Afterwards the resulting waves are put through some very large capacitors which holds a charge and don't have time to discharge resulting in a DC signal. The only thing to really note with this portion of the design is it increases our Vrms by roughly 40% of the original value.

4.3.2) FULL POWER SYSTEM DESIGN

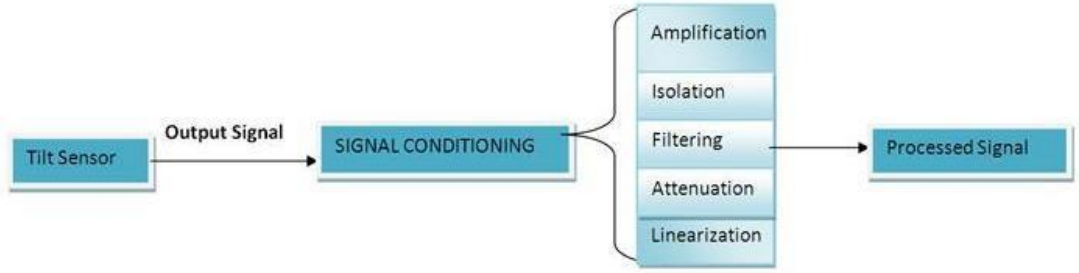


*Design was created using NI Multisim.

The rest of our power system is added after the AC to DC conversion portion. Essentially, diodes are added to allow attaching an external battery source along with Buck Converters which output the refined DC Voltage and current wanted to our system. After each Buck converter additional capacitors were added in the event of noise to help filter some of it out. Essentially, they are preferred to be very small ceramic capacitors as they are generally better at this process.

4.3.3) TILT SENSORS

The tilt sensor will communicate with the system wirelessly. We're using a pushbutton for the tilt sensor. We're using a pull-up resistor and connect the sensor to a digital pin that we will read when we need to. The setup is represented in figure below



The sensor will send the signal over, then go through some conditions before being processed.

The sensor will transmit a signal to the base station when the attached switch changes state (off/on). The transmitter is optimized for extremely low current consumption so that it is able to last over a year on a single set of batteries. The antennae of the transmitter can be adjusted for longer distances, or for greater strength to compensate for walls or other physical barriers. The transmitter comes preconfigured with a unique number identifier that will be used to uniquely identify the sensor. This enables you to have as many wireless sensors as you want. Each switch that is attached to the sensor is identified as Switch A and Switch B in the serial message received by the Raspberry Pi. The Python code required to read the serial communications stream is provided.

The micro-controller has the job of processing the signal from the sensor. Since it knows the timing for the excitation to the sensor, it needs to take voltage level samples from the sensor. This is done during each half cycle. Multiple samples during each half cycle are recommended since these can result in a higher bit count and can also be averaged for a more accurate result. One half cycle is subtracted from the other half cycle and the result will be an indication of the tilt angle. The sign of the result will indicate which direction from null that the sensor is tilted. The micro-controller now has a value that is a representation of the tilt angle. This value can be internally processed or sent directly to another external component. As mentioned above, the critical factors are the signals to and from the sensor. Depending on the choice of micro-controllers, consideration needs to be given to have equal and adequate source and sink drive current to the output ports. If the micro-controller loses power or is turned off in successively short intervals, it may become necessary to incorporate a shutdown routine to ensure that a full cycle is delivered to the sensor thus preventing non-symmetry or direct current.

4.3.4) EXTRA POWER REQUIREMENTS

One of the batteries we considered implementing into our design is a 9 volt, 175-mAh, nickel-metal hydride, rechargeable battery. With this battery, we will be able to

obtain around 3500 hours of use, in theory, without being replaced or recharged. This is, of course, neglecting the power consumption of the wireless transmitter that we will be including as well. If we assume that the user is going to only arm the system when they are away from their house or while they are asleep, then each week the battery will use up 96 hours of its life. This, of course, is assuming the user is going to obtain eight hours of sleep a day, be home on the weekends, and also work eight hours Monday through Friday. These calculations conclude that the battery system will last about nine months out of the year before we would need to recharge it. These calculations prove that we need a battery that will hold a lot more capacity. Reason being is that we don't want to keep changing and charging our battery system for our motion sensors every nine months out of the year. The process would become tedious since other developed alarm systems have motion sensors that last a couple years before needing to be replaced. Our conclusion from our calculations is that we are going to need a battery that has a higher amp-hour rating.

Since our main concern is a battery that has a much longer lifespan then we will consequently need to spend more money on a higher quality battery. A quick search online has shown us that for around ten dollars we can buy a 9-volt, 1200mA lithium battery. This would eliminate our battery lifespan problems entirely. With this battery running at a constant 50 microamps, it will enable the motion sensor module to operate continuously for 24000 hours. This means if the user decided to leave the motion sensor module on at all times then it would last roughly around 3 years. This would enable the user to not be too concerned about switching the motion sensor module's power off when it is not in use.
If the user decided to switch it off when the system is not armed, then it would last roughly around five years before it needs to be replaced.

Since we are implementing a transmitter to send data from the motion sensor to our Raspberry Pi, then we also need to account for the power consumption that the transmitter is going to use. The transmitter that we have chosen for this project is the TX433. It uses anywhere between 1.5-volts to 12-volts for its operating voltage, depending on how far you want to transmit data and a current draw of 5mA. Unfortunately, the transmitter will be on as long as the motion module is on as well. Therefore, this transmitter will be consuming the majority of the power from the battery.

To accommodate for this huge power requirement, we will be including an electrical switch that will discontinue power from the battery to the transmitter unless a signal needs to be sent to the receiver. There are a couple different switches we can consider implementing into our design. First off, there is a transistor. When a transistor is in saturation mode it acts like a switch. Although, because of the restricted time that we have to deal with we will most likely not be using a transistor. We were experiencing technical difficulties when we tried making the emitter output a voltage when the base was connected to the output of the motion detector module and the collector was connected to the 9-volt power supply.

The next switch we have considered implementing was a transistor-transistor logic and gate. One of the inputs of the and gate can be connected to the 9-volt rail, while the other input is connected to the output of the PIR module. The motion detector will always output a logical low value until the motion has been detected. Once motion has been detected then the two inputs will read logical high which will make the and gate output the

highest of the two voltage values of the two inputs. Like an and gate, we could also design a 2-1 multiplexer that will have one of the inputs set to ground, the other input set to the 9-volt rail and the output of the multiplexer will be connected to the input power of the transmitter. The selector for the multiplexer would be the output of the PIR module. Once the module outputs a high logic value then the multiplexer will select the 9-volt rail which will power the transmitter. The issue with a 2-1 multiplexer is that it has two stages of and gates that the logical values need to pass through. Since TTL circuits are not instantaneous then there will be some delay before the transmitter will be powered on.

Finally, the last switch that we have researched for this type of design is a tristate buffer. The buffer has three pins attached to it, an input, output, and select pin. The output will be connected to Vin of the transmitter. The select line will be connected to the output of the PIR module, and the input will be connected to the 9-volt rail. When the output of the motion detector is at a logical low value then the output will act as a high impedance. Likewise, when the select bit is high then the tristate buffer will pass through the input to the output, thus turning on the transmitter. Since we are able to turn the transmitter on only when an intrusion is detected, then this will increase the life on our battery supply. The only issue with this design is that the user has to physically turn the power to the whole module off when the system is not armed, or it could potentially draw power from detecting motion when it is not needed. Figure 4.3.4.1 illustrates our design for this system.

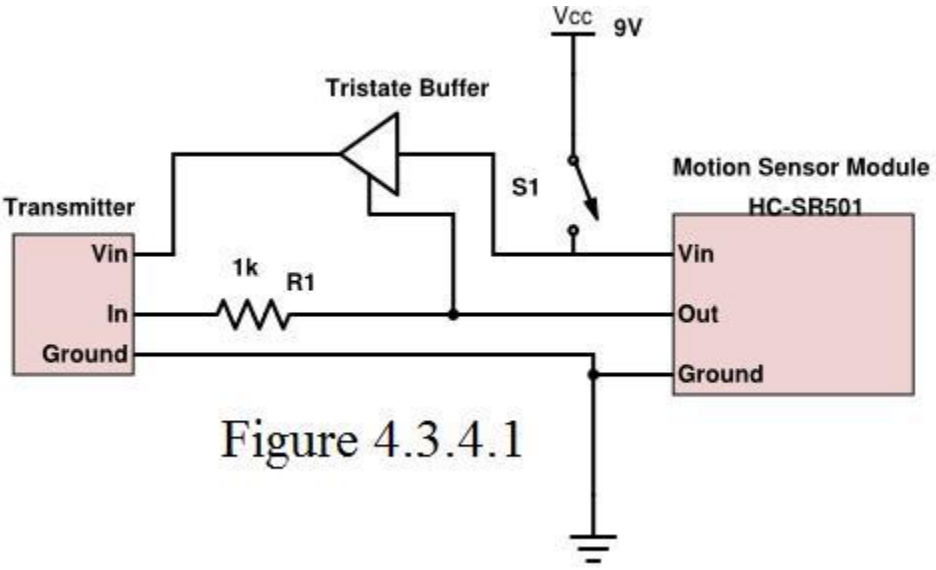


Figure 4.3.4.1

4.4) SENSORS AND WIRELESS COMMUNICATION

4.4.1) MOTION SENSORS

Initially, we wanted to design our own motion sensor device that would detect motion from approximately twenty feet away. Based off our research we have concluded that not only would designing the hardware for the device be extremely tedious, but the design process would be redundant since we would basically be reinventing the wheel. The professional hardware designs that we have researched online were very complex. The

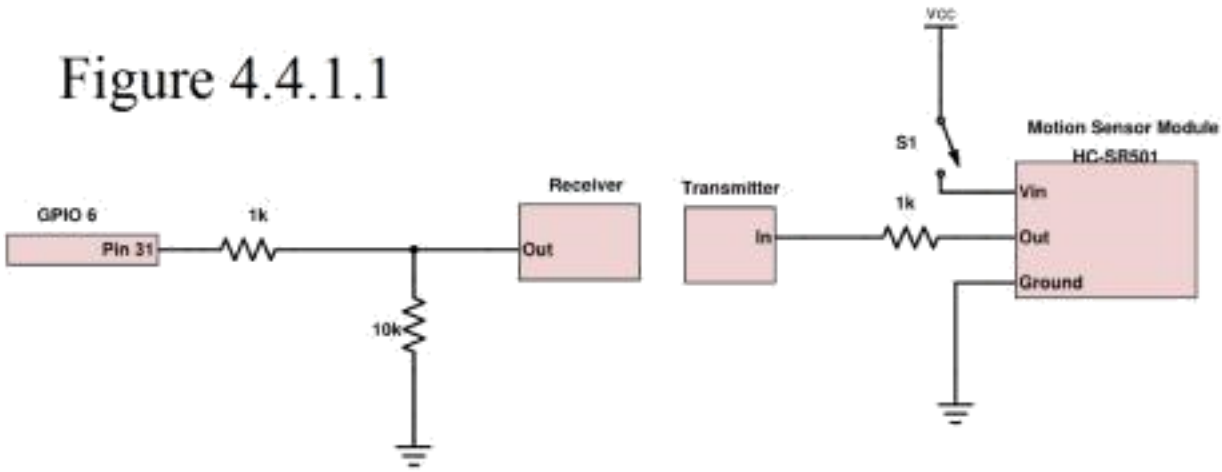
majority of these designs implemented controllers that were already programmed onto the board. Because we are limited on the amount of input and output GPIO pins on the Raspberry Pi, therefore we decided to use a motion sensor device that came pre assembled. The only thing we need to do is design a schematic so we can implement it in our hardware design. We feel as a team that designing a completely new type of motion sensor would be a hassle.

Therefore, the part we will use for the motion sensor will be the HC-SR501. Its operating voltage is anywhere between 5 volts to 20 volts. This means we can use a simple 9 volt battery to supply voltage to it. The reason why this device is a great choice for our project is because the power consumption is extremely low. The device uses around 50 micro amps of static current while operating. If connected a 9 volt barrty with around 100mAh of life, then the unit would last over 2000 hours before needing to be either recharged or replaced. Section 4.3.4 explains more in depth about our battery decision for this part of the project, so we wont need to elaborate anymore on the battery life.

The motion sensor has three different connecting pins ground, power, and output.The schematic of the motion sensor has a built in 10k ohm resistor on the Vin rail. Because of this resitor, this means we wont need to install a current limiting resistor on the positive pole of the battery that is supplying power to the sensor. Although, we will need to implement a pull down resistor on the GPIO pin that we will be reading the digital logic value from the reciever. Additionally, we will also include a current limiting resistor that is attachted to the GPIO for saftey precautions. Pull down resistors and current limiting resistors were further discussed in section 4.4.

We will also be attatching a current limiting resistor to the output of the motion sensor module. The reason for this decision is because the data sheets for the motion sensor did not mention the current value for the output. Therefore, to prevent damage to our transmitter, we will be including this resistor to our schematics. We will be running this type of sensor in the Single Trigger mode. The reason for this is because once an intrusion is detected, we don’t want the motion sensor to keep sending signals to the wireless reciever, thus wasting more power. It will simply detect motion, send a high logic value to the transmitter, and then reset to a low logic value. Because our design is rather simple, we need to implement a way to turn off the motion sensor when the system is not armed. To solve this issue we will be connecting a switch that the user will have to physically turn on or off in order to save energy when the motion sensor is not being used. This switch will be attached inbetween Vcc and the Vin port of the module. To save additional power, this switch will also determine if the transmitter is receiving power or not. Figure 4.4.1.1 illistrates the schematic of our motion sensor design.

Figure 4.4.1.1



4.4.2) FIRE SENSORS

Initially, we as a team wanted to design a flame sensor that will sound an alarm when a fire is present. Because we are compressed with time, we will be asking permission from AtomicMarket to use their fire sensor in order to complete our project on time. Reverse engineering the schematic of the fire sensor reveals to us that the actual ir sensor is an npn transistor. When a fire is detected it will pass current through the base of the transistor. The schematic also uses an LM393 comparator with the output of the transistor connected to one input and the 3.3-volt rail connected to a potentiometer that is connected to the other input. The power two leads on the op amp are connected to the 3.3-volt rail and ground. This will be used to output either a digital high or a digital low. Additionally, since the op-amp is outputting a high or low logic value level, this means we won't be using to read analog voltage readings from the sensor. We will be adjusting the potentiometer to compensate for the sensitivity of the sensor itself.

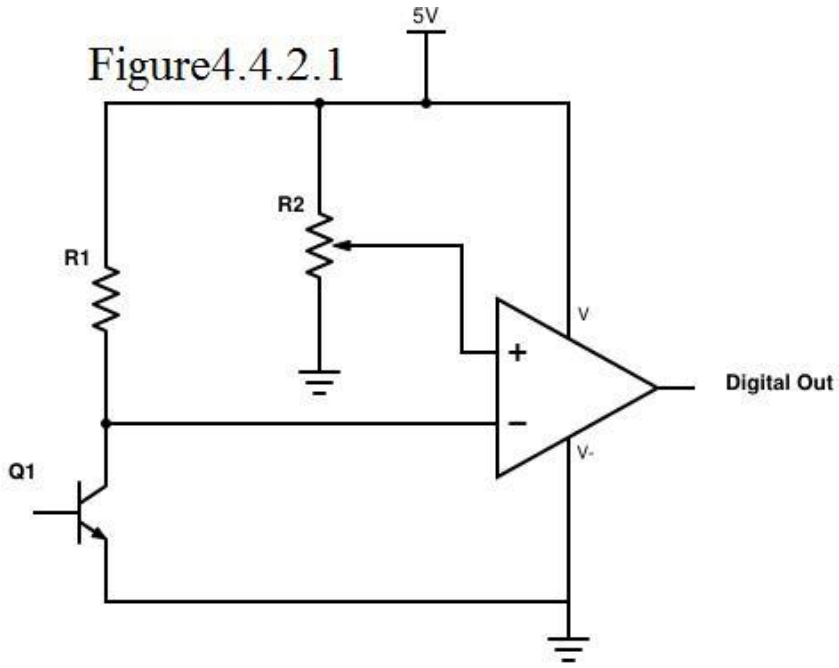
The fire detector has four different connecting pins: ground, power, digital out, and analog out. Since we will be outputting a digital logic value, then we will not be using the analog output pin. The operating voltage of this module is anywhere between 5-3.3 volts. Because we want to use as little power as possible on the 3.3-volt rail, we will be supplying the module with 5 volts from the raspberry pi. The datasheet on this module was very unclear as to what value the voltage will be at the output. Therefore, for safety precautions, we will be implementing a voltage divider at the digital output to step down the voltage from the 5-volt rail to a safe 3.3 volts so the GPIO will not be supplied with too much voltage which could potentially damage the microcontroller.

How these integrated circuit works is that when there is no fire detected, the base will have zero current running through the transistor. If there is no current running through the base, then based off transistor logic, the collector current is also zero. This means that the voltage at the collector where Vin is connected will be Vcc. The comparator will see that Vcc is higher than the voltage at the wiper of the potentiometer and the op-amp will output zero volts. When a fire is detected, the base of the transistor will allow current to flow. This means the collector current will also flow through the transistor and the voltage at the pin that is connected to the collector will be significantly lower. The operational

amplifier will see that the voltage at the output of the transistor is lower than the voltage at the wiper blade of the potentiometer, thus it will output 5 volts. Figure 4.4.2.1 shows a basic schematic of the module itself.

Our test measurements on the output of the module will further tell us if we need to implement a voltage divider or not. Also, we will be implementing a current resistor that is connected to the GPIO pin of the raspberry pi. Current limiting resistors were previously discussed in section 4.4. We will also be adding a pull-down resistor that will prevent the input of the GPIO from floating. Likewise, floating logic values were discussed in depth in section 4.4 as well, therefore we will not need to go in depth about the purpose of them in this section.

Figure4.4.2.1



Finally, our complete schematic will include the module itself along with a voltage divider circuit and a current limiting resistor. We will need to calculate the values of the resistors so that it will drop the voltage to a safe 3.3 volts for the GPIO pin, and also reduce the current going into the GPIO pin to around 5mA. Figure 4.4.2.2 shows the schematic of the voltage divider along with the current limiting resistor and pull down resistor.

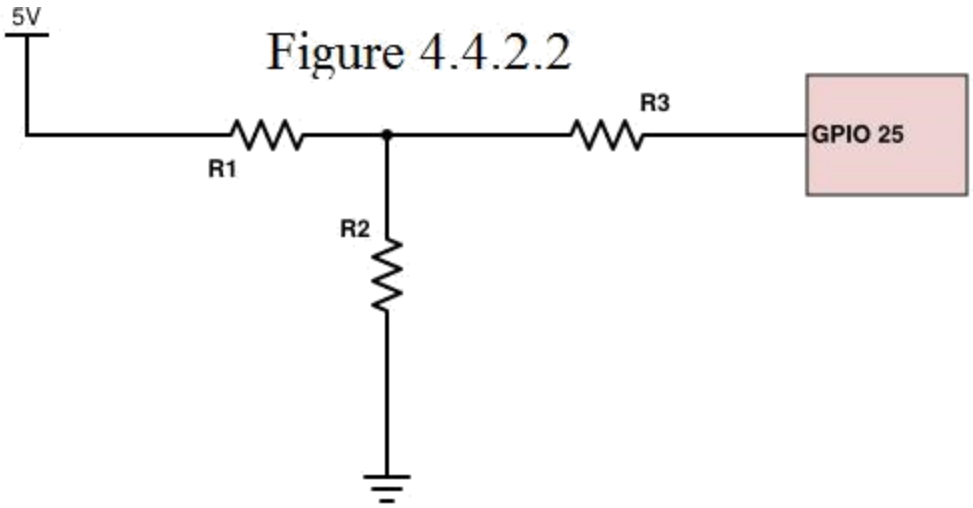


Figure 4.4.2.2

Since the datasheet lists that the output can supply a maximum of 15mA of current, and the node connecting all the resistors together is known to be 3.3 volts, then the resistor value for R1 can be easily solved as 113. Furthermore, R3 and R2 can be easily solved as well using ohms law. Since we want to introduce 5mA of current to the GIPO pin on the raspberry, then R3 will be 660 ohms and R2 will be 330. This will assume that the current leaving the digital output will be 15mA while the GPIO pin will receive 5mA of current while maintaining 3.3 volts as a digital high logic value. These values will alter depending on the ideal values that we will use for the resistors. Finally, figure 4.4.2.3 shows the schematic of our complete design that we will be including for our project.

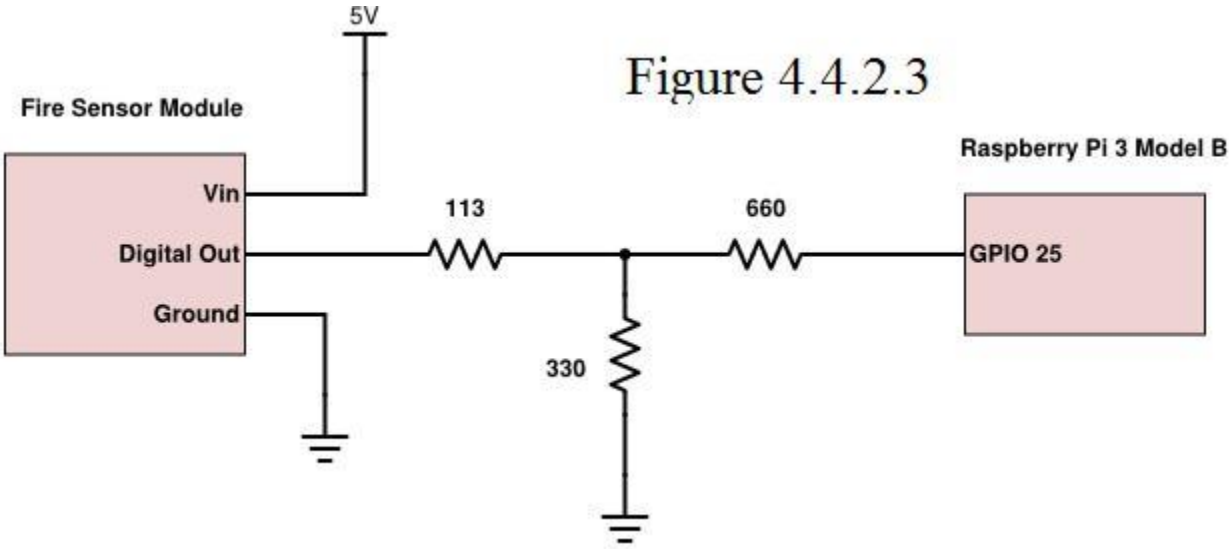


Figure 4.4.2.3

4.4.3) TILT SENSORS

When we are initially designing our circuit to use for preventing any intrusions through the window, whether it be forced open or from smashing the glass, we needed to keep in mind about the sensitivity of this type of sensor. The reason for this is

because we don't want to have a sensor that is too sensitive to where a light breeze or rainfall would set it off. Likewise, we don't want the sensor to not be responsive enough so that it could be easily bypassed by cutting the glass and gently opening the window. This is obviously a worst-case scenario. Never the less, we will be using the type of tilt sensor that operates off of pure mechanics. These types of sensors are relatively cheap, therefore it will keep the cost of our total project down.

The way this type of sensor works is by having a pin wrapped around a thin coil wire. The coiled wire is initially not making contact with the pin that it is around. It will make contact based on how much movement it encounters. Figure 4.4.3.1 illustrates the how the inside of this sensor looks. Because it its purely mechanical, this means it will act like a digital switch that will output logic high when the coil makes contact with the pin, and logical low when it is stationary.

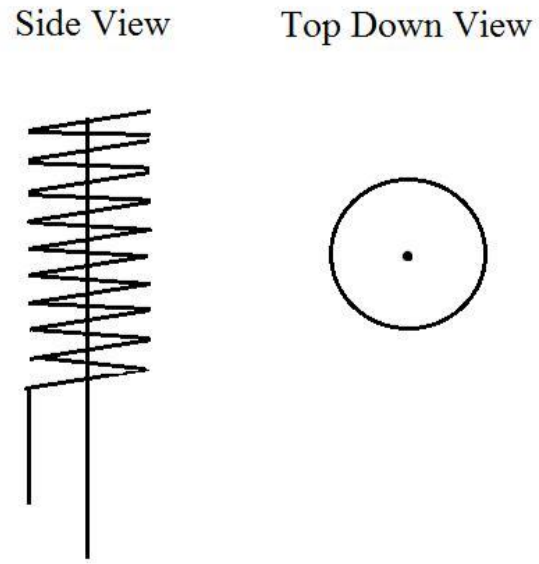


Figure 4.4.3.1

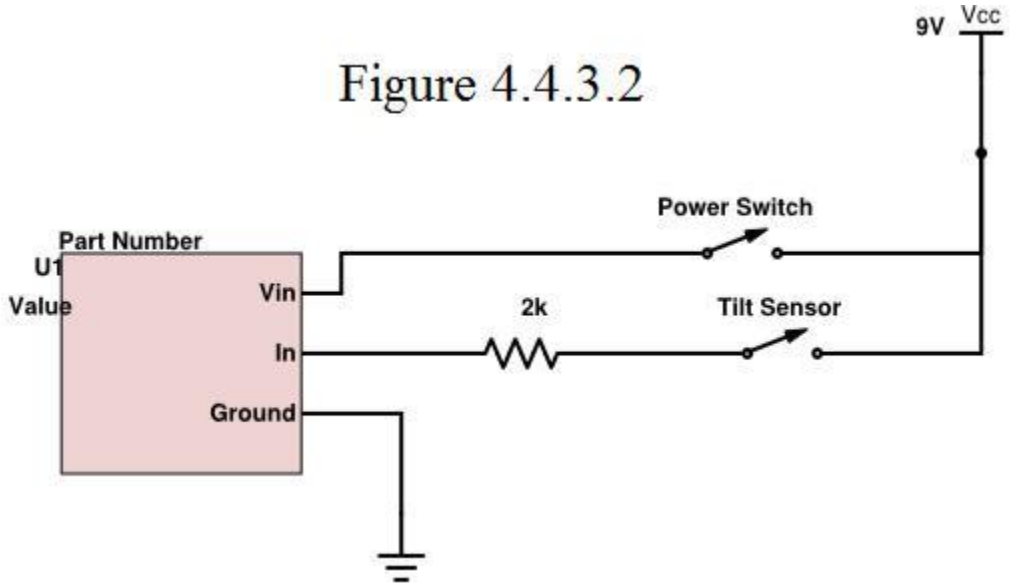
Because the sensor acts as a mechanical switch, and it will only complete the circuit for a short duration of time, then we need to figure out a solution so that our transmitter will be able to output to the receiver exactly when the sensor is tripped. The easiest way that we have decided to solve this problem is by implementing a physical switch that will turn on the transmitter when it needs power, and turn it off when it's not being used. This, of course, will save the user from the battery being drained. The battery life is explained in depth in section 4.3.4, so we will not need to further discuss it in this section.

The specs for this sensor will only allow a maximum of 20mA to pass through this device. Therefore, the resistor we will use for this application will be 2kohms. This will limit the current passing through the device to be roughly 4.5mA. Additionally, because the transmitter input can withstand a maximum of 12 volts, then we will not need a voltage

regulator to step down the voltage from the 9-volt battery. The voltage coming from the battery will be at a safe level due to the specifications that were given on the TX433 transmitter. The way we will hook up the transmitter is by having the power switch in between the 9-volt power supply and the transmitters Vin. Likewise, we will be placing the tilt sensor in between the 9-volt power supply and the 2kohm resistor which is connected to the input port of the transmitter. Finally, the ground pin of the transmitter will be connected to ground.

The way this sensor will work is that when the switch is turned on it will constantly read the voltage from the tilt sensor switch. As soon as an intrusion is detected, the coil in the sensor will make contact with the pin thus completing the circuit and sending a logical high value to the input pin of the transmitter. Since the sensor is connected directly to the transmitter, then we will not need to worry about the short contact time of the coil inside the sensor. Figure 4.4.3.2 illustrates the schematics of our design.

Figure 4.4.3.2



4.4.4) RFID/NFC

Integrating the PN532 to the Raspberry Pi is the next fete. We must figure out what pins are available to be able to add the PN532 to the Pi. The PN532 requires 4 GPIO pins and 1 pin for power and 1 pin for ground. Since the PN532 offers multiple communication methods, the most ideal one is the Serial Peripheral Interface because it is cross-platform. This board is designed to be used by a 3.3V, which the Raspberry Pi supplies so a shifter is not needed. The current that the reader draws are a max of 140 mA. To set up the PN532 you must add a jumper wire to configure the SPI communication by selecting SEL0 to off and SEL1 to off. To wire the PN532 to the Raspberry Pi you must make these connections the 3.3V must connect to the GPIO 1, the SCK must connect to the GPIO 25, MISO connects to GPIO 24, MOSI connects to GPIO 23, SSEL connects to GPIO18 and GND connects to the ground pin which is 6. The Raspberry has built in pins for SPI communication which makes it easy. These connections will use a software SPI

connection that implements digital inputs and outputs pins for the connection. This makes the implementation simple due to the face that SPI is flexible and fast. Once we have come to a consensus setting up the NFC breakout board will not take more than an hour. We need to free the UART port on the pi because it is dedicated to other purposes. Freeing the UART port will help us build the LIBNFC which will be needed for the RFID/NFC reader. Once we build the LIBNFC and update all the necessary changes we will be ready to start programming the NFC chip to be able to arm and disarm the system. The image below shows how to connect the PN532 to a T-cobbler which connected to the Raspberry Pi.

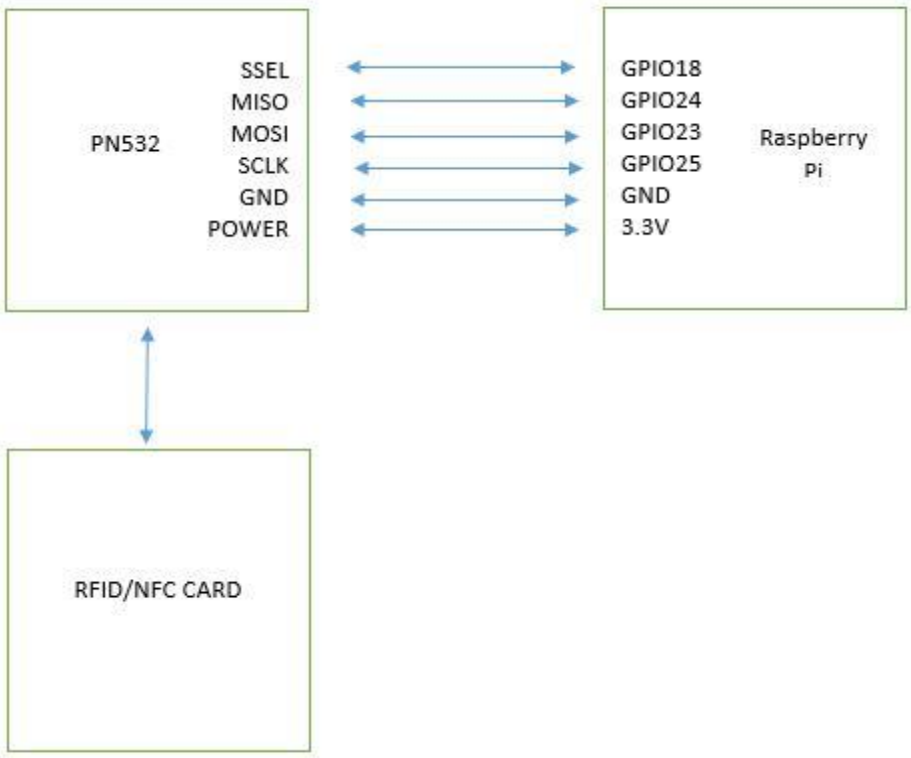


Figure 4.4.4.1 This figure shows the wiring configurations of the PN532 to the Raspberry Pi 3 and how the RFID/NFC Card will communicate with the system.

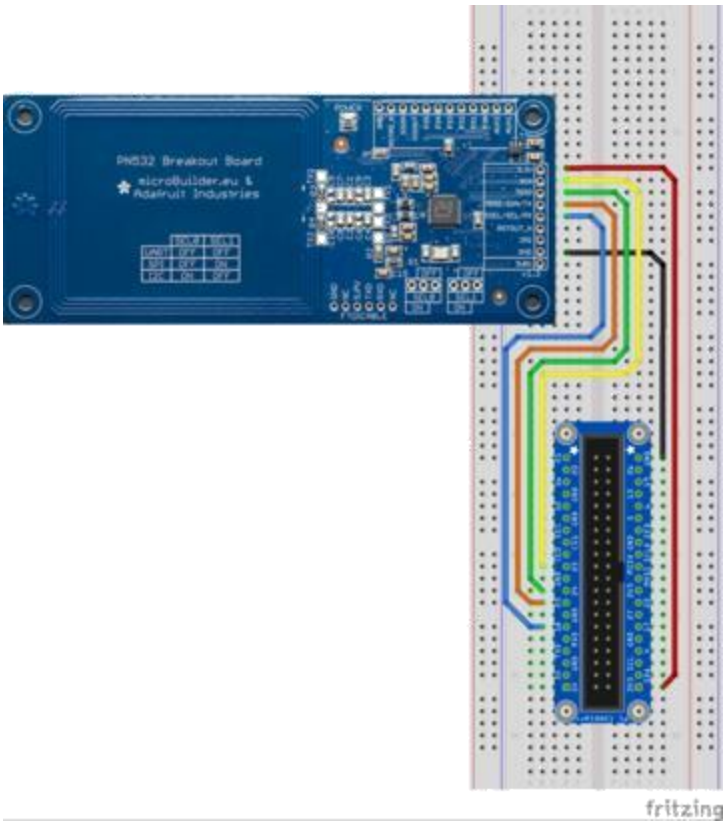


Figure 4.4.4.2 The figure shows the set up for the PN532 to the Raspberry Pi. This work is licensed under the Creative Commons Attribution 3.0 Unported License. To view a copy of this license, visit http://creativecommons.org/licenses/by/3.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA

The image below shows how the RFID card and the PN532 must be in close proximity for it to work and it can only work with certain types of RFID cards.

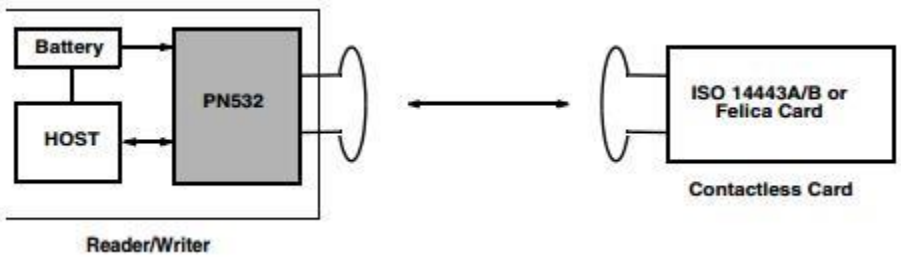


Figure 4.4.4.1 Communication between the RFID reader and an RFID card. Credit to Philips Semiconductors.

4.5) PUSH BUTTONS

As mentioned before, the main issue we have come across when deciding what microcontroller, we were going to use was the amount of input/output pins that each microcontroller had available. Since the push button setup we are going to implement for our main panel uses twelve buttons, then we are going to need at least seven input/output

pins dedicated to only those inputs. As a team, we feel that the raspberry pi development kit would be a perfect microcontroller for this project. Compared to the competitors like the Arduino and the MSP430, it will accommodate all of our requirements in order for us to complete this project.

First off, the button layout will be ordered from 1-9 including 0, *, and #. The main function for the pushbuttons will be interacting with the Graphical User Interface, or GUI. The layout of the buttons will ascend from a top down format. There were a couple of ideas we had in mind to implement the push buttons to not only use very low power, but also so that they don't require a lot of input ports. The first design that we thought about was having each button designated to each port with a pull down resistor attached to it. This was very straight forward, basic, and also low power. The problem with this design is that it will use a total of twelve input ports on the raspberry in order for it to work. We felt like this would not be the most efficient design as possible. Our second idea for tackling this design was to implement a binary code that represented the number being pressed as a visual binary number. If the push button nine was pressed, then the inputs on the microcontroller would read 1001. The microcontroller would see that port three and port zero are both reading high so then the embedded code will know that the push button nine is being pressed.

Instead of implementing the previous two designs, we are going to design a push button matrix that will use a total of seven general purpose input/output pins. The reason for this decision is because not only is this design efficient at using the least amount of GPIO ports on the Raspberry, it is also very simple to design the schematic compared to the second example as mentioned above. Four of the pins will be dedicated to the rows on the matrix, and they will be set up as inputs. The other three GPIO pins will be dedicated to the columns, and they will be setup as outputs on the microcontroller. All in all, we will be using a 3x4 pushbutton design in this project.

The pushbuttons will be connected in between one input pin and one output pin. When a button is pressed a short will be created between the two pins thus sending a signal from the output to the input. Because we are dealing with digital logic, we will need to take into consideration the floating logic signal. For example, if we were to connect a switch between the two pins, then while the button is not being pressed, the input pin would not have a voltage reference. This would "confuse" the microcontroller as to what state the input pin is, and our reading from the program would be inaccurate. To solve this issue, we can either implement a pull up resistor, or a pull down resistor. An analogy we can use here to describe why we are implementing these types of resistor designs is space flight. If you are flying through space without any kind of physical reference, then you would have no idea what direction you are going. This design would basically give the input pin a voltage reference so it knows if it is either high or low.

As mentioned previously, we have two choices of resistors; a pull up, or a pull down resistor. The way a pull up resistor works is it connected to the 3.3 volt rail of the micro controller and also in between the input pin and the ground. This means that the input will always read high until the button is pressed, then the current will negate the input and run through the short that the switch created. A pull down resistor (Figure 4.4.1) is the opposite. Instead of connecting the 3.3-volt rail, we would connect the ground to the resistor and also

in between the input and the 3.3 rail. The input would always read low until the switch is pressed which it would then read high.

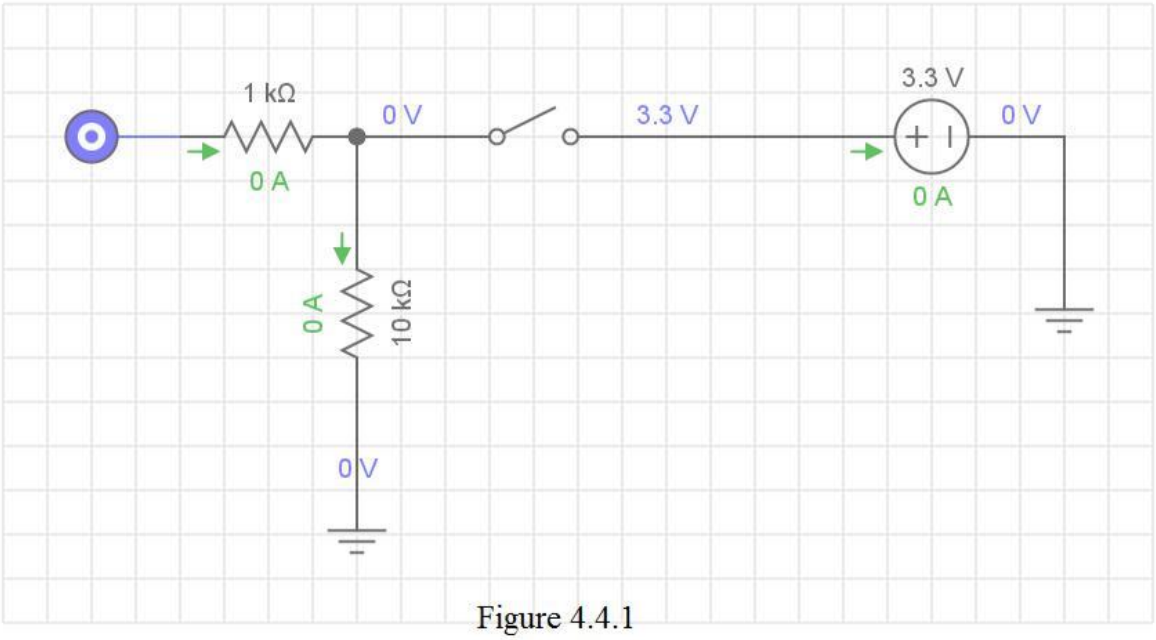


Figure 4.4.1

Because a pull up resistor design would constantly keep current running through the inputs, we decided to apply the pull down resistors in our project. This would mean only one input is drawing current instead of a maximum of four inputs drawing current. The reason for this decision is not only does a pull down resistor look physically appealing to code with (high=pressed, low=not pressed) but it saves us from using power that we could dedicate elsewhere in the project. If we were to use the pull up resistors, then this would mean that a maximum of six inputs will be drawing 5mA of current; four of the inputs drawing current from the 3.3V rail, and the other two outputs are set to high. If each pin was using a total of 5mA then we would be using a total of 30mA which is over half of the maximum current draw that the raspberry pi has listed in its data sheet for the GPIO pins and 3.3V rail.

Not only will we be implementing a pull down resistor, but we will also include a current limiting resistor in our design for safety precautions. This current limiting resistor will be connected in series with the GPIO output pins. The reason being is that if we were to accidently press two buttons at once then this would create a short between the two outputs. This could be disastrous for the microcontroller in that there is no resistance in between the two pins that are being connected.

Another alternative that we considered was implementing four current limiting resistors that are in series with the four inputs. The idea here was keep the current at a safe level when more than three buttons were pressed at once. This idea was kind of redundant once we computed the max current being produced by all three outputs from the raspberry.

The total output current from all three GPIO output pins into a single input was 9.9mA, 3.3mA for each output pin. We feel like this current output is fine since it's only using twenty percent of our total max current for the microcontroller.

Additionally, if we were to have another 1k ohm resistor in series with the input pin, then this would drop the voltage below the digital high reading. The voltage at the node connected to the 10k ohm resistor and the two 1k ohm resistors would be 1.57 volts. Since digital logic reads only a 1 or a 0, then the micro controller might be confused as to which state the pin is at. It could read that the state is high or low since the voltage is directly in between 3.3 volts and 0 volts. This is another reason as to why we didn't implement another current limiting resistor. Research has also revealed to us that the threshold maximum digital low voltage is 0.8 volts and the minimum threshold digital high voltage is 2 volts. If we implemented this design, then our predictions were right. The microcontroller would see that the node is at 1.56 volts, so it would get confused as to which logic state it is at. Figure 4.4.2 Shows the complete schematic of our push button design.

The process at which the push button matrix will operate is called "matrix scanning." The microprocessor will "scan" the four inputs by using an array of values in our program that are dedicated to the output pins. Above is a truth table that gives the logic values of the input/output pins. We will assign the output pins three different signals, 100, 010, and finally 001. The way the program will tell what button is pressed is by which signal will drive a single row bit to low. The microcontroller will continuously send the three different output signals (100,010,001) until the specifications are met. For example, if we press the button that represents the numeric value five, then the output that will drive the row to high would be 010 and the input would then read 0100(reading top row on the left, and the bottom most row on the right.) If we the outputs were 100 or 001, then since the outputs that are set to a logical high voltage, then the input logic states would be 0000. Since this is a pull down resistor configuration then high means the button is being pressed, and low means the button is not being pressed. In our opinion, this configuration actually is visually pleasing to work with when programming. Table 4.4.1 lists the truth values.

Calculations:

One of our goals for this part of our project was to use as little power as possible. Since we have three resistors connected in parallel, we have to be sure that the current won't be greater than the total output a GPIO pin can produce which is 50mA. This is a total of all the combined GPIO pins which output 3.3 volts. If one button is being pressed, causing the input and output GPIO pins to be connected together through a single 1k ohm resistor, then our calculations show us that the total current passing through a single output pin is 3.3mA. The total power being used by this 1 pin is 0.1089 Watts. So this configuration is uses very little power. Ideally this is what we would expect from a simple resistor circuit that is only reading the state of an input pin. We could replace the 1k ohm current limiting resistor with a 2.2k ohm resistor to reduce the power consumption even more, but we feel like we would be splitting hairs at that point since the power consumption would be 0.049 Watts.

Furthermore, when three buttons that are associated with three different outputs are being pressed, then the total current being drawn is 9.9mA. The total power being dispersed

is found using simple ohms law, which is 0.099*3.3. The total power from three outputs to one input is 0.3267 Watts. This means that even when multiple buttons are being pressed at once, the circuit uses very little power. Because this button configuration circuit is not using a lot of power, this means we have more power to use on the other parts of our project. Because we have kept the power consumption under 0.5 Watts, then this means we have completed one of the two goals that we have previously mentioned above.

Figure 4.4.2

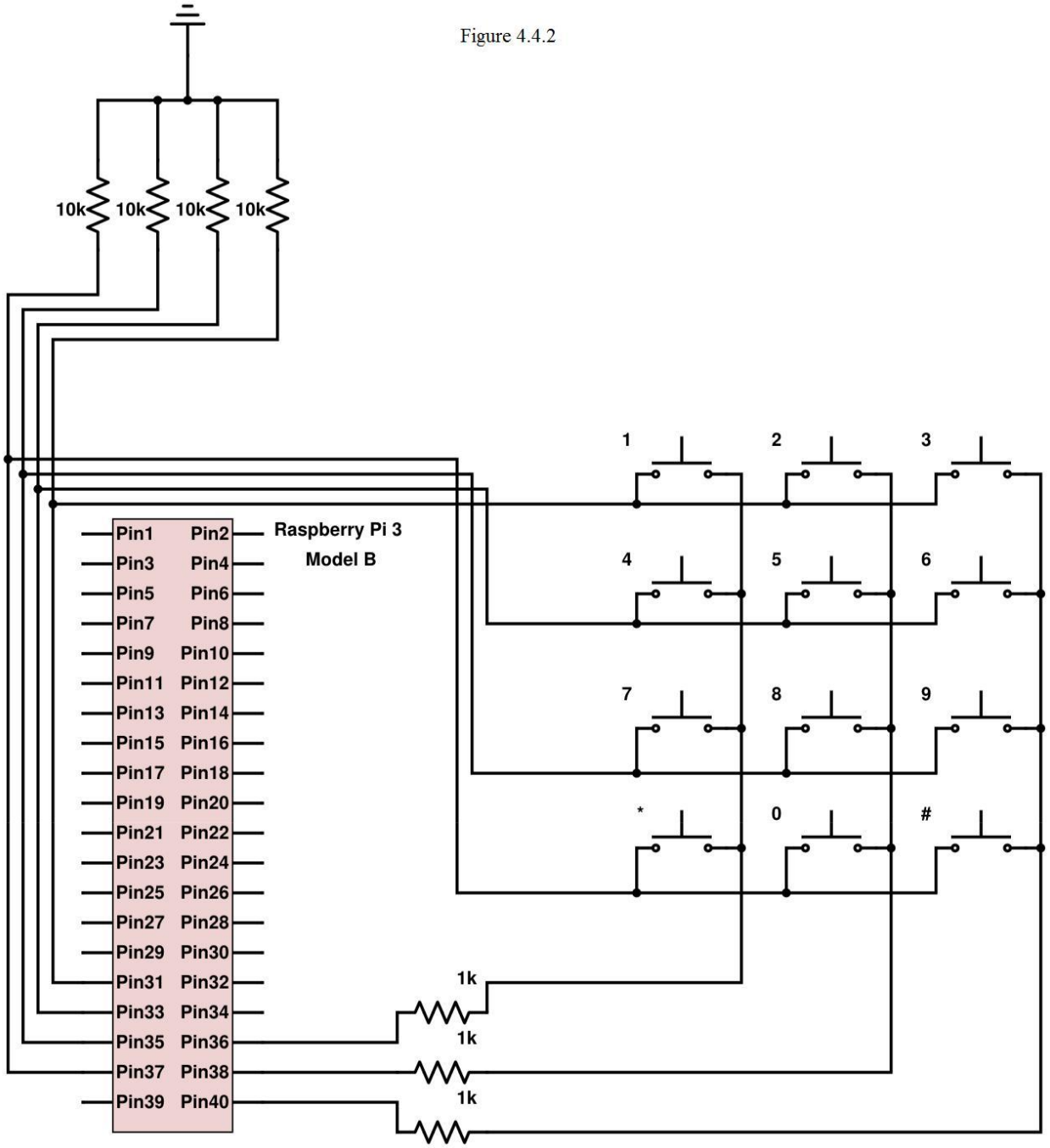


Table 4.4.1

Button	GPIO 6	GPIO 13	GPIO 19	GPIO 26	GPIO 16	GPIO 20	GPIO 21
1	1	0	0	0	1	0	0
2	1	0	0	0	0	1	0
3	1	0	0	0	0	0	1
4	0	1	0	0	1	0	0
5	0	1	0	0	0	1	0
6	0	1	0	0	0	0	1
7	0	0	1	0	1	0	0
8	0	0	1	0	0	1	0
9	0	0	1	0	0	0	1
*	0	0	0	1	1	0	0
0	0	0	0	1	0	1	0
#	0	0	0	1	0	0	1

4.6) DOOR CHIMES

As a whole, we feel like designing the switches that will activate the door chimes when a door opens is going to be the less tedious task of all the hardware design. The reason being is because the design is basically a switch connected to an input, and whenever that input digital logic value changes, then the microcontroller will send a signal to our speaker to let the user know a door was open. Additionally, when the system is armed and a door is opened then the microcontroller will consequently sound the alarm at an attempt to scare off any intruders.

There are a couple of switch designs that we have researched. The first design was to implement a physical pin that connects both the door and the door frame. When the door is closed, then the pin from the door makes contact with the socket on the frame, thus completing the circuit. The main issue with this implementation is that the circuit will always be completed since the door will always be closed throughout the day. If the circuit is always completed, then current will always be flowing through our circuit, thus wasting power.

Another issue with the door acting as the physical switch is that it could be damaged from the excess amount of opening and closing. The pin connected to the door could break off which then the system would assume a door was always opened. This would, unfortunately, lead to a huge security hole in our system since the user would have to disable the door chimes which would allow an intruder to simply walk right through an entrance.

Our second idea for this switch design system is reed switches. A reed switch works by introducing a magnetic field to the element which will either open or close depending on the switch design itself. This will completely eliminate any physical damage from opening, closing, or slamming a door. The only problem that we will have to keep in mind is finding a way to conceal the device into the door to make this system more physically appealing.

In order to consider designing these switches, we need to research how they operate. Our results concluded that there are two types of reed switches, normally closed, and

Normally Closed

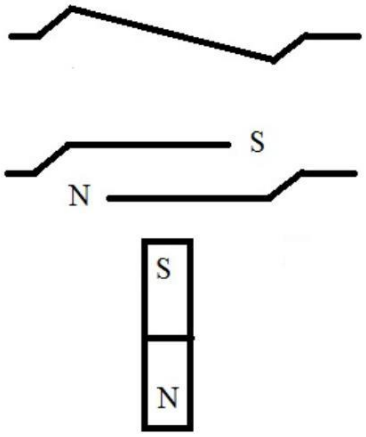


Figure 4.5.2

Normally Open

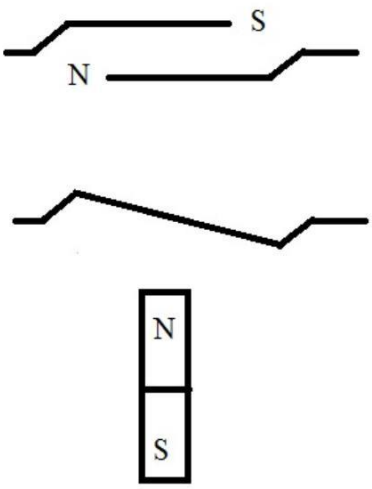


Figure 4.5.1

normally open. The names that are given to these two types of reed switches speak for themselves. A normally open reed switch is always open until a magnet is introduced thus snapping the two blades of the switch together. The way a normally open reed switch works is one of the “blades” is a magnetic north pole, while the other is a magnetic south pole. If you were to introduce another north pole to the blade that is a north pole, then the blade would repel as the other would attract, thus making contact and completing the circuit. Figure 4.5.1 better demonstrates the function of this magnetic switch.

The normally closed (Figure 4.5.2) switch operates the opposite of how a normally open switch would. The two blades are initially making contact until a magnet is present which they would then separate. The pole convention of these two blades remains the same as the normally open reed switch. The difference is once a north pole of the magnet gets close enough to the south pole of the blade, then the south pole will attract to the magnet and the north pole blade will repel away from the magnet, thus opening the circuit.

If we were to implement these types of switches, then we would have to decide what the best physical arrangement would be for our design. Embedding the switch into the door and the magnet into the door frame would not be an ideal configuration for this project. Reason being is that we are not enabling the switch voltage levels to communicate with the Raspberry Pi wirelessly. This would mean that not only would the wires be exposed and therefore enable an intruder to tamper with them, but it would also mean that part of the circuit is exposed to moving physical parts, such as the door hinge. If we simply reversed the configuration such that the magnet is embedded into the door and the circuitry is installed in the door frame, then we wouldn't have to worry about wires being snipped or pinched from moving metal parts. We could then run the wires from the door frame to the inside where the main security system's panel is installed.

Since we want to avoid innovating physical moving parts into our design, then a reed switch will be the element which we will use to read if our door is open or closed. Although both of the switches are not equally ideal for our design. A normally open reed switch only closes when the magnet is near it. In this case, the switch will be connected to the door frame, and the magnet will be connected to the door. Since the door is shut for the majority of the day then the switch will be closed. This isn't the most efficient way of implementing our switch because it will allow current constantly flow through the circuit. Consequently, this would mean more power would be dedicated to the door chimes alone that we could otherwise use somewhere else.

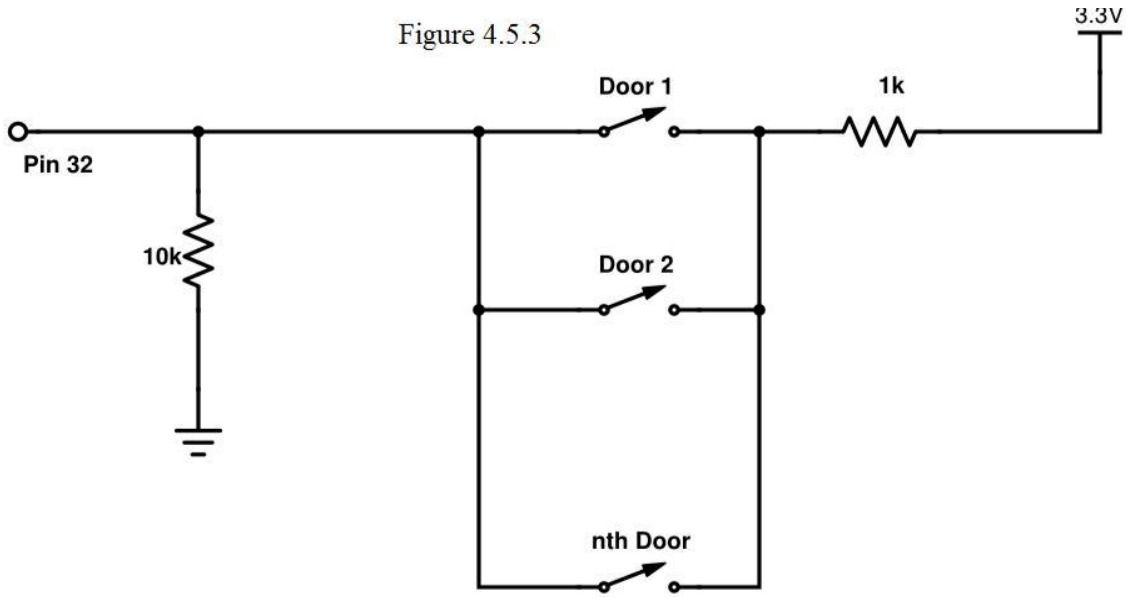
Additionally, since the GPIO pins on the raspberry can't exceed 50mA of current, therefore we also want to be careful in setting the current values for worst-case scenario situations. Let's assume that every single button is pressed, along with the LCD being displayed and each sensor is being triggered. We would have to make sure that every single GPIO ports current's sum does not exceed 50mA or it could damage our Raspberry Pi. Even though assuming every single button is pressed is being a little too cautious, we feel like we have to prepare for currents exceeding 50mA.

Firstly, we need to decide if we want a pull-up or a pull-down resistor configuration. Section 4.4 goes into more depth about what these terms mean. Since we want to use as little power as possible, then we will be implementing a pull-down resistor configuration with a 10k ohm value. We will be using a total of two switches that will be designated to two doors. These switches will be in parallel with each other. The reason for this is because each switch will be connected to a single GPIO pin. It would be extremely inefficient to design the switches to be connected to multiple GPIO pins unless we wanted to include a system that will notify you which door is open. This isn't the case for our design. We simply want the system to chime when a door is open, whichever door it may be.

On one end of the Switches will be the pull-down resistor. The other end of the parallel configuration will be the 3.3-volt rail which is connected to a 1k ohm current limiting resistor. Again, the current limiting resistor was discussed in depth in section 4.5, so we won't need to further explain the use for it in this design. The complete system will operate by the GPIO pin reading low (door closed.) Once the door is opened and the magnet moves away from the reed switch, then switch will snap shut and current will be able to flow through the switch and into the GPIO pin, therefore either sounding the alarm or sounding the door chime. This GPIO pin will be set as an input in the raspberry so it can

read logical voltage values accordingly. If we take a closer look as to why we are connecting each switch in parallel, then we will see that no matter how many doors are open at once, the current for the GPIO pin will always remain the same. Figure 4.5.3 illustrates our reed switch schematic design.

Figure 4.5.3



4.7) LED

As previously mentioned, our main issue when working with the GPIO pins on the raspberry pie is that they can only output a total current of 50mA combined. Since the LED's require at least 20mA of current to turn on, then we will need to figure out a way to design them so that they not only use a lot less current, but we can also control them using the GPIO pins so the user will see that the system is armed or not just by looking at the different color of the led that is being illuminated.

We can solve this issue by implementing a transistor. By connecting the GIPO output pin to the base of the transistor, the led and the 5-volt rail to the collector, and the emitter to ground, then we will have efficiently solved our problem. First off, we will have to make the transistor operate in saturation mode in order for it to act like a switch. To do this we will need to initially assume that Ic<Ib*beta. If this statement holds true then the transistor will operate in saturation mode. Before we start setting current values we will need to test the brightness of our led to assure it is not blinding the user when they are looking at it. Our test has revealed that 20mA is far too bright. After some resistor adjustments, we have concluded that around 2.5mA is when the LED gives off just enough lumens to where it's not blinding to the user. This is what we will set our Ic current as then we will solve for our resistors. Because calculated resistor values are not ideal in the real world, we will have to alter our resistor values to the closest values that are given to us and then calculate the current values again to assure accuracy. Using Kirchoff's voltage law around the collector and emitter loop of the transistor for the red LED, we are left with this equation, 5V= R*2.5mA+0.2+2. Solving for R gives us a resistance of 1120 ohms. Because

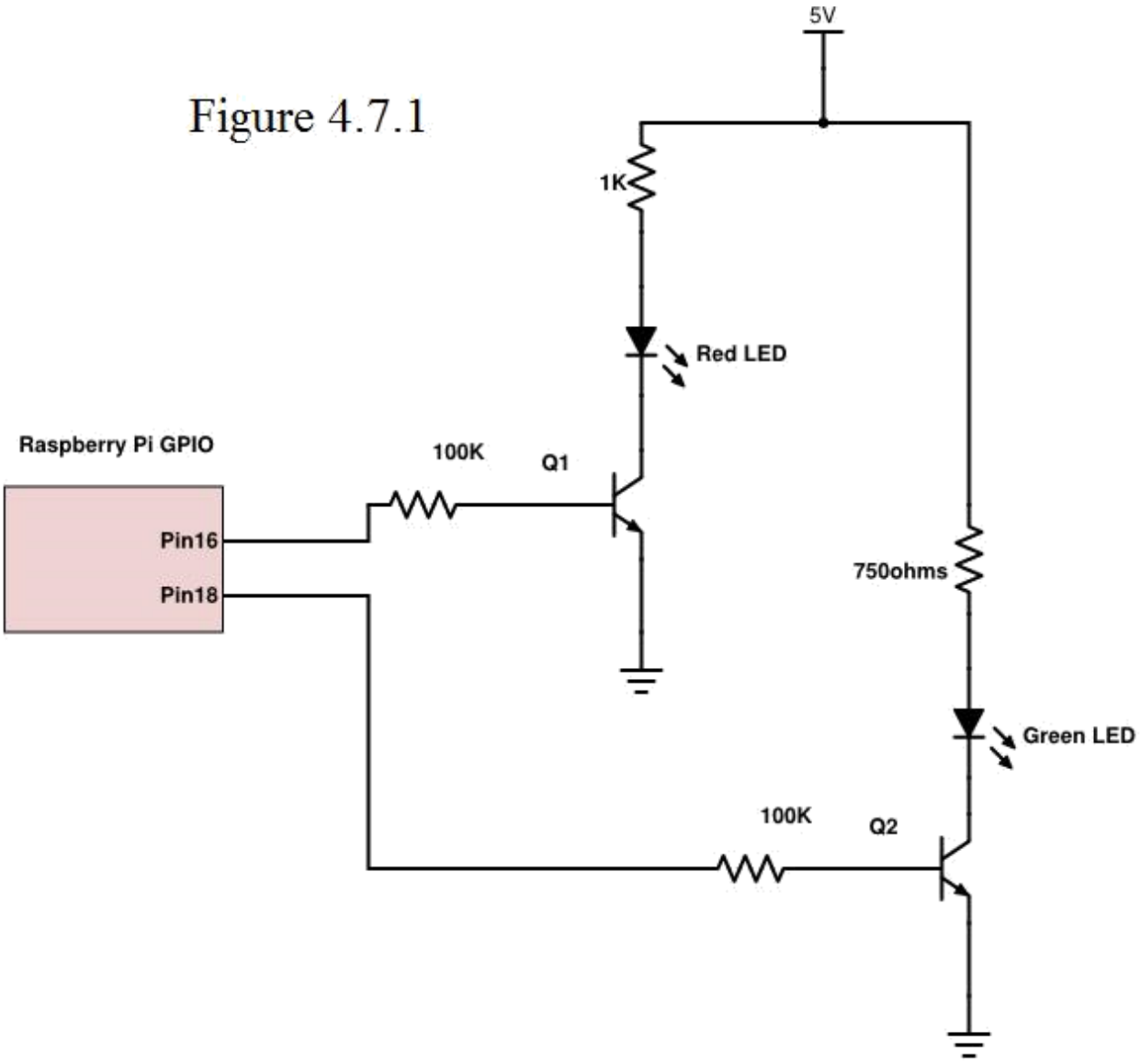
that resistor does not exist in the real world, we will be setting the collector resistor to a value of 1kohms. If we solve for the current this time using the same equation we can conclude that the collector current is equal to 2.8mA which is perfect.

Because Ic is equal to 2.8mA, then we need to make sure beta*Ib is greater than Ic. Since we are using the 2n2222 transistor, then we will be setting beta to the value of 150. Solving the inequality tells us that Ib > 16.6667 microamps. To make calculations simpler we will be setting our base current equal to 20uA. Next, we will use KVL to solve for the base resistor value. Using KVL in the loop of the base and emitter gives us this equation, 3.3= 20uA*R+0.6. Solving the equation for R gives us a value of 135kohms. Again, we do not have this resistor value so we will be using a resistor value of 100kohms. Since the resistance is less than our calculated resistance, we can assure that Ib will still be big enough to make the resistor operate in saturation mode. Solving the equation again using the 100kohm resistor gives us a base current of 27uA. To double check our calculations, we will be referring back to our original statement that is Ic<beta*Ib. Plugging in the values will verify that indeed 2.8mA<150*27uA.

The turn-on voltage for the green LED is significantly more than the red LED. Because of this, we are going to obtain different resistance values. The datasheet has labeled the green LED having a turn-on voltage of 3V. Solving for the current value of 2.5 mA will give us a resistor value of 720ohms which is perfect because we have a resistor value of 750ohms. Plugging in that resistor value for the KVL loop around the collector and emitter will give us a value of 2.4mA. Based off these calculations, this means that the base current should be greater than 16uA. Again, we will set the base current to be 20uA which will put the transistor in saturation mode. Because the base current is the same and we are using the same transistor as we did on the red LED, then we can conclude that the base current is the same for both of the LEDs which is 27uA with a 100kohm resistor attached to the base.

Finally, because our base of the 2n2222 transistor is connected to the GPIO pin that will be turning the led on and off, we have therefore eliminated our problem that we have been faced with over and over throughout this project; and that’s that the GPIO pins can’t output a combined maximum current of 50mA. Since a single LED will only be on at a time, then the current draw from a single output pin of the GPIO to activate our LEDs will be 27uA. In conclusion, our circuit diagram for these two LEDs is illustrated in figure 4.7.1.

Figure 4.7.1


5V
1K
Red LED
Raspberry Pi GPIO
100K
Q1
Pin16
Pin18
750ohms
Green LED
100K
Q2

4.8 PIN MAPPING

Below is a final mapping of all pins used within our project. This relates to the GPIO pins on our Raspberry Pi microcontroller. It can be seen we only have three unused pins when it comes to our deisign.

Pins/Use(DESIGNATE IF INPUT OR OUT)	**Pins/Use(DESIGNATE IF INPUT OR OUT)**
1 3.3V	**2 5v**
3 LCD	**4 5v**
5 LCD	**6 Ground**
7 LCD	**8 Not Used**
9 Ground	**10 Not Used**
11 LCD-Input	**12** LCD-Input
13 LCD-Input	**14 Ground**
15 LCD-Input	**16** Red LED-Output
17 3.3V	**18** Green LED-Output
19 SPI_MOSI	**20 Ground**
21 SPI_MISO	**22 Not Used**
23 SPI_SCLK	**24** SPIO_CE0_N
25 Ground	**26** SPIO_CE1_N
27 RFID-Input	**28** RFID-Input
29 Motion/Tilt Sensor Receiver-Input	**30 Ground**
31 Row 1 (Push buttons)-Input	**32** Door Chimes-Input
33 Row 2 (Push buttons)-Input	**34 Ground**
35 Row 3 (Push buttons)-Input	**36** Column 1 (Push buttons)-Output
37 Row 4 (Push buttons)-Input	**38** Column 2 (Push buttons)-Output
39 Ground	**40** Column 3 (Push buttons)-Output

5) SOFTWARE DESIGN

5.1) STANDARDS AND SOFTWARE CONSTRAINTS

5.1.1 PUSH BUTTON CONSTRAINTS

One constraint that we forgot to keep in mind while programming our push buttons was the capacitors that are attached to the GPIO pins. The reason for this issue is that the capacitors will take time to charge and discharge. If we neglect these changes in voltages on the GPIO pins, then we will obtain inaccurate readings from our input pins. The most important factor for us to keep in mind while coding our program is the natural response of the capacitor and also how fast our microcontroller can execute one line of code. If we set an output to be high and try to read an input the next line, then the capacitor might not have fully charged thus we will obtain an invalid digital reading.

This issue became apparent when we were testing our code and we started to receive wrong values for the input. After a little research on the schematics of the Raspberry, we concluded that the capacitors were the reason for our faulty readings. This set us back two days in our design because the data sheets and the schematics for the microcontroller were a tedious process to find. After some research we finally were able to isolate the issue and make adjustments accordingly.

To solve this issue, we simply inserted a time delay function before each reading on the input. This assured us that the capacitor was either fully charged or discharged when storing the digital logic value from the input. Another issue we have come across when setting the delay values is the microcontroller's processing time. If we set the delay values too low, then the processing time becomes too long for reading a simple push button. If we set the delay too short, then the capacitor might not have enough time to adjust the voltage on the input.

In order to figure out how long it takes for the capacitor to fully charge we need to solve for its time constant. The Raspberry Pi schematics reveal to us that the 3.3-volt rail has a 100nf capacitor attached to it. This means if we use a 10k ohm pull down resistor, then the capacitor time constant will be 1ms. To compensate for this delay, we decided to insert a delay of 5ms to give the code a proper amount of time to get a valid reading from the input.

Once we inserted the time delay function into our code, then everything ran smoothly. We were receiving correct values for our pushbuttons. Even though this was a very small complication, it put us behind slightly in our design because of the lack of information we could find about the schematics of the microcontroller.

Additionally, another constraint that we have run into while programming is the debugging software, or better put the lack of the debugging software in the raspberry pi. Because there is no debugging software included in the raspberry, we had to understand our code inside and out. If we made one mistake, there was no way to step through our code to see where the issue might be. This is becoming extremely frustrating for us.

5.1.2) GUI

The LCD will be the gap between the user and the system. We will set up a very easy yet beautiful interface for our users to interact with. We have choices between a few GUI code for Python like: Kivy, PyQt, PyGUI, TkInter, wxPython etc… With a 480-320 resolution the touchscreen will display a color menu with icons. If using the PyQt, In QtCreator, just save your GUI. It will update the MainWindow.ui file. In a terminal (or command shell for Windows), we're going to make use of the pyuic5 utility. We can name the result anything we want; I will keep the name mostly the same. Each time we update the GUI in QtCreator, we will want to run pyuic5 to update the "auto.py" file.

Functional description -The interface will be simple for ease of use. It will consist of basically two main icons at the front screen like: Menu and Setup etc. The setup option is for the user setup the different mode they can change the system into. Inside setup, the user will be able to create pins, update pins or remove pins. The system will allow up to three pins or accounts for administration purposes; only people with the administrator accounts/pins will be able to configure or make changes to the system's display. At startup the user will be prompted a message to register or create at least one account to administrate the system, but later on they can add more account up to three. Also in the setup the user will be able to able to select the screen color for the display. Inside the menu will be different options to set the sensors and alarm sounds. When going into the menu, the user will have choices to disarm or arm a particular or all the sensors. If the user chooses to disarm a sensor a warning message will appear with description explaining the risks of doing so and asking to confirm the disarming. The user will be able to manually test the sensor to make sure they are always functional by clicking the On/Off button. The menu will give the user options to lower or higher the alarm sounds. Inside the menu there will also be options like reset, mode etc. The reset button will be used to reset some features in the system in case the user cannot undo a feature manually or if the system is freezes. The mode option will be used to set the sensors to a specific mode. For some sensors it may be temperature, distance, level of noise, level of smoke etc. There will be four arrow navigation panel to allow the user to navigate with the menu. Once inside the menu there will be a title (the name) of the selected option on top, on the bottom left there will be a back button to go back to the previous step once you go back there will be a forward to go forward; on the right side there will be an exit button if the user decides to go the main menu; also there will be an Ok/accept button to confirm the user's choice.

5.2) LCD AND INTERACTIONS

5.2.1) PUSH BUTTON INTERACTIONS

To design the program that we want to use for the microcontroller to be able to communicate to the LCD, we needed to figure out how we are going to implement a scanning algorithm that will set each output high at different intervals to determine which input will also read high. One difficulty we have come across while writing this program was the header file that we used in the code. We had to download and research the bcm2835 header file that we will be using in order to write to the input and output pins. Once we figured out how each function operated, then the coding process became very simple.

First, we started out by declaring our constants, ROW1, ROW2, ROW3, ROW4, COL1, COL2, COL3. We set each of these to the pins that they represent. The header file has its own constants that represent the GPIO pins. For example, GPIO 35 would be RPI_V2_GPIO_P1_35. After digging through the header file, we have found that these values are equal to the GPIO number, and the label is the physical pin number. So RPI_V2_GPIO_P1_35 as you can see has a label of 35. This means that it's the physical pin 35 and it is equal to 19 since that is the GPIO pin connected to the physical pin number 35.

After we declared all of our constants we then began by declaring our variables. The table to the right lists all of the variable and their usage. The next step in this process was to set each of our pins to either outputs or inputs. To do this, the header file has a function called bcm2835_gpio_fsel(x,y) where x represents the pin you are addressing and y represents the value you would like to set it as. Once again, we looked up in the header file which value will set the gpio pins and we discovered that you can either use the header file's constant that came with it, or we could use a 1 which sets the pin to an output, or a 0 which sets the pin to an input.

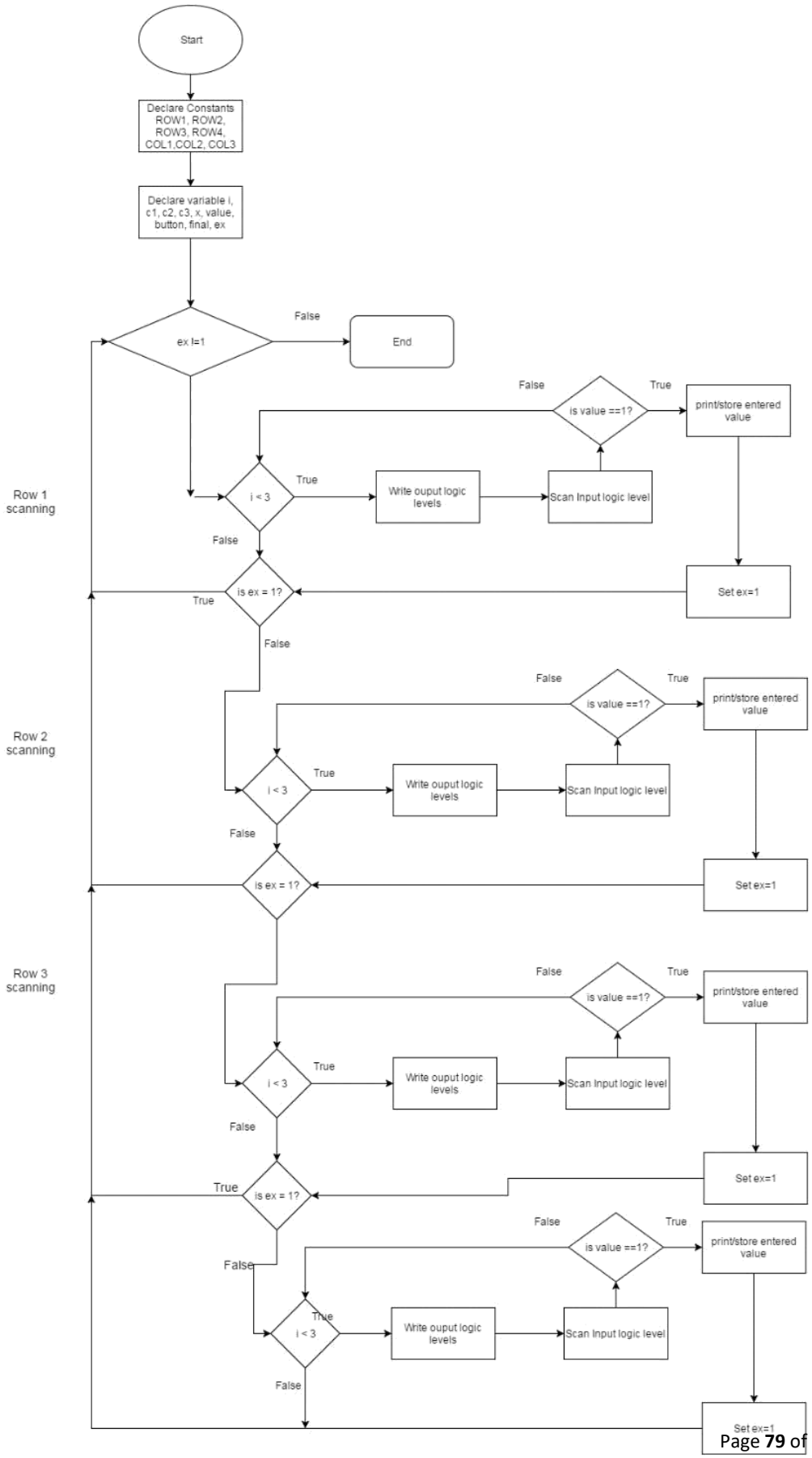
Variable	Usage
i	Loop Counter
x	Button Counter
c1[]	Array values for column 1
C2[]	Array values for column 2
C3[]	Array values for column 2
Value	Input pin logic state
Button[]	Array values for printing to the lcd

We began the code by using a while loop that will terminate while value=1. In the while loop we have four for loops that each have a counter (i) that will terminate when i is greater than three. At the beginning of the while loop we initialized x to zero. This will be the offset for the button array that we will use to send a value to the microcontroller so it will know which button is being pressed. For example, if x=4 then it will point to the 5^{th} element in the array which will represent the numeric value of five on the LCD.

The three for loops start by initializing the loop counter to zero, sets the condition to i is less than 3, and then increments i by one for each loop. Each for loop represents the different rows on the pushbutton design that we will be testing: for loop one=row one, for loop two = row two, and for loop three = row three. First off, the loop will set each output to the corresponding value in the column arrays. The array values are c1[3]= {1,0,0}, c2[3]= {0,1,0}, and int c3[3]= {0,0,1}. The function that came with the header file we will use to set the pins to either high or low is bcm2835_gpio_write(x,y), where x represents the pin number, and y represents the value you would like to set it to (1 being high and 0 being low.) We will be using the counter from the for loop as the offset of the column arrays. The first loop will set column 1 to high and the rest to low. Once all the output pins are set to their logical voltage levels, then we will store the input logic voltage in the variable called "value." The function we will use to store the input value is bcm2835_gpio_lev(x) where x represents the GPIO pin you we would like to store. The line of code would look like this, value = bcm2835_gpio_lev(ROW1). This will set "value" to either a one or a zero.

In order to check the condition of the variable "value" we included an if statement in the for loops. The if statements have the condition "value==1" which checks if a button

is being pressed. If the button is being pressed, it will offset the button array with the value of x and then send it to the microcontroller, thus exiting the scanning function. If the button is not being pressed, then it will increment x and continue until the for loop has finished. All three for loops are identical except that the second, third and fourth for loop will be checking the logic states of ROW2, ROW3, and ROW4 respectively. The variable "x" will be incremented after each for loop. After the four for loops have finished and a button has not been pressed, the program will then continue to run this routine until the user has pressed a button. This flowchart will help us better visually understand how the code is being processed, and how it runs.


Start
Declare Constants ROW1, ROW2, ROW3, ROW4, COL1,COL2, COL3
Declare variable i, c1, c2, c3, x, value, button, final, ex
ex !=1
False
End
Row 1 scanning
False
is value ==1?
True
print/store entered value
i < 3
True
Write ouput logic levels
Scan Input logic level
False
is ex = 1?
True
Set ex=1
False
Row 2 scanning
False
is value ==1?
True
print/store entered value
i < 3
True
Write ouput logic levels
Scan Input logic level
False
is ex = 1?
Set ex=1
Row 3 scanning
False
is value ==1?
True
print/store entered value
i < 3
True
Write ouput logic levels
Scan Input logic level
False
is ex = 1?
True
Set ex=1
False
is value ==1?
True
print/store entered value
i < 3
True
Write ouput logic levels
Scan Input logic level
False
Set ex=1

5.2.2) GUI

The LCD will be the gap between the user and the system. We will set up a very easy yet beautiful interface for our users to interact with. We have choices between a few GUI code for Python like: Kivy, PyQt, PyGUI, TkInter, wxPython etc… With a 480-320 resolution the touchscreen will display a color menu with icons. If using the PyQt, In QtCreator, just save your GUI. It will update the MainWindow.ui file. In a terminal (or command shell for Windows), we're going to make use of the pyuic5 utility. We can name the result anything we want; I will keep the name mostly the same. Each time we update the GUI in QtCreator, we will want to run pyuic5 to update the "auto.py" file.

Functional description -The interface will be simple for ease of use. It will consist of basically two main icons at the front screen like: Menu and Setup etc. The setup option is for the user setup the different mode they can change the system into. Inside setup, the user will be able to create pins, update pins or remove pins. The system will allow up to three pins or accounts for administration purposes; only people with the administrator accounts/pins will be able to configure or make changes to the system's display. At startup the user will be prompted a message to register or create at least one account to administrate the system, but later on they can add more account up to three. Also in the setup the user will be able to able to select the screen color for the display. Inside the menu will be different options to set the sensors and alarm sounds. When going into the menu, the user will have choices to disarm or arm a particular or all the sensors. If the user chooses to disarm a sensor a warning message will appear with description explaining the risks of doing so and asking to confirm the disarming. The user will be able to manually test the sensor to make sure they are always functional by clicking the On/Off button. The menu will give the user options to lower or higher the alarm sounds. Inside the menu there will also be options like reset, mode etc. The reset button will be used to reset some features in the system in case the user cannot undo a feature manually or if the system is freezes. The mode option will be used to set the sensors to a specific mode. For some sensors it may be temperature, distance, level of noise, level of smoke etc. There will be four arrow navigation panel to allow the user to navigate with the menu. Once inside the menu there will be a title (the name) of the selected option on top, on the bottom left there will be a back button to go back to the previous step once you go back there will be a forward to go forward; on the right side there will be an exit button if the user decides to go the main menu; also there will be an Ok/accept button to confirm the user's choice.

5.3) COMMUNICATION AND PROCESSING

5.3.1) WIRELESS COMMUNICATION

Wireless communication is the basis of how the sensors will communicate with the Raspberry Pi. This is a very important aspect of the project because this will warn the user if there is an intruder in their home or business. There were many constraints that came with the implementation of this because we weren't sure if we were going to use Wi-Fi or Bluetooth. The problem with Wi-Fi would be that the user needs to have internet access in their environment they intended to add the security system to. Also depending on the environment coverage would be another problem because Wi-Fi depending on the Wi-Fi

router being used it can only cover a finite area. Bluetooth was another option, but the lack of security is a problem. If someone intercepts the signal, there is a possibility that they can gain access to the system and disarm it. This is why we decided to go with a regular transmitter and receiver. We must figure out what frequency to set each device so they can transmit and receive properly. These devices are less susceptible to an intruder gaining access to the system. When it comes to arming and disarming the system with we have decided to give the user an easier way to do so by implementing RFID.

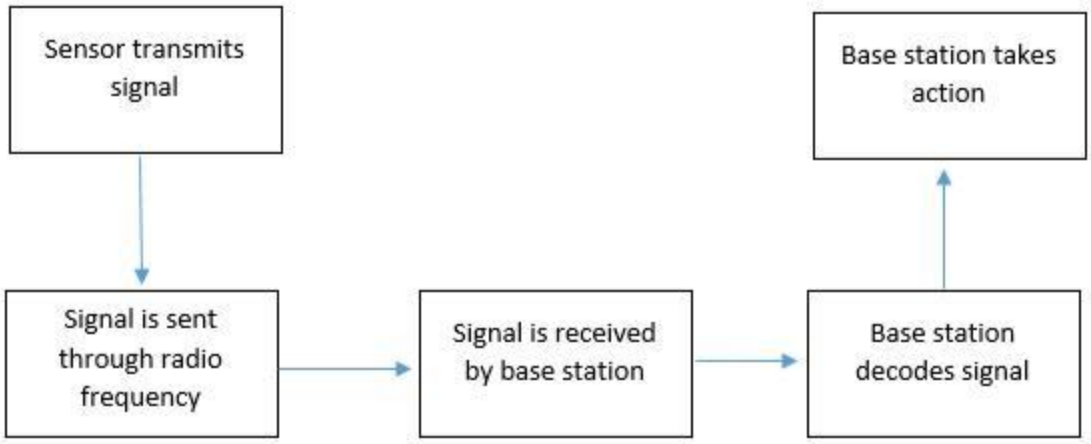


Figure 5.3.1.1 shows that data will flow in a cycle. The top left shows that a signal must be transmitted through a desired frequency. The frequency is sent over a radio wave where a receiver implemented in the base station will pick up the signal. Once the signal is obtained it will be decoded and the microcontroller will decide on the actions that should be taken if the result of the signal shows there is an intruder.

The receiver will be implemented within the Raspberry Pi and will scan the surrounding area looking for any signals given off by the TX433 transmitter that will be implemented in the sensors. Once the receiver picks up any of the surrounding signals from the transmitters that are implemented it will decode the necessary information and will tell the Raspberry Pi if there is an alert that is needed. If there is an alert needed due to one of the sensors detecting some type of fault, then there will be an alert signal to let the user know that there is an intrusion in their environment.

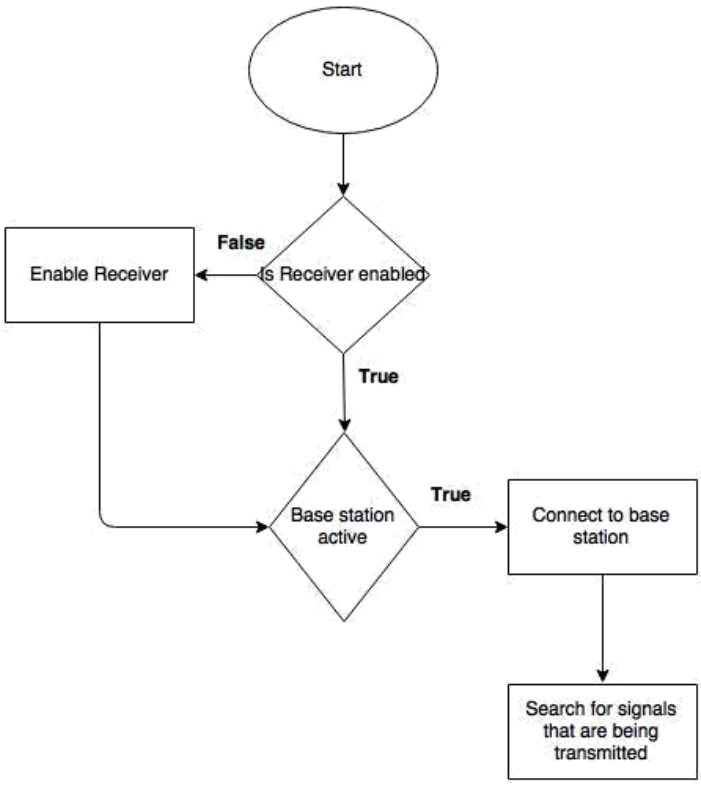


Figure 5.3.1.2 This diagram shows the software design of the receiver. The software will determine if the receiver is enabled. It will obtain the signal from the sensors implemented and decode it and then the base station will determine the status that the system should be set to.

The transmitter will be implemented in the sensors we have designed. The transmitters main purpose is to communicate with the receiver implemented with the Raspberry Pi. The transmitter will be transmitting in at a set radio frequency in the surrounding area and will send a signal depending on whether there is an intrusion or if the system is stable and there is no alert to report.

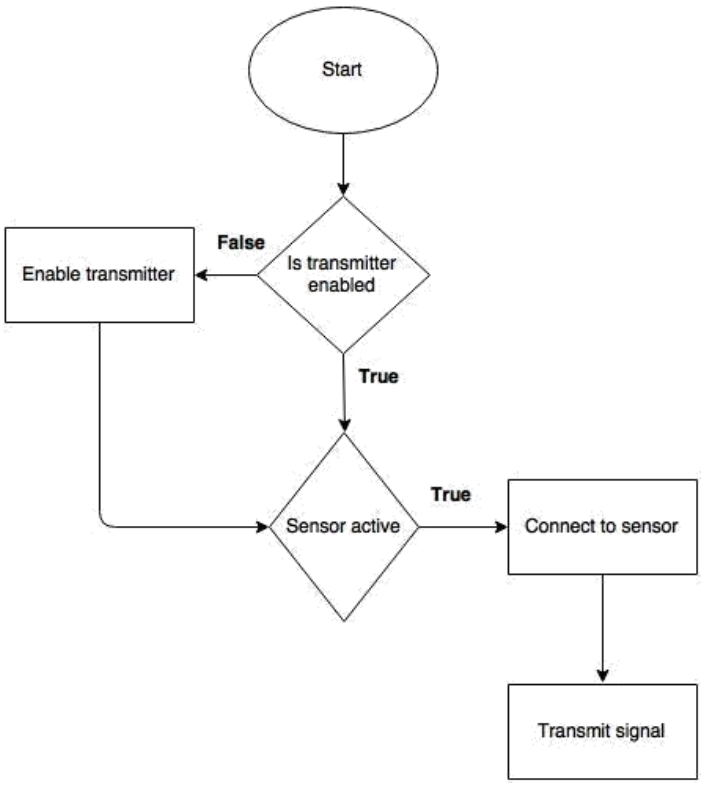


Figure 5.3.1.3 This diagram shows the software design of the transmitter. The software will determine if the transmitter is enabled and if it is it will transmit a signal in the environment to the base station to determine the status of the environment.

5.3.4 RFID/NFC

The way the PN532 breakout board is programmed to search for nearby RFID is very simple. It uses the libnfc library where it pulls the source of the information. All of the variables for the SPI connections are defined for the pins that they will be implemented on the Raspberry Pi. The board works on a baud rate of 9600 and then the NFC reader is initialized. The program then requests to see if the board is available and gets the firmware version. If it is not found the program will return with a prompt saying it is not available. If the board is found the prompt will return saying that PN532 is active. It will print the version of the board and the firmware version that it is running. It will also print out what supports are available. Then it will search for any nearby NFC cards available and print out a 32-bit unique identification associated with that card. It only supports the MIFARE ISO14443A cards or tags. Once the card is authenticated then you are able to read or write to a card.

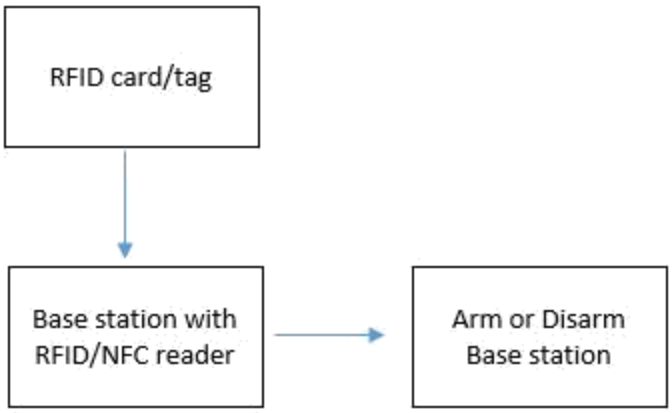
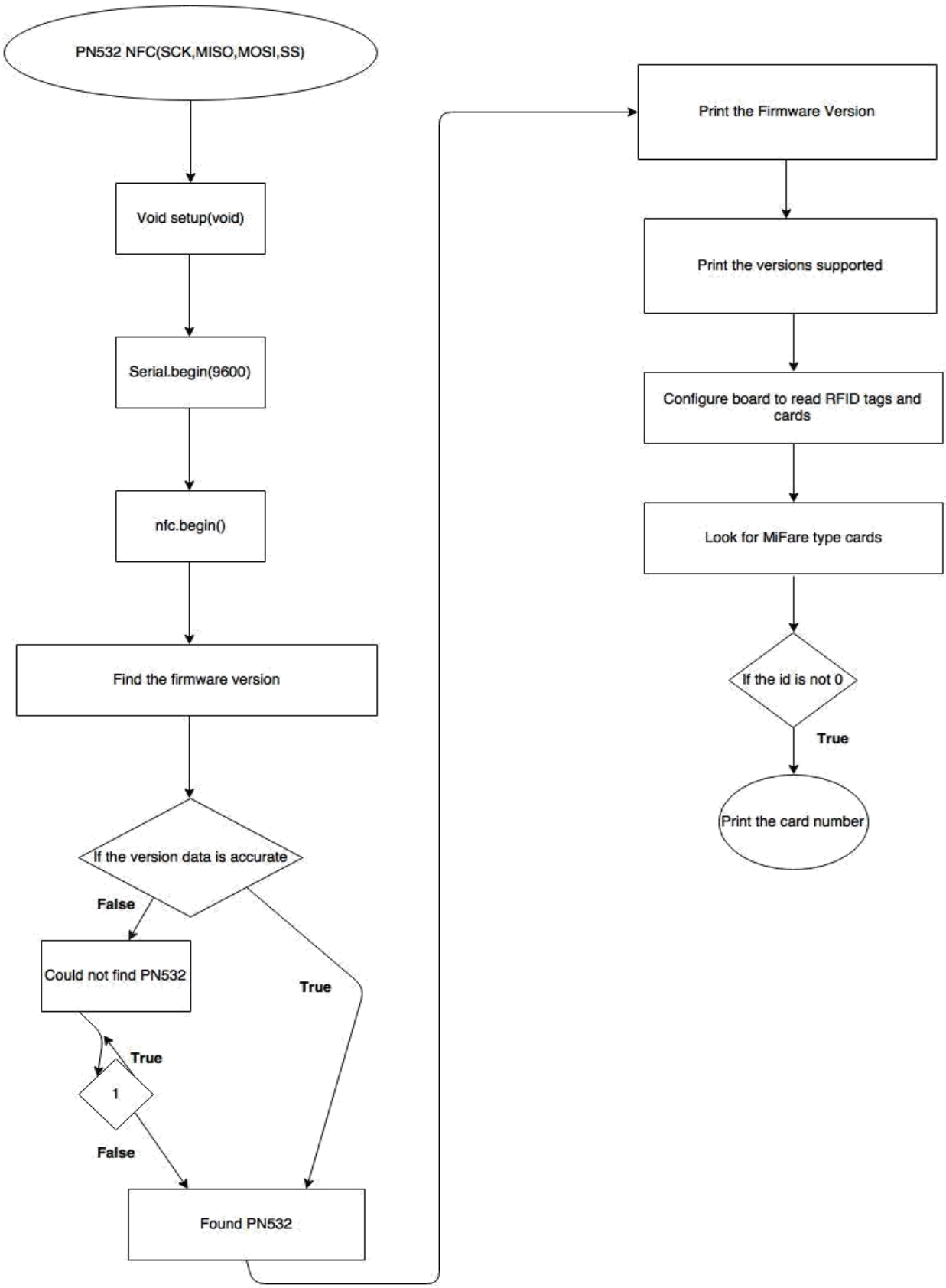


Figure 5.3.4.1 shows the data flow cycle. From the top left the RFID card or tag is in close proximity for the RFID/NFC reader to detect the card. Once scanned the microcontroller will decide whether the user wants to arm or disarm the system.

Below is a flowchart that shows how the RFID/NFC software design for the PN532 to be able to read the MIFARE card that will be used with the system.


PN532 NFC(SCK,MISO,MOSI,SS)
Void setup(void)
Serial.begin(9600)
nfc.begin()
Find the firmware version
If the version data is accurate
False
Could not find PN532
True
True
1
False
Found PN532
Print the Firmware Version
Print the versions supported
Configure board to read RFID tags and cards
Look for MiFare type cards
If the id is not 0
True
Print the card number

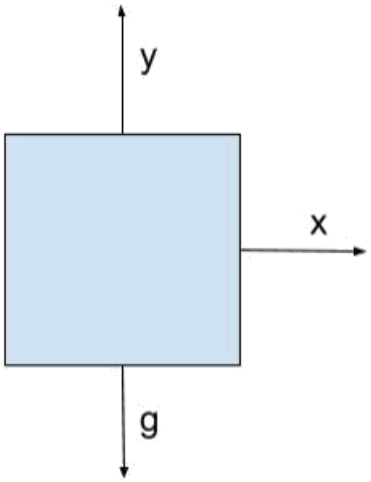
5.3.2) RFID/NFC

Implementing the PN532 to the Raspberry Pi is one aspect of this project. Next we must set up the software to enable the PN532 to be able to read the NFC tag when it is in close proximity. You will need to install the libnfc library to be able to work with the PN532 breakout board. This gives you the ability to read and write to the near field communication tag. The way libnfc will be implemented is using the UART feature on the Raspberry Pi. But the UART feature is temporally dedicated to other purposes so it must be enabled for libnfc. Having a clean Raspbian install is ideal for freeing up UART. The first step is to disable the shell and kernel messages via UART. Next you must enable UART for user usage by configuring it and setting enable_uart = 1. Reboot the Pi so the changes can take effect. Before the library can be built it needs to be configured for the targeted system and the parameters that PN532 requires. You need to place the libnfc file in a specific location of the raspbian operating system. Once the file is placed in the destination it must be updated and the line allow_intrusive_scan = true must be added. Next process is using the configure tool and installing the library. All that is left is to run the software and require the PN532 to poll and see if it reads the NFC tag when it is in close proximity. When programming the RFID/NFC the user will have a unique username and password. With this the user will be able to arm and disarm the alarm by being in close proximity with the Raspberry Pi. The Pi will refresh every so often checking for the NFC tag and will be ready when the user swipes over it.

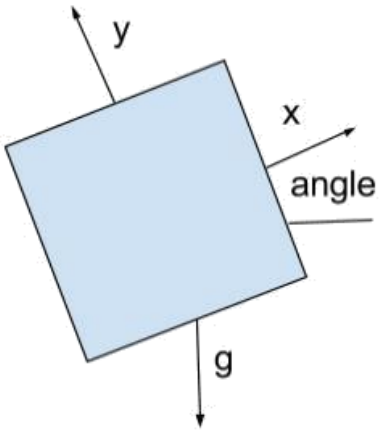
5.3.3) SENSOR PROCESSING

Sensors are used to gather information about our environment. With different types of sensors, there multiple ways we process the data from sensors. In our project we are using tilt sensor, motion sensor and fire sensor. We are developing some algorithms that will take of the way each sensor processes data and communicate with the microcontroller. It is very important for the processing of the sensor's data to be precise and accurate for better operation of the system. Sensor processing is now in a new era where you can get a multitude of high performance sensor with low cost. Sensors are used in pretty much every aspect technology world from advanced high end military type sensors like infrared, radar, electro-optical etc to small device (cell phone, tablet, etc) using cameras, accelerometers, GPS, gyroscopes etc. One key factor of sensor processing is that for maximum benefit, both the processing and the sensing must take place at almost the same time.

The processing of the tilt sensor - tilt sensor is essential in motion detection and measurement. It can track down the movement of different objects to provide useful information. Tilt sensor uses angle and displacement to give feedback on the object in question. The sensor has an initial tilt angle and/or acceleration; so whenever this angle and/or acceleration change the sensor will signal the changes to the processor.

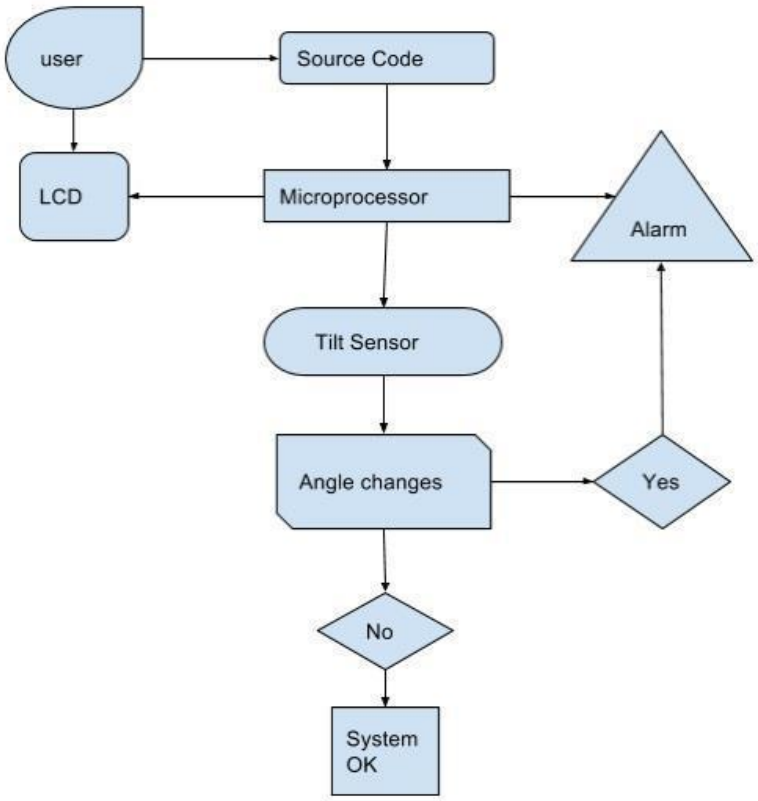


Initial object position



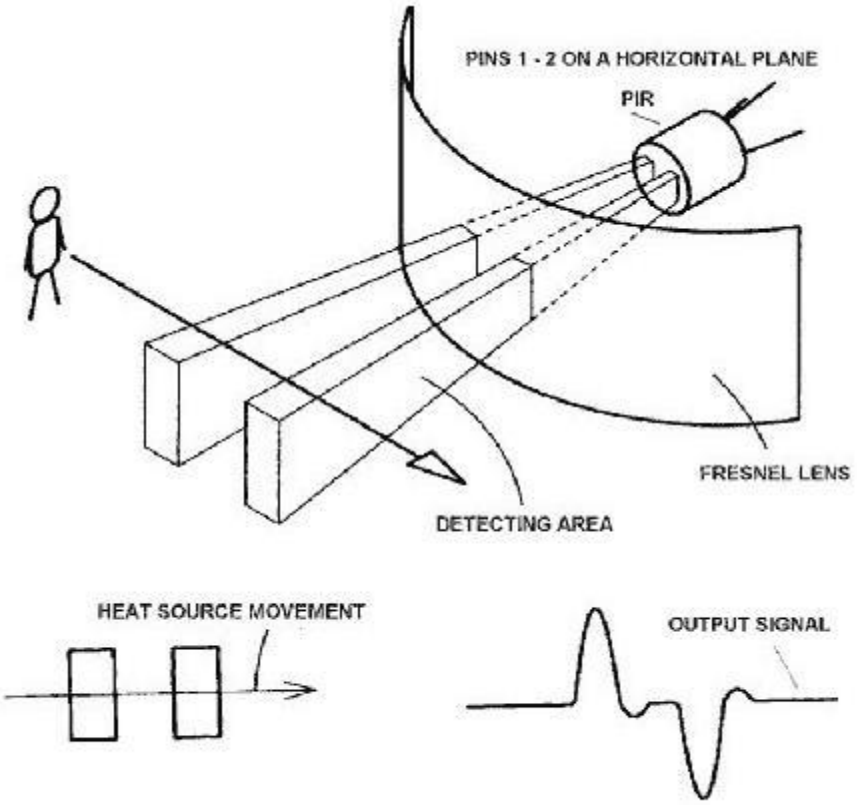
Position after displacement

The signal will be processed as soon as it is received by the microprocessor, the microprocessor will then send the state of the sensor to the alarm system so it can recognize the state of the sensor. The way the program will handle the communication is shown below on the diagram:



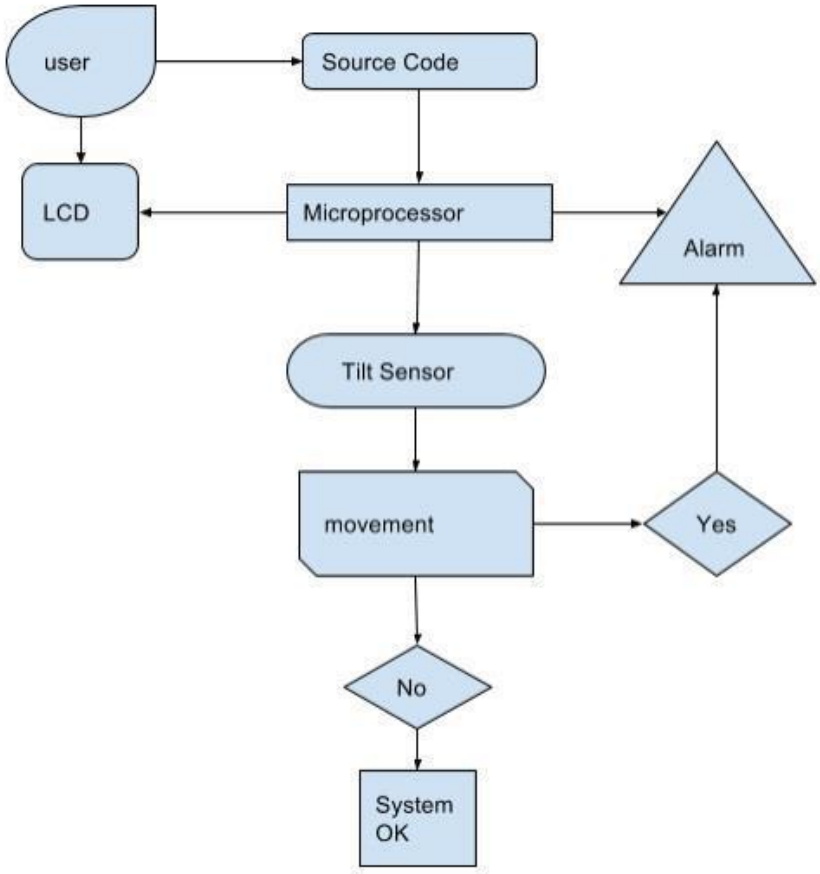
Flow chart of the tilt sensor

The motion sensor processing - the motion sensor is used to detect movement within certain calculated area of the system. When movement is detected, the sensor will send notification to the processor. The sensor will let the processor knows when there's movement in that area and will react by taking actions to the movement depending what it is setup to do. We can set lights to turn on and off automatically if someone enters or leaves the area. It send out alert with seconds of detecting movement in the area. The sensor can be set to do a variety of things such as: ring a doorbell when someone approach your door, send alert if an unauthorized person enter a restricted area or save you energy by turning on and off lights in some area in the house. Depending on the type of mention sensor you are utilizing, you will have options to do a lot of things.
This is a diagram taken from adafruit.com showing how the PIR sensor is set



The communication between the microcontroller and the motion sensor will be done wirelessly. Just like the tilt sensor, the motion sensor will be communicating with the system based on a set conditions written in the code. The response will be quick and accurate. The sensor we are using is a PIR (Passive Infrared) meaning it will detect infrared energy (body heat) to send alert to the system and the user. When the system is armed the sensor is active; as soon as it warms up it will start detecting movement and heat in the environment it will create a what is called a grid. Whenever an object is too big and blocks too many grid zones and energy levels of the infrared change quickly, the sensors will trip and will start sending out alert. Some sensors like a microwave sensor can detect microwave pulse and measures a moving object's reflection. They can cover a much larger

area than the Infrared sensors and they are way more expensive, but they can't handle electrical interference. Some sensors are dual technology motion sensors. They have a combinations of features that reduces false alarm. If we take for example a Passive Infrared sensor, it could be combined with a microwave sensor. They operate in different areas, one is passive and one is active. This type of sensors is not like the others that will cause false alarms because both sensors have to be tripped for the alarm to trigger. However, they do trip sometimes but not often. This is a representation of the motion sensor working in the system.



6) PROJECT DESIGN SUMMARY

6.1) HARDWARE

The hardware portion of our system was divided into two separate systems in which one didn't require any sort of programming but was to ensure the system functioned as needed. Overall, on the power system side of our system it was divided into about four parts which started with step down transformation of a walls electrical AC system to make it more useable for us in general. Afterwards the second part of the system consisted of simply rectifying the now reduced Voltage that we had to work with. This would raise the Vrms value of our system in general. The third step to this was to smooth the AC wave into a useable DC waveform in general. The final part of the system was that the new DC waveform available to use would be stepped down into a very stable version of what was needed. The major challenge when dealing with the AC to DC power system was that the output had to give an extremely stable voltage or else it would cause damage to our microcontroller in general.

For our sensors side of the project a wireless communication system was setup for use and tested. Afterwards, various types of attached sensors would be mirrored to this system to give a wireless sensor network. The premise is simple but, various issues occurred when trying to fulfill the needs of the project on this front. Things such as needing to design a low power system to not drain batteries and such all became factors when designing this system for our use. Some of the other struggles that became apparent and had to be factored into the design of our hardware was not only the basic power requirements, but the power requirements of the microcontroller in hand. One of the big design choices that were made when talking about the microcontroller was that the GPIO pins only supported a very tiny amount of power that could be drawn from the microcontroller on those pins. This was fulfilled by designing a system that essentially worked as a flow control for a pipe. This had a stronger voltage/current rail on the microcontroller feed into the system which the weaker one which was in charge of if the system was feeding or not. Other than this, the only other major consideration in hardware design was that we had manage our pins efficiently as a very big amount of pins ended up being required for the entire system design. We essentially had 40 pins to deal with while almost half of them were marked for other uses in general.

6.2) SOFTWARE

This section of the document is to fully explain how the system will work when it comes to software. It will allow any developer using the system to have an idea on how the system operates. We will detail all the functionality of the system.

The software system will mainly be based on a set of loops and conditions to facilitate communication within the system. The microcontroller will be programmed to display our menu on the LCD touchscreen display and also receive signal from the sensors to change the state of the system. When the system starts, the LCD will display the menu with the appropriate icons to facilitate the user to easily get the system up and running. After starting, the microcontroller will communicate with the sensors to figure out if they are operational; if the microcontroller can't reach the sensors the LCD will display an error message indicating that the system is not fully operational. If the communication is

successful, the system will go on. The user may want to test the system by testing each sensor to make sure they are responding correctly to the actions and sending the correct signals to the microcontroller. For the Raspberry Pi 3 we're using, the code will be written in the language more suitable or easier for the system.

7) PROTOTYPING

This section describes our final prototype design for our hardware and software that was used and will be used in our PCB forward.

7.1) AC->DC CHARGING CABLE PROTOTYPE WITH USB

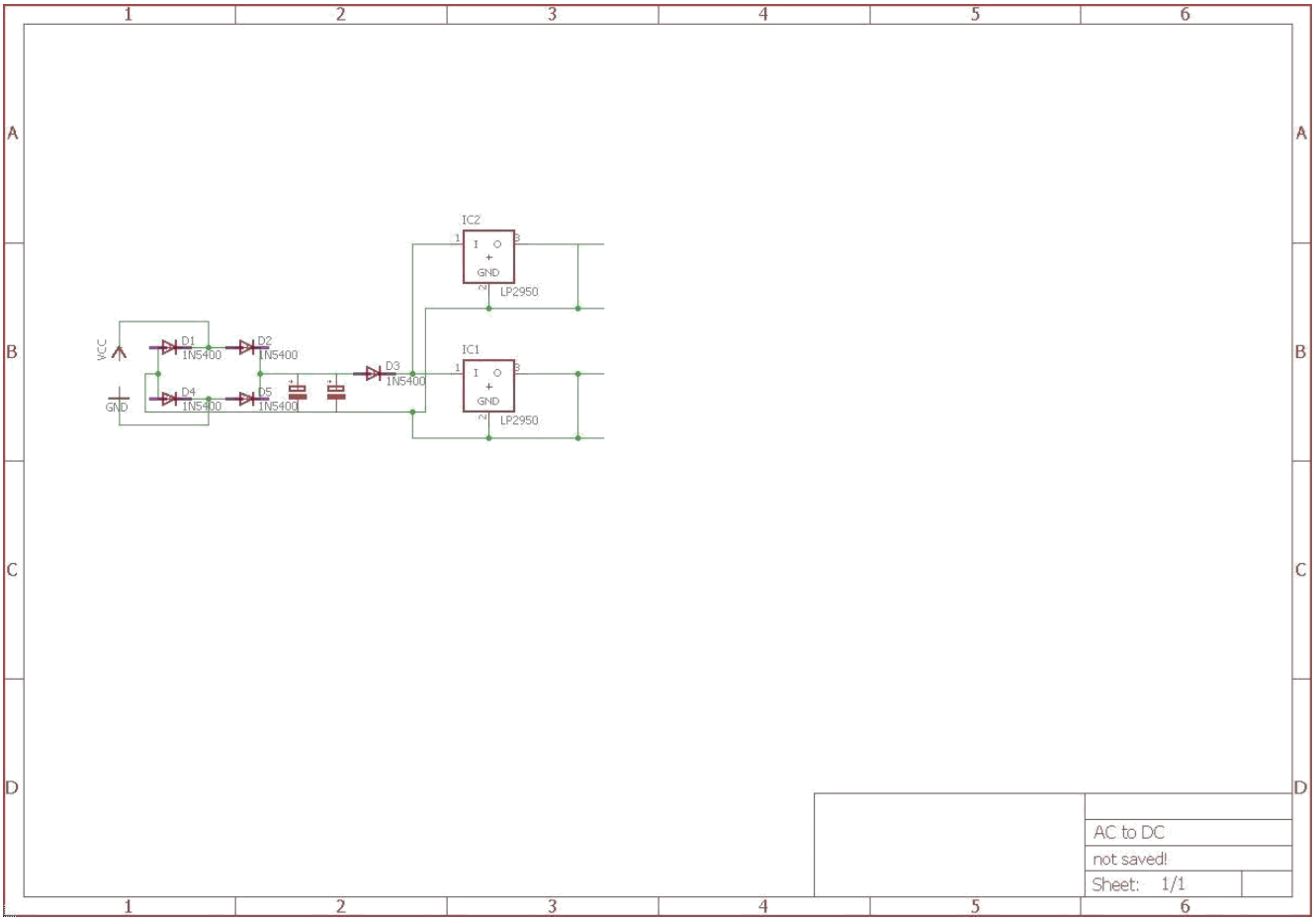
7.1.1) PARTS LIST

This section contains the tabulated reference data on all parts used within the design of the AC to DC power system. Our "Part #" can be used to find data on the specific parts used within the design.

PART ID	MANUFACTURER PART #	DESCRIPTION	QTY	UNIT PRICE $	TOTAL COST $
001	MGT-1220	TRANSFORMER 120VAC 60HZ TO 12VAC 60HZ	1	12.90	12.90
002	1N5408	DIODE 1000V 3A	10	.423	4.23
003	D24V25F5	BUCK 5V 2.5A	2	10.95	25.85
004	50UT4700	CAPACITOR 4700 UF	2	2.21	4.42
005	CPC05-03	CAPACITOR .5-1 UF	2	0.00	0.00

7.1.2) PRE-PCB SCHEMATIC FOR PCB CONVERSION

The primary program that will be used for the PCB schematic will be Eagle, which is the industry standard currently. In order to utilize the PCB function for Eagle a Schematic had to be drawn up within it. This is mainly because the program isn't used for simulation so it doesn't allow easy conversions between them. The program itself does allow customizability of parts to meet our needs in the end and provides a library of many common to find parts and the dimensions for where holes should be created.



*Design was created using Eagle.

7.1.3) PCB PROTYPE ORDERING AND PCB DESIGN

The following is the PCB design that will be sent to the company ExpressPCB and will be the final hardware received for testing of our power system. It is essentially the finalized product other than a case to be added to protect the end user from being shocked by high voltages. The company selected was used as they offered a very quick turnaround time along with a decently low price for us. The turnaround time is very important in case for some reason our PCB design doesn't work originally or we need to reorder a corrected PCB. The price between single layer and double layer was also very low for our needs in a PCB. It appears the actual price difference between single and duo layer PCB's happened to be negligible in terms of difference. So a duo layer PCB is preferable for our part design and ordering. For the actual creation of the PCB design instructions we used the program Eagle due to it being the most widely used currently. It isn't the easiest program to use, however, it provided a massive look into how the industry does things on a professional level and will be a valuable tool for once we start employment.

8) PROTOTYPE TESTING

Testing is the most crucial element of our project as it tells us if we have it working at all or not. Our overall plan for testing was to compartmentalize it into roughly 4 parts that can be tested independently and then combine them at all the end for one final test. The primary driving factor behind this is that it allows the group to cut down on in person meetings while still yielding appropriate results for our project. We broke it down it down into four sections which were Hardware Test 1,2 and Software Test. Hardware test one is planned to be our entire compartmentalized test of the power system. Hardware test two is planned to be the sensors and pushbutton testing for the project. Finally, the software test is planned to be a combination of all the parts to ensure they are functioning to maximum efficiency. This helps to cut down bugs within our test environment and lets us know if the issue is hardware or software related before even attempting the systems together. It also means there will be less of a headache for programmers when it comes to debugging the system.

8.1) HARDWARE TEST 1

8.1.1) TESTING PLAN

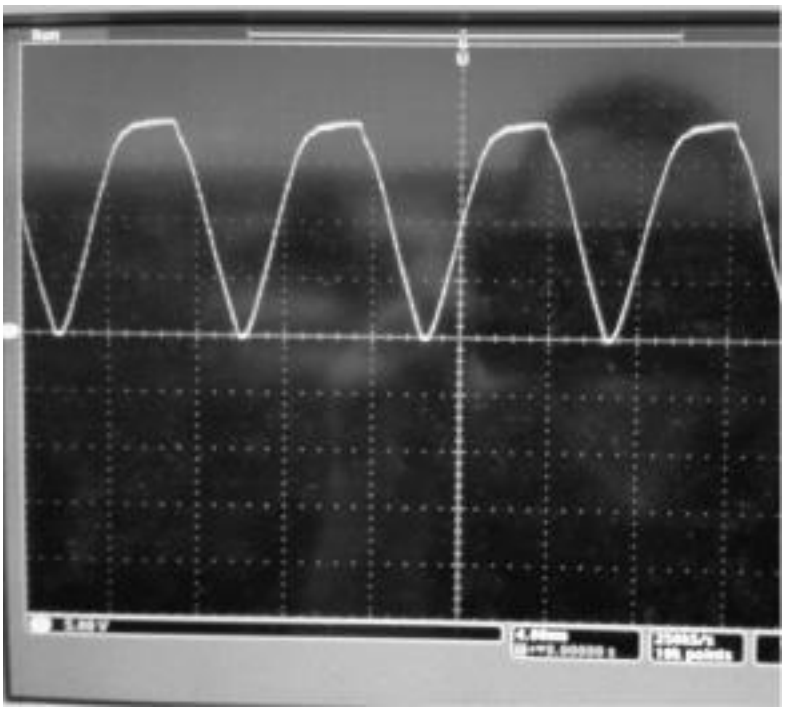
The entire AC to DC power system was designed to be tested in four parts. The plan for our system was to design and test each stage in the same way that the current flows throughout our circuit. The exact order of parts was as follows: AC Transformation, Full Wave Rectification, AC to DC Smoothing and our final buck conversion. Specifically, all parts were checked using an Oscilloscope to verify voltage and waveform as we had very exact requirements. The first and most important reason for testing each portion individually was because of a very real risk to persons and circuitry. Along with this no person was allowed to actually go near the circuit while hooked up to testing equipment due to the high voltages that come with dealing with a wall outlet. Our second reason for deciding to test in this pattern was to cut down on errors that may arise. If any portion displayed an error in its output, it would be sent back to redesign and fix the error. Lastly it was tested in this order to document results for each step in the event of expanding upon our circuit.

8.1.2) AC TRANSFORMATION

This part wasn't one we actually designed by us but had to be evaluated to see if it fulfilled our needs. The part specifically was to step down transformer which was supposed to change 120VAC to around 20VAC and maintain a frequency of 60Hz. Due to AC->AC transformers not being a common item on the market we had to grab one from a HVAC store. During the test for this part it was plugged into a wall socket and had used wiring provided by UCF. The result when hooked up to an Oscilloscope was that the part displayed an output of 26 VAC at 60Hz. We had a lot of leeway when it came to the voltage available to us, with the only real limiting factor be the Buck Converter. The part fell well within our range and had minor distortion on the output waveform. Overall, the parts testing was a complete success.

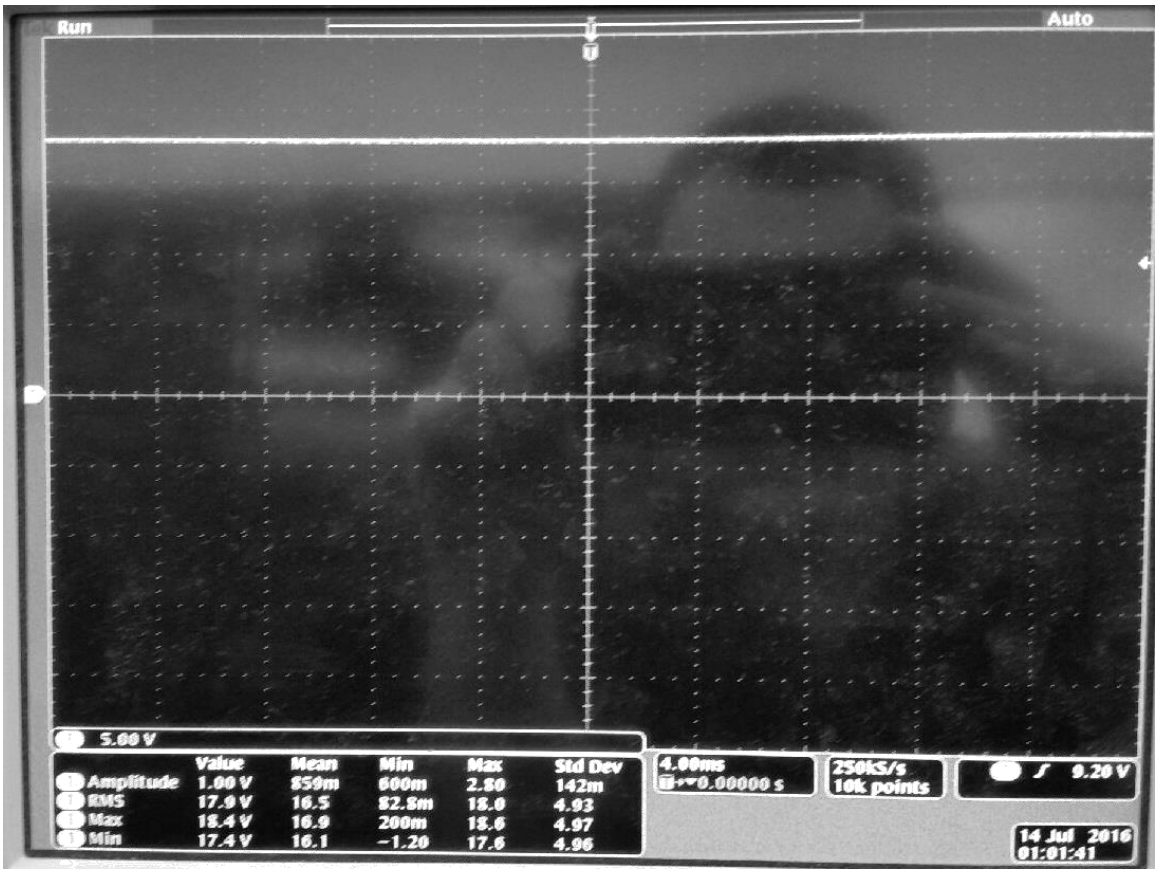
8.1.3) FULL WAVE RECTIFIER

Our next part up for testing was a combination of our previous transformer and diodes. This portion involved a standard full wave rectification layout involving only four diodes. The most important thing was that it rectified our wave in general. The actual accuracy of the wave wasn't nearly as important, since in the end in general it would be smoothed over into a DC signal. As can be seen with the image there is minor distortion and the bottom of our wave is slightly clipped. This was expected to happen with the circuit though however as the diodes have a turn on voltage before they are active. One thing to note especially with the circuit is that our amplitude actually is slightly lower then measured. This is because the diodes actually cut down some of our voltage for use. However, the most important measure which was our Vrms is actually about 40% higher due to the wave being rectified. This does mean that our current will however take a minor hit due to the increase in voltage. Our output voltage was still well within our useable range which could had been as low as 6-9V.



8.1.4) AC TO DC CONVERSION

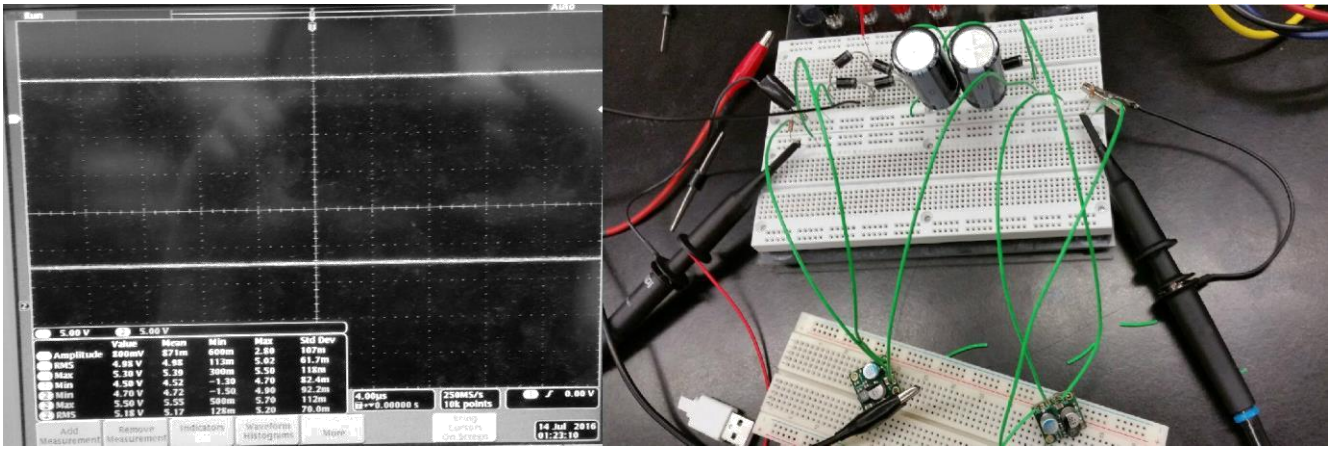
Our next portion of the circuit was to take our rectified wave and smooth it into a DC wave. This was done by using a smoothing capacitor to filter out all the waves within the AC waveform. We expected the voltage of our new wave to have an even higher Vrms as a result and possibly a higher amplitude. One of the considerations for the circuit was the ripple had to fall within our needed guidelines. This was to avoid causing issues with our buck converter so it wouldn't drop out due to the voltage drops. The secondary reason why it was a major concern wasn't for fear that our buck converter was damaged but that



it would destroy our microcontroller that we utilized for our project. The results can be seen here.

8.1.5) BUCK CONVERSION TO NEEDED FORMAT

The final portion in the testing procedure when it came to our prototype was that it downgraded our now higher DC voltage to a 5V 2.5A output. The reason this was our needed Voltage/Amperage was because that is the maximum supported input for the Raspberry Pi 3 line of microcontrollers. We had a very small amount of tolerance when it came to the voltage of the Raspberry PI 3, so ideally we needed an exact 5V output. This also would put us close to the needed Voltage/Amperage for our USB charging mechanism. The only thing out of the ordinary that resulted from our final test was that due to the size of the capacitors we used there is actually a considerable charge-up time for the item. This means that there must be a time allowance when it comes to first plugging the power into the Raspberry PI or else the Voltage may not be ideal or as stable as required. After the capacitors have charged though, the voltage was incredibly consistent and sat at a perfect level for use within our circuit. This means that we can conclude that about 30s should be given to the power supply before plugging it into our system.



8.2) HARDWARE TEST 2

Push Buttons:

Our main concern when designing the pushbuttons was to limit the current coming from the columns of the pushbuttons to a safe value for when all of the push buttons are pressed. First, we needed to design a simple code that would test a single column for the push buttons. This way we could see if our setup was working or not. The reason for this is to eliminate complications down for when a button wasn't reading correctly. It would be a hassle to assemble all of the push buttons and having a whole row or column not working correctly.

Initially, we assembled a complete column of pushbuttons (buttons 1,4,7,*). We started off by pressing a single button which will send out a print command on the screen to let us know that the button has been pressed and that it is working. Our next test was to measure the voltage drop across the resistor connected to that row to ensure the current draw was not exceeding our calculations of around 3.3mA. Our measurements revealed to us that the voltage drop across the single column resistor was around 0.08V. Immediately, our thoughts were that the GPIO input pins don't have a specific current that it needs to be at in order for it to read a digital high or low. It determines how much voltage is at the node that it is connected to in order for it to make that decision. Additionally, this conclusion that we made about the GPIO input pins was expected since a logic high or low is purely dependent on the voltage readings at a single node. Since our current values were extremely low, then that means the internal circuit of the GPIO pins obviously has something to do with these values. Unfortunately, the schematics for the Raspberry Pi's GPIO pins are not fully accessible.

After a successful first test with a single button, we decided to press multiple buttons to determine the current draw coming out of a single GPIO pin. When two buttons were pressed the voltage drop across the resistor connected to the column was 0.263V which means, based on ohms law, that the current draw coming from the column was 0.263mA. When three buttons were pressed the voltage drop across the resistor was 0.31V. Finally, when all four column buttons were pressed, the voltage drop across the resistor was 0.66V.

Our objective for this design was to use as little power as possible. Because the maximum voltage drop across the resistor connected to a single column was 0.663V, then this means the maximum power that the push buttons are using is 0.439mW. Therefore, our test for the pushbuttons was a success.

Led:

To test the transistor connected to the LED's we decided to initially connect the base to the 3.3-volt rail since that's what a GPIO pin will output and the collector to the 5-volt rail. When the connection was complete and the LED was illuminating we decided to measure the voltage drop across each resistor and also the voltage drop across the LED and the turn-on voltage for the base-emitter/collector-emitter. Here are lists of those measurements.

Base Resistor Voltage drop: 2.647V

Collector Resistor Voltage drop: 2.91V

Led Turn on Voltage: 1.858V

Vbe Turn on Voltage: 0.653V

Vce Turn on Voltage: 0.232V

After measuring the voltage drop we decided to calculate the current values at the base and collector. Our calculated values for the base was 27uA and our measurements revealed to us that the base is using a total of 26.47uA. Therefore, our calculations have a very small percent error, which is great. Additionally, our collector current measured to be 2.91mA and our calculated value was 2.8mA. This gives us a percent error of around 3.92. This is an acceptable percent error for our application. After measuring the current values we can then determine the beta value for this specific transistor which is 110. We can use these values for future calculations just in case we need to make some adjustments to our design.

After testing and measure our elements in our design, we then implemented a simple blink code that will turn on the GPIO pin for around half a second and then turn it off for half a second. Our results were successful. Additionally, our goal for this was to limit the amount of current that the GPIO pin would be using by implementing a transistor switch into our design. We successfully limited the amount of current draw from a single GPIO pin down to 26uA from a total of 2.6mA. Therefore, the amount of power that the GPIO is using for this design is 85.8uW.

Door Chimes:

To test our reed switches we will be implementing the led design that we previously tested. We wanted to be sure that current draw from the 3.3-volt rail did not exceed our calculated value. Initially, we set up the circuit by connecting the 3.3-volt rail to the 1k resistor which was in series with the reed switch. The reed switch lead to the pull-down resistor and the GPIO pin that would read the voltage values at that point. We started off by writing a code that would turn on the led when the magnet was not present and would turn off the led when the magnet to open the reed switch was in range. We have concluded that the magnet we will use will need the distance of about an inch to activate the reed switch to switch it open.

After we figured out the distance for the magnet placement we decided to measure the voltage drop at the resistors. The current limiting resistor had a voltage drop of 0.35V when the magnet was not in range and 0V when the magnet was opening the switch. This means based on ohms law that the current draw from the 3.3-volt rail is 0.35mA. This is actually significantly less than we calculated. The reason for this is because we are not able to access the schematics of the Raspberry Pi's GPIO pins. Although the current draw was calculated at 3.3mA, this is not an issue since the less current that is being drawn from the 3.3-volt rail the better. Additionally, the voltage drop across the pull-down resistor is 2.9V. This means that the current running through this resistor is 0.29mA. Using Kirchoff's current law, the current going into the GPIO pin is roughly 0.06mA. Because we know the current going into the GPIO pin and also the voltage at that pin, we kind determine the internal resistance of the GPIO pins. Using ohms law, we have concluded that the total resistance is 48.33kohms. Finally, the total power consumption that the 3.3-volt rail is producing when the reed switch is closed is 1.155mW.

Tilt Sensors:

The biggest issue we had when testing the tilt sensors is obtaining the measurements across the 2kohm resistor. Because the tilt sensor only makes contact for roughly 2ms then we will have to use our best judgement when reading from the multimeter. Initially, we decided to use the led circuit again for our testing. When the LED is on then the microcontroller detected a high logic value coming from the tilt sensor. Likewise, when the LED is off then the microcontroller detected a low logic value coming from the tilt sensor.

Because we are experiencing some difficulties with our wireless transmitter and receiver, then we will design a test plan for this component when we finally get it working. Our first plan of action will be to measure the current draw from the 9-volt battery. We will obtain this current by measuring the voltage drop across the 2kohm resistor and then connecting the multimeter in series with the Vin of the transmitter and the 9-volt rail. From this information, we can then calculate the current draw and the power consumption of the battery while the transmitter is on. We will also need to note the contact time of the tilt sensor. We need to assure that 2ms worth of current will be enough for the transmitter to read a logic value and then transmit that data to the microcontroller. We are assuming that it the contact time should be efficient enough. Finally, when we connect the tilt sensor to a simple resistor LED circuit, we can note that the sensor works when it is shaken simply by looking at the state of the LED.

Motion Sensors:

As mention previously, since we are experiencing trouble with our wireless communication module, then we will need to test the motion sensor module separate to ensure that it is working. To do this, we will simply connect a current limiting resistor to the output of the module. This resistor will connect to a pulldown resistor which will then connect to a GPIO pin for reading the state at which the output of the module is at. We will then measure the voltage drop across the 1k resistor and also the total voltage that the output of the module is producing to make sure it agrees with the datasheet. We will also be implementing the LED circuit like the previous examples. Once motion is detected then

the LED will turn on, and when motion is not detected then the LED will turn off. These are the voltage values for initially testing our motion sensor.

Current Limiting Resistor: 0.282V
Pulldown Resistor: 2.37V
Output Voltage: 3.311V

Our tests reveal to us that the current running through the pull-down resistor and the current limiting resistor are 0.282mA and 0.237mA respectively. This means that the total current that the GPIO pin is receiving is 0.045mA.

Fire Sensor:

To test our fire sensor, we need to consider our assumed voltage levels at the output as being 5 volts. The reason being is that there was no voltage listed on the datasheet for this module. When we tested the output, we discovered that it was producing 1.68V when there was no flame detected and 0.08V when there was a flame present. This means that the GPIO pin will constantly be receiving power from this module until a flame is present. Additionally, this means we will not be implementing a voltage divider or current limiting resistor since we calculated the GPIO resistance previously.

First, we set up the fire sensor and LED circuit as we previously did with our other tests. Next, we decided to not introduce a current limiting resistor because we did not want to have the voltage drop below the digital high reading which is around 1.2V. We wrote a simple code that will check to see whenever the input drops to a logical low value. When it does, this means a flame is detected. Since this is an IR sensor, we also waved our hand over it to make sure it will not drop low from body heat. Our tests have shown that the fire sensor did not, in fact, drop low when body heat was detected. Finally, we tested to see how far the sensor could detect a flame from a lighter. The distance at which we recorded was around 6ft.

8.3) SOFTWARE TEST 1

To test the software, we will be using the codes written section by section. No formal software-hardware testing has been setup yet, so we are using the various section code that have been written so far to describe how the testing will be conducted. When testing the software, we will be checking for the conditions that we set. The first set of the test will be focused on the sensors. We will check our sensor-processor communication to assure a good functionality of the system.

Tilt sensor test - By implementing a wireless network which can be configured and operated remotely, there is little overhead involved in installing and maintaining the system on-site. Additionally, with the ability to power the nodes using batteries, the system can be set up without extensive support infrastructure, such as power or signal cables over large distances. This is specifically beneficial on construction sites, where cables can often be accidentally damaged.

Because these nodes are often deployed in inaccessible locations, it is important to minimize personnel interaction with them by maximizing their battery life in the field. Configuring the nodes to transmit collected data at predefined intervals considerably prolongs battery life. Carefully selecting transmission intervals can result in battery life of up to three years; should longer monitoring periods or high sampling rates be required, we

can implement alternative off-grid power supply methods, such as solar panels. Users can view received data either through a direct PC connection or via a network connection. The network connection, achieved by connecting the gateway to a modem, is ideal for monitoring applications remotely for potential worldwide viewing.
The actual data logged from each node consists of both the sensor measurements as well as each node's system health. This system health information includes a timestamp and the battery voltage and signal quality of each node.
wireless tilt sensors have become a powerful tool for building and construction monitoring, as they require virtually no maintenance while being easily adaptable and reconfigurable.

8.4 SOFTWARE TEST 2

The software test plan will primarily involve how the system will work in collaboration with the sensors and the RFID/NFC reader. It will be broken up into different sections to test each of the sensors that require wireless communication with the system and the radio-frequency identification that will be implemented in the microcontroller.

8.4.1) MOTION SENSOR

With the Passive Infra-Red motion sensor, it will detect if there is an intruder. Before implementing this design with the full system it must be tested to ensure that it works. The software developed for this to work the motion detector will be connected to the microcontroller because the PIR has a digital output all that will be needed is for the pin to either read high (which mean there is motion detected) or low (there is no motion detected). The motion detector is searching for some type of infrared signals from body heat and the radius it is searching for is between 6-7 meters. For testing purposes an LED will be implemented on the breadboard and will emit a light when motion is detected. For the purpose of this test GPIO#21 is chosen for the Led pin. The input pin for the PIR is GPIO#11 and we will assume there is no motion so we will set the state to low. We will set the LED as the output and the sensor as the input. We will set a variable that will read the input value with the built in function. If the value is high, then we will write to the LED to turn it on meaning that there was motion detected and print to the screen that there was a detection. If the value is equal to low, then the LED will be off and we will print to the screen that there is no detection.

This will help with the wireless communication aspect because when there is a detection the transmitter will transmit a certain frequency to the receiver embedded to the microcontroller which will alert the user that there is motion detected. To ensure that Piezo buzzer will work with the system. It will be implemented with the motion sensor. When motion is detected there will be an alarm sound that will be played through the buzzer. If there is motion detected, then the alert will be played 3 times. The software implemented will have different melodies and when the system is triggered it will play a high frequency tone that is distinct from the other tones.

8.4.2) TILT SENSOR

The tilt sensor will be implemented on the doors and windows checking for any type of intrusions. Before implementing this design in the full system a software test must be done to ensure that it will work with the system properly. The sensor will be set up on a breadboard connected to the microcontroller with an LED. Much like the motion sensor

software test, but instead we will be looking for some type of incline that will set off the sensor. When the sensor is offset by some degree it will trigger the system and the digital output of the LED will be high which will make it light up. There will be a prompt printed to the screen that will alert the user that there is an intrusion. If there is no incline, then the LED will be set to the low state which will not trigger it to light up and the prompt printed to the screen will alert the user that there is no intrusion. When an intrusion is detected the transmitter that will be implemented with the tilt sensor will send out a designated frequency to the receiver embedded in the microcontroller which will alert the user of the intrusion.

The piezo buzzer will be implemented on the breadboard with the tilt sensor as well to make sure that the designated tone will work when the tilt sensor is activated. When the tilt sensor detects the incline and the LED state is set to high the piezo buzzer will play a tone that is specifically designed for the tilt sensor to notify the user that there is an intrusion at one of the doors or windows.

8.4.3 RFID/NFC

The radio-frequency identification is a vital part of the system because it gives the user an ease of use when it comes to activate or deactivating the system if they are unable to type in their passcode. The software test designed to ensure that this system will function properly when implemented in the full system is as follows. The PN532 is connected to the Raspberry Pi via SPI communication which will then communicate through a serious connection within the system. When the system is active it will scan the surrounding area searching for any MIFARE cards that it is compatible with. Once the card is in close proximity with the reader it will print the cards unique identification number to the screen. Then the system will determine if the user wants to activate or deactivate it based on the current status of the system. The test will commence with an LED that will be the digital output for the system and helps to indicate the status of the system. When the LED state is high it will show that the system has been activated and when the LED is in the low state it will show that the system has currently been deactivated.

When it comes to arming and disarming the system the user must have something to notify them besides the LED lights that will be implemented or the LCD screen. With the piezo buzzer there will be a specific tone that will be coded to notify the user when they pass their RFID card over the base station to arm the system. Then there will be a very specific sound that plays in order to notify the user that the system is then armed. When the system is disarmed through this system there will also be a very specific sound notifying the user that the system is now in a disarmed state.

Once all the hardware and software component testing have commenced and all test components have checked out that they are functioning currently. Then the home secured system will be ready to be tested completely within the expected user environment which falls under a normal home setting. This system will be ideal for any user that is trying to monitor their household from any intruders and is looking for a system that is an easy set up solution while also containing a simple graphical user interface that will not only look nice, but offer the ease of use that comes with it. The system offers many benefits

from the ease of activating or deactivating their system with RFID/NFC to simple motion and door sensors.

9) ADMINISTRATIVE CONTENT

Throughout the work on this project, we quickly realized that when taking on a project such as ours, organization and teamwork is absolutely essential. One of the biggest things we saw was a need for strong leadership and devotion to putting many extra hours into making the project work. Since we required many features to be pieced together between Electrical and Computer engineering along with programming for virtually every single step, it required a huge amount of organization as well as teamwork.

The project was begun by brainstorming ideas of a commercial product that could be improved upon. We then had to see which products people were actually interested in and was reasonable to be worked on. After figuring out what our group wanted to work on in general, we all had to brainstorm ideas on what our system would contain. The ideas to be brainstormed was a mix of standards in the industry along with newer improvements. We all independently came up with things that we would like to see within our system as we believe this would encourage more creativity from our group members. Afterwards we compiled our list of things we wanted and discussed what was practical given our time constraints. The only ideas given that were a requirement were ones that would count as an industry standard. Afterwards we had to choose what would be considered an improvement or convenience over current day systems. One of the considerations we made when it came to the creativity side of our project is that they had to be practical for us to implement within our time given.

We originally began the project working fairly loosely together and on our own for most of it. We did however have very exact assigned roles for each person that were decided on. However, this quickly showed that not all of us were working at the same speed and some portions required people to work more closely together. This is when we saw firsthand how important communication and timelines were becoming. We decided due to the small amount of people to organize through a big text chat and a few google docs as not all of us had Facebook, Instagram etc... Google docs was decided to be used because it allows for setups of graphics, marking whose working on what and let everyone simultaneously work on the same document. The other primary factor for using it was the fact it was free to use. The first Google doc we setup was to map out the general expectation of what everyone should be doing along the actual mapping of our microcontroller pins. The reason the microcontroller pins being mapped out was extremely important was we were still assembling parts independently and didn't want to overlap pins used to cut down on errors and needing to configure the code used in the end with the LCD screen. Essentially this document laid out what the final GUI should look like in our project when completed. The other drive document used for organization purposes of our project was one that listed the table of contents of our report only. This was used to indicate what sections people were working on and if that section of the report was fully completed or needed more work. This was to eliminate early overlapping we had originally within the document specifically as people were turning in multiples of the same work and causing issues in general. Afterwards, it came down to sharing code between documents for testing purposes. Due to Google docs we could simultaneous work on the same code and test new builds instantly and without

issue. Text chat served mainly as a reminder then as a primary work spot to tell people to work on documents.

In person meetings weren't the most vital part until the end of the semester. We originally had a few in person meetings just to discuss ideas and sort out the goals we wanted for our project. They were essential to ensure people not only shared all the parts they had available with each other, but to ensure the code works for the hardware we had available to us. The use of technology helped to minimize the exact need of meetings until the very end just to demonstrate correct functions. The group meetings were fairly consistent and generally at someone's house. It didn't need to be too specific as to whose house we met at, as everyone has a copy of the microcontroller at the heart of our project. The only other considerations for these meetings were they had to be based around the time most people didn't have classes which generally fell on late
Tuesday or Thursday.

9.1) MILESTONES AND SCHEDULE

9.1.1 ASSIGNMENTS AND MEMBER RESPONSIBILITIES

In order for our project to meet our goals in the timeline we wanted established, we had to break down the work into four parts for each person and set deadlines. At the very beginning of the semester we came up with all the work needed for the project and tried to map them into four equal parts in terms of the amount of work each portion would encompass. Afterwards we divided the Hardware and Software portions into two parts each. For our hardware we had the hardware design of Sensors and the hardware design of our AC->DC power system that was to be used to power our microcontroller in an original fashion while also being able to power other subsystems as needed. For the programming portion one person was responsible for the I/O of sensors and the other was to program the LCD screen and piece all the systems together into one big functioning system. It was from there on out we tried to assign writing portions based on the parts they personally will be doing in general. We essentially tried to keep people working on an honor system for their work and to meet three big deadlines but this turned out to be a failure and was eventually swapped to turning in two pages of work every two days. Unfortunately, our planning was very exact and left little in terms of leeway to meet this goal. Each person ideally was supposed to turn in 30 pages and no less. Towards the end of our timeline snags were hit and most had to work overtime to meet the goals needed. People falling behind was tracked in our reports turned in every two days with pages being tallied up for each failure to meet a deadline. Table 9.1 shows how many pages were assigned and how many pages were actually turned in per member to meet our goals.

Table 9.1: Page Amounts Total

MEMBER	ASSIGNED PAGES	ACTUAL
PHILLIP	30	30
JAMES	30	30
TONY	30	30
JOSH	30	30

Table 9.2 Shows the general sections each member was assigned to and category while Table 9.3 is our detailed schedule and breakdown for both Senior Design 1 and Senior Design 2 that needs to be met. The dates are however very flexible as some of us finished much later and some of us finished much earlier than expected.

Table 9.2: Overall Discipline breakdown

HARDWARE ASSIGNMENTS	
ASSIGNMENT	Member
TILT SENSOR DESIGN	Josh
FIRE SENSOR DESIGN	Josh
SENSOR COMMUNICATION (CHOOSE)	Josh
MOTION SENSOR DESIGN	Josh
TRANSFORMATION(CHOOSE)	Phillip
RECTIFICATION DESIGN	Phillip
AC->DC CONVERSION DESIGN	Phillip
BUCK CIRCUIT(CHOOSE)	Phillip
SOFTWARE ASSIGNMENTS	
TILT SENSOR PROGRAMMING	Tony
FIRE SENSOR PROGRAMMING	Tony
MOTION SENSOR PROGRAMMING	Tony
COMMUNICATION PROGRAMMING	Tony
SENSOR INTERPRETATION	James
GRAPHICAL USER INTERFACE	James

Table 9.3: Overall Assignment Schedule

Number	Task	Start	End	Duration (Weeks)	Task completed by
Senior Design 1					
1	**Brainstorming**	5/16/2016	5/23/2016	1	The Team
2	**Project Selection & Role Assignments**	5/23/2016	5/30/2016	1	The Team
Project Report					
3	Initial Document - Divide & Conquer	5/27/2016	6/3/2016	1	The Team
4	Table of contents	6/3/2016	7/1/2016	4	The Team
5	First Draft	6/3/2016	7/8/2016	5	The Team
6	Final Document	7/8/2016	8/2/2016	4	The Team
Research & Documentation					
7	Microcontroller	6/3/2016	6/24/2016	3	The Team
8	Power System	6/3/2016	6/24/2016	3	Phillip
9	LCD Display	6/3/2016	6/24/2016	3	James
10	Motion Sensors	6/3/2016	6/24/2016	3	Josh
11	Fire Sensors	6/3/2016	6/24/2016	3	Josh
12	Tilt Sensors	6/3/2016	6/24/2016	3	Josh
13	RFID Tag implementation	6/3/2016	6/24/2016	3	Tony
14	Wireless Communication	6/3/2016	6/24/2016	3	Tony

15	Software	6/3/2016	6/24/2016	3	The Team
Design					
16	Microcontroller	6/24/2016	7/15/2016	3	The Team
17	Power System	6/24/2016	7/15/2016	3	Phillip
18	LCD Display	6/24/2016	7/15/2016	3	James
19	Motion Sensors	6/24/2016	7/15/2016	3	Josh
20	Fire Sensors	6/24/2016	7/15/2016	3	Josh
21	Tilt Sensors	6/24/2016	7/15/2016	3	Josh
22	RFID Tag implementation	6/24/2016	7/15/2016	3	Tony
23	Wireless Communication	6/24/2016	7/15/2016	3	Tony
24	Software	6/24/2016	7/15/2016	3	The Team
25	Order and Test parts	7/15/2016	8/2/2016	4	The Team
Senior Design 2					
26	**Build Prototype**	8/22/2016	10/10/2016	7	The Team
27	**Testing & Redesign**	10/10/2016	11/7/2016	4	The Team
28	**Finalize Prototype**	11/7/2016	11/21/2016	2	The Team
29	**Peer Presentation**	TBA	TBA		The Team
30	**Final Report**	TBA	TBA		The Team
31	**Final Presentation**	TBA	TBA		The Team

9.2) BUDGETING AND COST CUTOFF

Our project was partially picked due to how affordable it was for most of us as students, as many of us already live on a fairly shoestring budget. This means there was quite a bit of leeway if errors arose with our selection in parts and that we could easily just choose new ones if an overlooked problem arose affecting us.

The part that was considered the biggest expense in terms of cost was the microcontroller and was the one that got reconsidered a few times. We were originally looking at a cost of 100$ for a development kit that would be specifically designed to be

integrated into a PCB. However, some issues arose not only around the cost but the implementation of it as it lacked a few features that were needed to make our project meet our goals. This turned out to actually be a bonus for us on the basis that we could use a non dev kit version of the integrated microcontroller. This lead to an even bigger cost reduction that dropped the expected cost from 100$ to approximately 50$ for the latest and greatest common use microcontroller on the market. In the end this change lead to approximately a smaller amount of pins, but still a very decent amount available to use for use.

For our LCD screen we set out budget very high for the item expecting it to be not only a hard to utilize part but also a part in which the cost would be fairly cost prohibitive. However, it turned out that even though we set our cost cutoff extremely high for the item, that it would end up being one of the cheaper items available to us. The selection of items available to use was extremely high, however, the ones that fit the criteria we were looking for was actually fairly low for the item in general. Our requirements we had set and were expecting was at a minimum a 3.5-inch screen with or without color at a cost of about 50$. This was factoring in shipping and such. In the end we decided to go with Amazon as we had prime shipping bringing the shipping cost to 0$ and ended up with a 3.5-inch color screen that supported a higher resolution than expected with a cost of only about 18$. This was the result of not being afraid to try an item from an off brand manufacturer by the name of Soyoo. The only consideration this brought in due to saving on cost was that the manufacturer ensured we had to use their specific software with the LCD if we wanted a less time consuming way to use the product.

The only other major cost for our project would be the specialized Buck Converters used within the project. We ended up using two of them in the end which resulted in a cost of about 25$ for two of them. This was virtually the only item on the market that met our exact specifications needed for our project. We had to look for one that had a very exact output of 2.5A 5V with a variable input voltage. The actual power system was based around utilizing this chip in particular and everything within it was designed around meeting the range of the particular chip we used. We checked to see if a lower costing alternative existed or if we should make our own, however, we found no alternative that met the exact specs and creating our own was too risky as it required a very exact voltage output within the system to avoid damage.

Essentially this covers bulk in terms of cost of our total project. There are lots of miscellaneous items that that added smaller costs. Things such as RFID readers, transmitters, receivers and USB cables. Many of these items were given to us for free or ones we had in reserve before the project was started due to kits we had for other projects out there. The following table shows how much money we allocated for our budget of all the items.

Table 9.2.1: Total Budget for Items

Part	Price
Microcontroller	$50
LCD	$50
RFID	$40
Keypad	$10
Fire Sensor	$50
Transmitter/Reciever	$30
Tilt Sensor	$30
Motion Sensor	$30
USB	$5
Diodes	$20
Regulators	$10
LED	$10
Capacitors	$10
Misc/Wires	$40
Total Cost	$385

9.3) EXPENSES

This section is a summary of all the parts used within the project and the actual paid value. It also indicates the amount received even if not all parts were utilized within the project due to only coming in certain numbers of parts. In essence, to reproduce the system this is what the exact cost would be without trade deals.

Table 9.3.1: Actual Cost of Parts and Shipping

Part Name(Amount)	Cost $
MGT-1220(1)	$12.90
1N5408(10)	$4.23
D24V25F5(2)	$25.85
Raspberry Pi III	$35.00
1506 LCD Module	$17.99
JianHan USB (2)	$0.00
Transmitter/Receiver	$8.99
Tilt Sensor	$9.99
Motion Sensor	$8.99
RFID	$42.99
Transistors	$0.00
Nichicon E-VZ (2)	$8.84
LED	$10.35
Pushbuttons	$9.98
Fire Sensor	$6.99
Misc/Wires	$59.41
Total	$262.50

9.4) PART LIST AND SERIALS

This section contains information on the exact part numbers or the serial of the respective parts in the event of needing to reorder them. There is also a description of what the exact part does when it comes to utilizing the system for use.

9.4.1: List of Parts and Serials

Part Name	Description
D24V25F5	Buck Converter(DC->DC)
MGT-1220	AC->AC Transformer
1N5408	Diodes
Raspberry Pi III	Microcontroller
1506 LCD Module	LCD Screen
Nichicon E-VZ	Capacitors
IL112	LED
Gikfun EK1019	Pushbuttons
Gikfun EK1042	Tilt Sensor
HC-SR501	Motion Sensor
IR Flame Sensor	Fire Sensor
2n2222	Transistors
JianHan USB	USB Cables
RF 433	Transmitter/Receiver
Generic 24 Gauge	Wires

10) SOURCES

*For Section 3.1.1 http://www.electronics-tutorials.ws/diode/diode_6.html

*For Section 3.1.2 http://www.circuitstoday.com/simple-ups-construction

*For section 3.3.12 http://www.itarticle.net/wp-content/uploads/2012/05/The-USB-port.jpg

11) COPYRIGHT PERMISSION

*Full Wave Rectifier Design from http://www.electronics-tutorials.ws/diode/diode_6.html Status - Approved

*Continuous UPS design from http://www.circuitstoday.com/simple-ups-construction Status - Approved

*RFID chip configuration from http://www.

Electronics Tutorials (webmaster@electronics-tutorials.ws) Add to contacts 8:39

To: 'Phillip Dannelly'

Hello Phillip,

Thank you for you email and question.

It is normal practice to ask permission first before copying items from the internet.

However, as you have now been asked by your tutor to gain permission I will grant it if used as part of your senior design project.

I must therefore insist that you correctly reference my tutorials, images and site: www.electronics-tutorials.ws accordingly within your project and presentations.

Kind regards.
Wayne Storr
webmaster@electronics-tutorials.ws

-----Original Message-----
From: Phillip Dannelly [mailto:pdannell@hotmail.com]
Sent: Thursday, July 14, 2016 9:50 AM
To: webmaster@electronics-tutorials.ws
Subject: Permission Of Use

Sent from the contact form on Basic Electronics Tutorials:

From: Phillip Dannelly <pdannell@hotmail.com>
Subject: Permission Of Use

Message:
Hi, I used your design as a reference in my senior design project from the

Re: Permission to use content.

CircuitsToday Store (info@circuitstoday.com) Add to contacts 5:57 AM

To: Phillip Dannelly

Hi Philip,

You can use the content after giving a proper Link Back to the page. If you are publishing it in any web page of yours or university, proper back link should be given there as well.

Please lets us know after linking.

regards,
Jojo Joson
CircuitsToday

On Thu, Jul 14, 2016 at 1:21 PM, Phillip Dannelly <pdannell@hotmail.com> wrote:

> Hi, I used your design as a reference in my senior design project from the following page.
>
> http://www.circuitstoday.com/simple-ups-construction
>
> It is being used as Education/Non-profit use and I am required to ask if you have any objections and if it is copyrighted or if the design is public use?
>
> -Thank You
> -Phillip Dannelly

Cecilia: Hi Tony
Tony: Hello
Cecilia: Thank you for contacting NXP
Commercial **Cecilia:** How may I be of help?
Tony: I am working on a project for school and I am implementing the PN532 breakout board that your company designed and in the datasheet there is a schematic I would like to use for my report. Is there any legal permission I need to be able to add it?
Cecilia: I'm checking with our engineering
team **Tony:** Okay thank you
Cecilia: basically if the information that you are going to use, can be found on our website **Cecilia:** this means it's not confidential
Tony: Okay thank you so much
Tony: I'll just give credit to the company
Cecilia: yes, you can reference us
Cecilia: just double the information
Cecilia: and documentation
Cecilia: that you want to use

Re: Query
2 messages

Engineers Garage <support@engineersgarage.com>	Thu, Jul 21, 2016 at 3:07 AM
To: James Tavil <toukwek@gmail.com>	

Hi James,

Thank you for connecting with us.

As long as you are mentioning the circuit source from Engineersgarage.com on your website, you can utilise the respective circuit.

Regards,
Team EG

On Wed, Jul 20, 2016 at 11:54 PM, James Tavil <toukwek@gmail.com> wrote:
Submitted on Wednesday, 20 July, 2016 - 23:54
At time of Submission User IP was
199.87.224.67 Submitted by user: Anonymous
Submitted with
Name: James Tavil
Email: toukwek@gmail.com
Subject: Query
Message: Hi, I'm requesting a permission to use a diagram for my
senior design project on the website at this address
http://www.engineersgarage.com/articles/what-is-tilt-sensor?page=4.

The results of this submission may be viewed at:
http://www.engineersgarage.com/node/130/submission/7313

James Tavil <toukwek@gmail.com>	Thu, Jul 21, 2016 at 11:10 AM

To: Engineers Garage <support@engineersgarage.com>

Thank you

On Jul 21, 2016 3:07 AM, "Engineers Garage" <support@engineersgarage.com> wrote:

Hi James,

Thank you for connecting with us.

As long as you are mentioning the circuit source from Engineersgarage.com on your website, you can utilise the respective circuit.

Regards,
Team EG

On Wed, Jul 20, 2016 at 11:54 PM, James Tavil <toukwek@gmail.com> wrote:

Submitted on Wednesday, 20 July, 2016 - 23:54
At time of Submission User IP was
199.87.224.67 Submitted by user: Anonymous
Submitted with
Name: James Tavil
Email: toukwek@gmail.com
Subject: Query
Message: Hi, I'm requesting a permission to use a diagram for my
senior design project on the website at this address
http://www.engineersgarage.com/articles/what-is-tilt-sensor?page=4.

The results of this submission may be viewed at:
http://www.engineersgarage.com/node/130/submission/7313

atomic <atomic@atomicmarket.com> Jul 30 at 11:03 PM

To jfry321@yahoo.com

On 2016-07-30 23:20, Joy - Atomic Market wrote:
> Name: Josh Fry
> Email: jfry321@yahoo.com
> Telephone:
>
> Comment: Hello, I am required to ask permission by my university in
> order to use your fire sensor for our senior design project. This
> project is non-profit and will be for educational purposes only.

go right ahead.
regards
shane @ atomic market

Reply, **Reply All** or **Forward** | **More**

12) APPENDICES

12.1) DATASHEETS

This section contains a list of datasheets or available data in terms of parts used within the project.

Part Number	Link
D24V25F5	https://www.pololu.com/product/2850/specs
Raspberry PI	https://www.raspberrypi.org/help/faqs/#power
4700 uF VR Series	http://www.nichicon.co.jp/english/products/pdfs/e-vz.pdf
1N5408	https://www.fairchildsemi.com/datasheets/1N/1N5408.pdf
TX433	http://www.vellemanusa.com/downloads/7/tx433n_datasheet.pdf
RX433	http://www.apogeekits.com/PDF_Files/RF_Receiver_Module_Datasheet.pdf
Motion Sensor	https://www.mpja.com/download/31227sc.pdf
MGT-1220	https://www.amazon.com/gp/product/B002LG8008/ref=oh_aui_search_detailpage?ie=UTF8&psc=1