Senior Design I

Final Document

G.A.R.V.I.S.

Gesture-Controlled Automated Residency Via Intelligent System

Group 23

Joshua Illes

Andres Mujica

Jackson Schleich

Sarah Strauss

TABLE OF CONTENTS

1 EXECUTIVE SUMMARY

GARVIS, or Gesture-Controlled Automated Residency Via Intelligent System, is in essence a home controller using an adaptive approach to improving home energy use patterns. This implies that the system will take as much data as possible regarding the energy use habits of the home owner, and apply intelligent analytics to that data with the end goal of controlling the home in a way that will reduce home energy consumption without disrupting the lifestyle of the home owner.

The GARVIS system is created to address two fundamental problems in the way that people control the energy usage within their own residences. First of all, homeowners rarely have complete and convenient access to information regarding detailed energy consumption of their residence. At most, the homeowner is capable of seeing one temperature; the temperature of the air immediately surrounding their thermostat. This information is not very telling regarding the temperature, and therefore the energy consumption, of the house as a whole. The second problem is that current smart thermostat solutions are not able to control anything except the on/off status of the heating ventilation air conditioning (HVAC) unit, which gives even the smartest of systems little control over the utility of the hot or cold air being pushed into the residence. On a similar note, systems are not able to control lighting devices to respond to user habits or current lighting needs.

The main components of a system that can accomplish this goal are: distributed sensors capable of gathering useful information about home energy consumption, devices capable of changing energy use throughout the house without relying on the home owner to do anything, and a main controller capable of storing user data, performing analytic algorithms and communicating effectively with both the home owner and the system peripheral devices.

The system to be implemented by the group involves the following approaches to addressing the problems described. First of all, a central controller was designed with enough processing power to handle the analytics of the user data. The central controller will also improve the user interface capabilities to make an easy and convenient manner of controlling the household preferences and providing energy usage feedback. Secondly, the system will have sensors distributed throughout the residence to provide more detailed household energy usage data. These sensors will be packaged within household light switches, and will communicate back to the central controller via power line communication. Addressing the issue of improved energy control will be two devices, one for outlet electricity control and the other for HVAC air control. Controlling the electricity will be a load controller device, which will be capable of communicating with the main controller and reducing or cutting the electricity draw of various household devices. Controlling the HVAC air control will fall to an off-the-shelf electrically controlled air duct. The final benefit is to explore gesture control as a way to improve user interface within the household. Gesture control has already

been proven as a viable way to interface with entertainment systems and mobile devices, so applying the same technology to the house as a whole is plausible.

The user interface is a large focus of this project. The user will have a number of interface options for controlling their house. The most obvious will be the home controller, which will have a touchscreen interface and will provide options for changing temperature in various parts of the house, and will provide real time feedback regarding energy usage. Another user interface will be the smart switches housing the distributed sensors. The homeowner will have a simple user interface to control the environment of that specific room in a quick manner. The homeowner will also have the ability to control the house from a mobile application and web server interface.

Overall, the project aims to address a very broad problem of home energy usage through an approach that involves taking better data from around the house, providing better automated control of the household environment, and interfacing with the homeowner in a simple, convenient manner that allows the home controller to make the best decision regarding energy usage, while still understanding the user preferences. When completed, GARVIS will be a prototype yet usable version of a complete home control system.

2 PROJECT DESCRIPTION

2.1 PROJECT MOTIVATION

A senior design project is meant to give senior engineering students the opportunity to learn about how to work with a team to research, design, develop, build, integrate, and test a system as it is done in the industry. As a team we have multiple motivations for creating and developing our project, GARVIS. We want to provide innovation to the home automation industry, create a means of control based on gesture recognition, learn about many new technologies and ideas, and ultimately create a capstone project that will finish out our bachelor of science careers at the ABET accredited University of Central Florida.

As a place where people live, sleep, eat, and work a home is arguably considered the most important place for a human. Many people cater their lifestyles to their home but we aim to reverse this and enable a person's home to cater to their ever changing lifestyle. We are achieving this by creating an intelligent home system that goes beyond simple timers and is able to adapt and learn a person's habits and needs. In addition, this system is able to focus on energy efficiency by filling the gaps where humans have frequent errors. It functions as a plug and play system that requires less installation and more customizability.

What sets our automation system apart is its user interface and control system. We want the user to have the ability to control their house from the palm of their hand. A custom glove will allow the user to use intuitive gestures in order to change settings, turn items on and off, and gain status information about their home. This glove will have the capability to interface with multiple types of hardware and can have additional uses outside the scope of a home automation system. Not only would they be able to interface to the system via the glove—but also in a secure application accessible from one's phone, computer, or tablet in any location with internet.

The team to accomplish this task consists of four very ambitious and motivated students—three electrical engineers and one computer engineer. This project was chosen because it has a very good variety of work involved that will provide all team members an ample but achievable challenge. It also integrates two very important concepts we have agreed we want to get experience with—gesture recognition and home automation. We have made sure to incorporate a good balance of technology we have worked with in the past and the new technology we want to learn about. This idea has provided us with a lot of new design experience for PCB boards, hardware systems, and multiple types of software systems.

In order to complete our education successfully we must be prepared for the industry by meeting the Accreditation Board for Engineering and Technology (ABET) criteria. This includes getting major design experience that is supported by what we have learned in previous classes and incorporating engineering

standards in a capstone project. We also must deal with realistic constraints and multiple considerations that span from environmental and sustainability to the ethical and political side. This is a very large motivation for our project and has allowed us to have a clear vision of what we hope to achieve from our senior design.

2.2 GOALS AND OBJECTIVES

Our goals for the home automation system are to make it affordable, easy to customize, adaptable to a person's habits and energy efficient. We want it to be very user friendly and have plug and play functionality for an easier set up than the tradition home automation system. It will have a central controller for setup, the addition of new appliances to control, and customization along with a mobile application to remotely control the home.

The objective for the glove is to ultimately enhance the home automation system control process. It will do this by having a sleek system design and a low latency response. It will utilize gesture recognition from multiple sensor inputs and transmit pertinent information to other the home automation system for a new means of control. In addition to this, it will interface to other outside hardware such as the Oculus Rift through the creation of special API libraries.

2.3 BLOCK DIAGRAM

This system is composed of 4 main parts. The Central Manager, the Smart Switch, the Load Control, and the Wireless Glove. The brain and controller of the whole system is the Central Manager. This is the first figure shown here. The central manager has a Cortex A8 processor on it that will allow to control multiple things. The communication as master over Power Line Communication in order to get information from the smart switches along with controlling the load controls. The server which is going to have information stored about the home. Using algorithms to use the information on the server to make smart decisions reducing the overall power consumption of the house. It will control your HVAC system. It will control the LCD screen that has the user interface on it to control the home. It will communicate wirelessly with the glove in order to have a gesture based control. Along with this the central manager will be powered straight from the 110 VAC lines in your home.

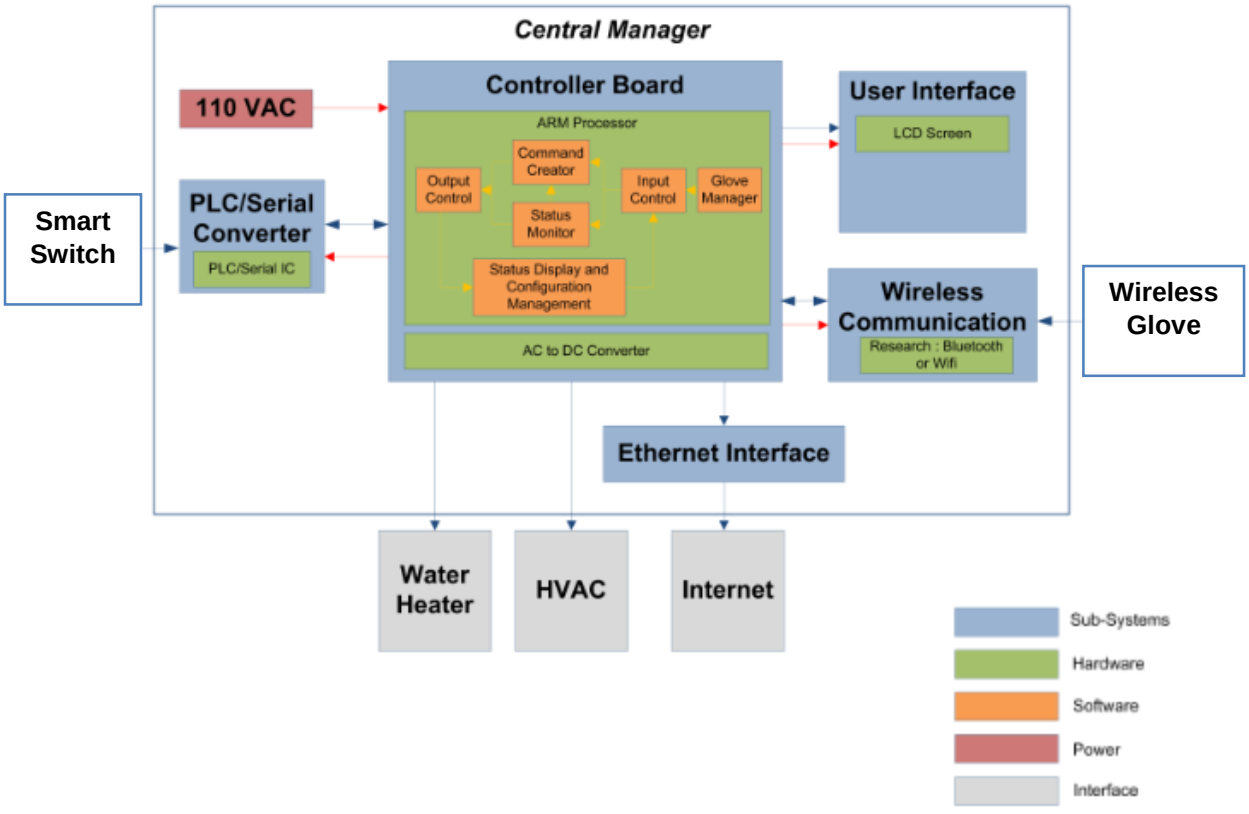


Figure 1: Block Diagram Part 1

The Smart Switch is responsible for getting useful data through sensors. It is going to record temperature, motion from passive infrared, light, and touch. The advantage of our system is that there will be multiple smart switches. So temperature won't be recorded from just one place. Along with the automated vents in the system the house will be able to have every room at the right temperature as opposed to just one room. The Smart Switch also has a capacitive touch interface for the user to have control over things in the room like the lights, fan, etc.

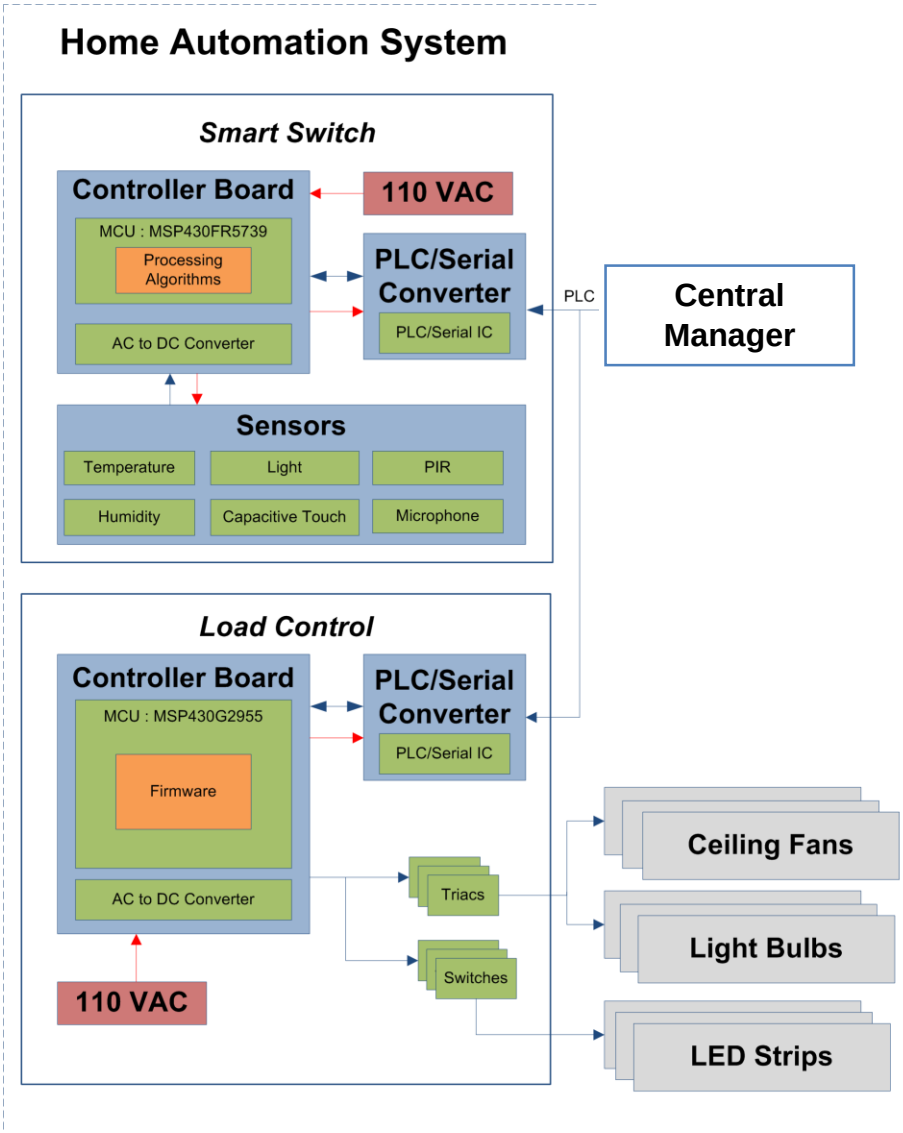


Figure 2: Block Diagram Part 2

The Load Control is in charge of actually controlling the house through an analog interface. It will be composed of triacs and switches to provide full control of lights and fans while also having the ability to dim the lights and control the fan speed. This also has the added bonus of reducing power consumed.

The Wireless Glove is responsible for a gesture based control system. It will take input from an accelerometer, gyroscope, magnetometer, and flex sensors in order to get kinesthetic data from the user. This will be sent through Bluetooth where the raw data will be processed into actual commands. It will be powered through LiPo battery.

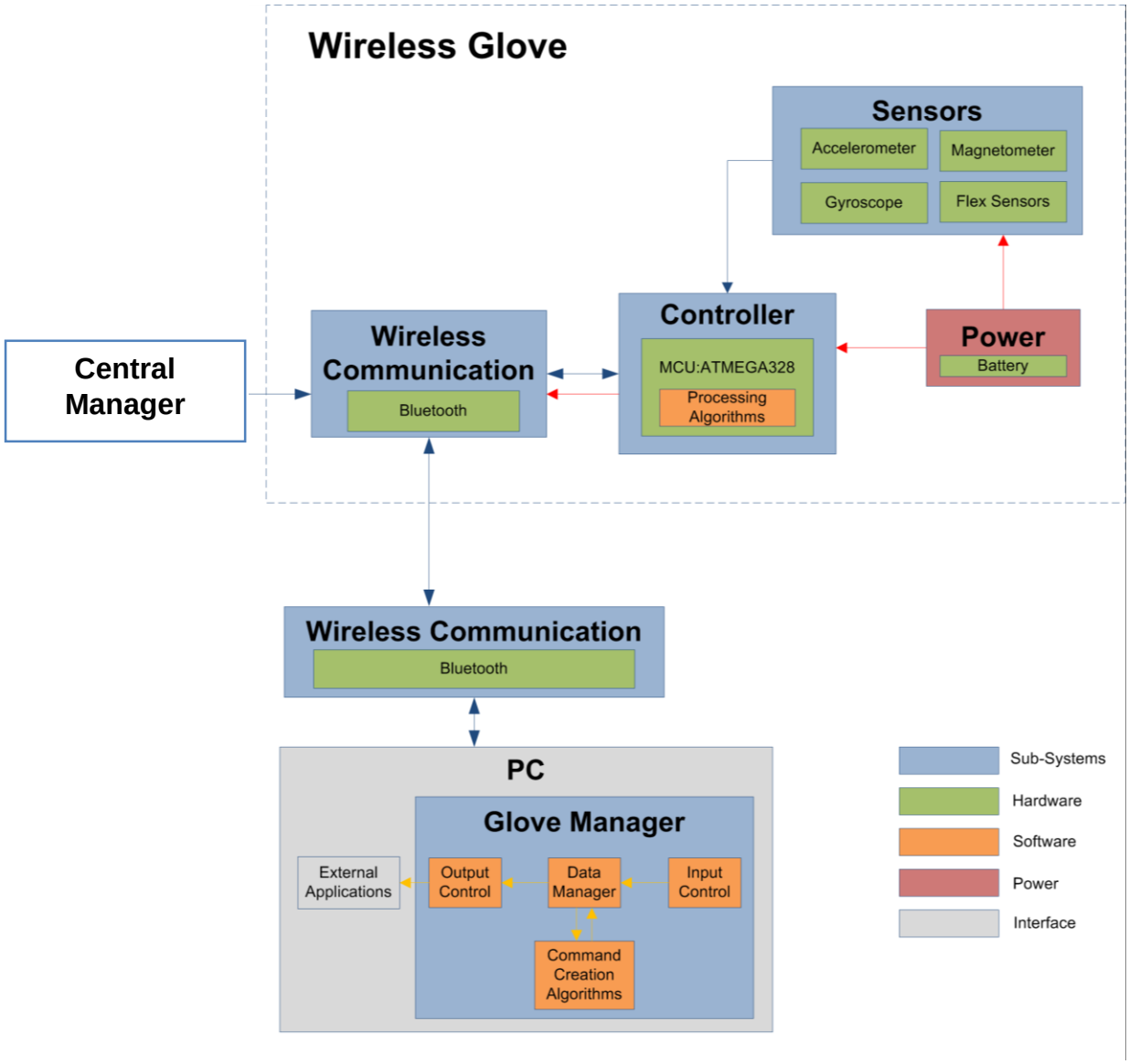


Figure 3: Block Diagram Part 3

3 RESEARCH RELATED TO PROJECT DEFINITION

3.1 EXISTING SIMILAR PROJECTS AND PRODUCTS

3.1.1 Automated Residencies

The automated residency market has skyrocketed since the current technology has allowed for high affordability and simplicity through the use of smartphones and other devices. For decades the idea of a self-running, self-sufficient, and easily controllable home has been written about but only recently has technology caught up with the creative writers' dreams. Today the automated residency market is very large and all different types of companies have tried to get a foothold on it. The main issue with the currently existing home automation systems is that there is no universal means of control or protocols for interfaces so one system very rarely works with another system. Another large concern is the amount of initial setup necessary to get an automation system up and running. Some pre-existing systems use additional control wiring that needs to be installed in the house, others use RF communication, communication with wires already existing in the house, or wirelessly. Our system works to combat this second concern by providing a plug and play functionality with PLC where there is no additional wiring for any new appliances that are going to be controlled.

Currently, the goals of an automated residency range from providing lighting controls, audio and visual controls, shading controls for temperature and brightness, security, intercom system control, appliance control, and domotics. Our system's goal is to provide a means of control for multiple of those features in a home via a Central Manager. Some companies focus on a single aspect of control while others try to allow for the user to control their entire house from a single point. Systems such as the products from Lutron Electronics put a strong focus on a single aspect, in this case lighting, and work on creating an automated lighting system but forgo any other sort of control for the home. Honeywell's system allows for the customization and control of many aspects of the home such as lighting, temperature, fans, and entertainment systems but requires the installation of additional control wiring [1]. Our system will try to achieve what Honeywell is doing but without needing additional wiring that can cost the users a lot in installation and maintenance. Two of the main automation systems will be further analyzed and built off of in the development of our automated residency system, Lutron Electronics and X10.

3.1.1.1 Lutron Electronics, Quantum

Lutron is a lighting company that deals with customized and automatic lighting systems [2]. The Quantum is an energy management and lighting control system created by Lutron which includes but is not limited to lighting controls, LED drivers, motorized window shades, and sensors. This is all possible with an array of sensor, control and mechanical hardware along with advanced management and user interface software. The sensors used on this system

include occupancy sensors, ambient light sensors, and shadow sensors. The control hardware includes both wired and wireless sliders, push buttons, and keypads. The mechanical portion is used to control the drapery and shading schemes in a given area. Software for this system is very versatile and includes a management software called “Quantum Vue”, software for energy usage, multiple applications for remote statusing and controlling, and a PC/Web based software package.

This lighting system is on the very pricey end of the spectrum, which is something our project is trying to avoid. Getting the Quantum installed would cost a user thousands of dollars because of the complexity of the system, the necessity of a server, and the necessity of new wiring for any part of the building that will use the Quantum.

The communication of the system takes on many forms depending on the hardware entity. It uses RS485 for the QS link and power panel link, interprocessor Ethernet, a building’s preexisting Ethernet network, and wireless RF. The software applications communicate using the building’s Ethernet network and they transmit data to the Quantum server in the same manner. The Quantum server transmits to the Quantum hub via RS485 and once the Quantum hub has done the necessary processing the command data is sent to dimming and switching panels. Those panels will command loads of all types such as in auditoriums, bathrooms, kitchens, etc. also via RS 485 in the power panel link. Once installed this system will add additional lighting features to a building and improve upon preexisting lighting through this complex command and control scheme. One large downside of this system is that it only controls lighting in a building when it has so much potential to control the HVAC system, other appliances, and other loads.

Notable features:

- Time based control where the user can set up a calendar of events to determine lighting needs
- Energy savings by only allow lighting to be active when it is being used
- Smart Grid awareness to lower lighting levels when warnings are given from utility companies to further save money
- Data is sent to the Quantum server and Lutron for analysis for lighting management so it is not locally hosted

3.1.1.2 X10

The X10 company is a well establish home automation team that has many switch, controller, motion sensors, security, and other products that allow for plug and play functionality [3]. A wireless controller sends RF signals to a transceiver module which will send PLC signals to an appliance module to control that appliance. They also have a wired version of control that can automatically send signals to control appliances on a timer basis. Their products range from about $10-$50 dollars depending on the functionality of the item but a system for about 5 appliances would cost the user about $150 to set up.

X10 also has control and setup software called ActiveHome. It can be used to set up timers, to run tests on hardware to find bugs, to turn appliances on and off, and to create macros. It is free to install and use but the X10 company does not offer technical support for it.

The important properties to note for the X10 technology are:

- Modular to allow for easy set up and the addition of extra appliances at any point
- Controlled via RF signals or from a wired control module with timers
- Functionality allows for home security, monitoring, and control
- Utilizes power line communication

3.1.2 Gesture Control Systems

3.1.2.1 Keyglove

The Keyglove is a wearable hand gesture input device that is open source and can interface with many gaming, art, music, data or text entry, and other applications along with provide mouse and keyboard control [4] [5]. The mouse is controlled from the hands movements while the keyboard is controlled by touching certain parts of the fingers on the glove's hand. The software is open source so it has been very helpful to our research and development efforts.

This glove can wirelessly control many computer applications through Bluetooth and a specialized API that uses the standard HID protocol. The glove is detected automatically by the host computer as a mouse, keyboard, and joystick without any drivers because of the HID protocol that it follows when sending data via Bluetooth to the computer. It also uses a mini-USB connection for charging and configuration. It has been interfaced with the Oculus Rift and Google Glass proving that this sort of communication is realistic for us to accomplish with our gesture recognition system.

The gesture recognition part of this glove works based on the input of rotational and linear motion detections from a 6 or 9 DOF board, depending on which configuration is being used. In the first iterations of the Keyglove Arduino products were used for the on glove processing but currently a customized board is in the process of being designed and built. Most of the design schematics, wiring, and building instructions are publically available for this product.

Notable features:

- The glove has been constructed for both a left hand and a right hand
- The controller board and other components are removable from the glove very easily for upgrading
- There is vibration, light, and audio feedback mechanisms on the glove
- Glove API available to make interfacing to devices very easy and enable developers to very quickly make applications that can interface with the glove

3.1.2.2 The Sign Language Glove

The Sign Language Glove was created in 2013 by a team of four engineers at the University of Central Florida. The goal of this glove was to create a user friendly means of receiving sign language information and interpreting it on an Android device. By using strategically placed sensors to determine hand shape, movement, orientation of the palm, and hand location the glove was able to send hand data to an Android device via Bluetooth [5]. The Android device would then parse the data and process it with gesture control algorithms to determine which symbol the user was gesturing. This glove is capable of interpreting all the sign language letters and numbers but their gesture library doesn't go into any more depth in terms of other symbol recognition.

The glove hardware consist of an IMU, flex sensors, and pressure sensors to determine current hand traits as well as an ADC, MCU, Bluetooth client, battery and voltage regulator. Since their gesture sensing technology is very similar to ours, we can use their research and further develop gesture recognition. The other main component of the system was the Android application to receive, parse, and process the gesture data.

An issue this team encountered in development was the inability to set universal boundaries for all hand sizes and shapes. This is something we need to do further research in and decide how we would like to approach it with our gesture recognition system.

3.2 RELEVANT TECHNOLOGIES

3.2.1 Communication

3.2.1.1 UART

UART stands for Universal Asynchronous Receiver/Transmitter. It is a very common serial protocol used between digital devices. It is asynchronous therefore both devices communicating must have some predefined parameters including baud rate (the speed at which the devices send complete messages), number of data bits (from 5 to 8), whether there is a parity bit, and whether to have 1 or 2 stop bits. [6]If both devices don't have matching parameters then the devices will not be able to get data through to each other.

UART is a full duplex protocol therefore it has two lines involved with it. A Transmitter and a Receiver line. Otherwise known as Rx for Receiver and TX for transceiver. The Transmitter line of one device will go to the Receiver line of the other and vice versa. When the devices aren't transmitting the line is Idle and is left high. A start bit (which signifies the beginning of a message) is when the line is pulled low. This is then followed by the data bits, the parity bit if there is one, and then the stop bits, then the line is left high again until next start bit.

The positives of using UART is that it is such a common interface to use. It is extremely easy to implement and most digital devices already come with the hardware for UART built in. Even if you don't have the hardware it isn't hard to

get a software emulation of UART to throw in to a microcontroller or processor. Therefore it is pretty easy to have multiple UARTs in. In order to have other UARTs you just have to have another two lines. Along with this if you only need communication one way you can strip out the other line and just use one.

The downside to using UART is that it does require two additional lines every time you want to implement a full duplex protocol there. This can add up very quickly. Along with this the asynchronous part of the protocol makes it so if you want to have good signal integrity you have to oversample the signal. A good standard for these protocols is that your device should be able to sample every bit at least 8 times. This means that the fastest UART you can run on your board would be the max speed of your clock divided by 8. Along with this it can be hard to debug your UART lines. If for some reason the parameters don't match, you might end up putting an oscilloscope probe and the line looks fine but your device isn't getting in data.

3.2.1.2 PWM

PWM stands pulse width modulation. PWM is a method of using a digital signal, which only allows for either a high voltage or a low voltage to be transmitted, in order to transmit data with time modulation instead of serial messages. In order to understand PWM we must understand the concept of duty cycle. In a digital signal or square wave the duty cycle is the percentage of the time the signal is high in a fixed period of time. So as the signal repeats itself at a constant frequency the amount of time the signal is high over the period is the duty cycle.

By modulating the duty cycle of a signal you get a PWM signal. By modulated the width of the pulse. A pulse being a digital signal with a single rising edge and falling edge during per period. Each different width of time for the pulse represents a different message. The single unit of time that this pulse can be increased or decreased by is the resolution of the PWM signal. So the higher the resolution the smaller the time increments that the pulse can be modulated by, therefore the more messages that can be represented.

This protocol can be used for applications like dimming lights and digital control of things like motor speed and servo motor angles.

3.2.1.3 PLC

3.2.1.3.1 Background

Power Line Communications (PLC) is a technology that allows for data to be transmitted over existing power lines. This is advantageous because it requires less copper wire and the labor to retrofit existing systems which reduces overall project cost. Power companies have been implementing PLC since the beginning of the 20th century [7] to harvest data from distant substations. In the 80's the first PLC devices were released commercially. These devices used protocols such as X-10, CEBus, and LonWorks. With the Internet of Things (IoT) developing so rapidly, it makes sense to have products that are plug and play into existing fixtures and do not require new data lines to be run. Many of the

new IoT devices use Wi-Fi however wireless communications has its known drawbacks such as connectivity issues. Power line communications is broken up into two sub categories based on frequency and application; narrowband PLC and broadband PLC.

3.2.1.3.2 Narrowband PLC

Narrowband PLC is the communications that occur at lower frequencies (3-500 kHz) which translates to lower data transmission rates (≈100kbps.) The advantage of these slow frequencies is their range which has been tested to a range of more than 500 km [8]. Narrowband PLC utilizes single-carrier modulation which requires lower peak to average power ratio (PAPR) and less complexity compared to its multi-carrier counterpart.

X10

3.2.1.3.2.1 X10 is both an industry standard communication protocol for power line communications and a home control company. It was one of the first in existence and was developed in 1975 by Pico Electronics.

X-10's notable feature is it synchronizes with the zero crossing of the AC sine wave. Whenever the mains voltage is at zero, a bit is sent. A bit is represented by a 120 kHz burst lasting 1ms [9]. The presence of at 120 kHz burst represents a binary 1, an absences represents a binary 0. Since there is two zero crossings per sine wave period, X10 can send 120 bits/ second on the standard US mains 120VAC/ 60 Hz. The transmission is broken up in the following manner:

Start Code				House Code								Function/ Number Code									
1	1	1	0	0	1	1	0	1	0	0	1	1	0	1	0	1	0	0	1	0	1

Table 1: X10 Message Bitfield

The start code is always a binary 14 and the house code which is an address to intended device and the function code is what the operation is being performed. With the house code and function/ number code, there is an alternating transmission pattern. That is, to send a binary 1, on the positive edge of the sine wave a high signal is sent, on the falling edge, the inverse, a low signal, will be transmitted. This complementary transmission scheme is a way to do quick error checking. In between the transmission, there is 3 cycles of silence. Do to the complimentary transmission scheme; the 8 bit house code and the 10 bit function code are actually only 4 and 5 bits, respectively. With 2^5 potential

3.2.1.3.2.2 functions, X10 is limited to 32 operations, one of its downfalls. Another downfall of X10 is it is slow at essentially 60 bit/sec. The advantage is it is very simple and robust.

CEBus

CEBus stands for consumer electronics bus and is a standard for communications in consumer and residential networks. It covers implementation for radio frequencies (RF), infrared (IR), twisted paired wire, coaxial cable, and communications over power lines. The basis of CEBus is that a new network

must be: low cost, universal among manufactures, utilize high quality cabling and reliable [10].

CEBus utilizes an addressing scheme to point to the desired address. Since power line communications in general stretch beyond the single residence since there are upwards of ten residences tied to a single transformer. The addresses are all individual, unique and have been hardcoded by the factory. The address field is 32 bits long which translates to more than 4 billion unique addresses.

The electrical network inside of residential and commercial buildings is not very noisy and not conducive to communications. CEBus utilizes a technique known as spread spectrum signaling. Spread spectrum signaling is doing a frequency sweep as transmission of data. That is for a sending a bit, the transmitter transmits a frequency that ranges from 100 kHz to 400 kHz. The CEBus standard is very strict on this frequency range. Inside the frequency band the transmission levels is between 2.5V and 7V. Outside of the frequency band the transmission level must be below 5mV.

The receiver side looks for the initial stream of pseudo random numbers called the preamble. Once the receiver sees a certain amount of the anticipated pattern, it “locks on” and synchronizes because the transmissions occur every 100μS. With the spread spectrum signaling, reliability is high, 98% of the original signal transmitted is received and on the second attempt nearly 100% of the data is transmitted. Spread spectrum signaling is robust because it caters to both of the pitfalls in the power grid; attenuation and noise. With the drastic impedance mismatches that can occur in the power network comes large signal attenuation. CEBus receivers can detect a signal that is as low as 1.25V. The spread spectrum is optimal because if the network filters out a certain frequency, the transmission has many other frequencies capable of reaching its destination.

3.2.1.3.2.3

Lon Works

Lon Works, developed by Echelon Corporation, is one of the first standards in narrowband PLC for low data rates. It is known as ANSI standard 709.1. The standard was issued in 1999 and became industry standard in 2008 [8]. Lon Works is a narrowband PLC modulation that operates in the 115-132 kHz frequency range. Lon Works utilizes the seven layer Open System Interconnection protocol. The data transmission rate is up to 6 kbps.

3.2.1.3.3 Modulations

3.2.1.3.3.1

There are different techniques to encode the digital data to send it over a medium, these techniques are referred to as modulation schemes. Since we are going to brief over a couple different options we have for our signal modulation.

Frequency Shift Keying (FSK)

FSK is a modulation scheme that encodes data through varying frequencies. The advantage is the data is relatively resistant to noise. FSK is a commonly used modulation scheme for cheap radios [11]. Binary frequency shift keying (BFSK) is the simplest type of FSK, being that there are two transmission

frequencies. The first being the frequency that represents a binary 1, known as the mark frequency. The second frequency represents a binary 0 and is referred to as the space frequency. Both the mark and the space frequencies are centered on a center frequency. FSK utilizes a voltage controllable oscillator to be able to quickly switch between frequencies corresponding to the digital levels.

A specialized version of FSK is known as Minimum Shift Keying (MSK.) MSK is specialized because the mark frequency is half that of the space frequency. This enables for easier continuous phase when switching frequencies because the two are semi-synchronized. MSK is also particularly resistant to noise. We have the ability to use FSK with our analog chip via a DAC input

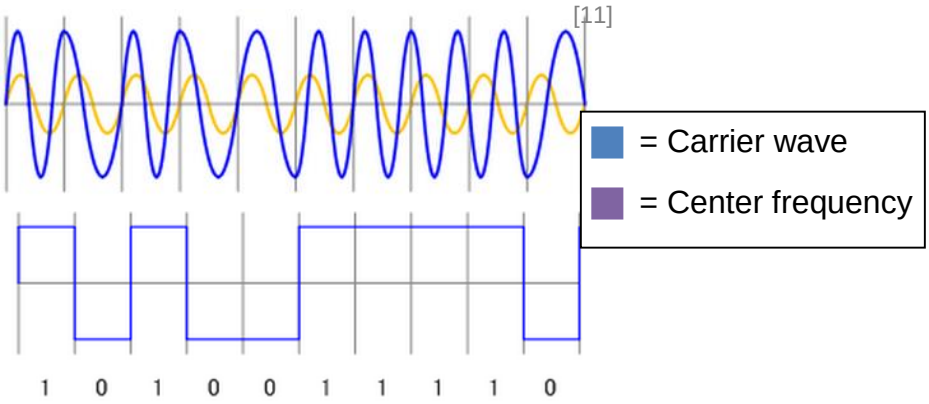


Figure 4: MSK Modulation

3.2.1.3.3.2 ***Phase Shift Keying (PSK)***

Phase modulation is less frequently used than frequency or amplitude modulations because it is more expensive to implement, however it is more reliable. As the name alludes to, PSK is when the phase of the wave is shifted by a set amount based on the modulation signal. The easiest type of PSK to understand is binary PSK. With binary PSK, the signal is not shifted with a 0 binary input and shifted by π radians when the modulation is a binary 1. Figure 6 depicts phase modulation very clearly. The disadvantage of the phase modulation is that both the sender and the receiver have to be synchronized which is why PSK is less common and more costly [11]. The synchronization allows for the sender to know when it should be sampling to determine if the phase has shifted or not. All in all, PSK is reliable, however it is expensive and more complicated to implement, so we are not going to utilize it in our project since our analog chip does not support it.

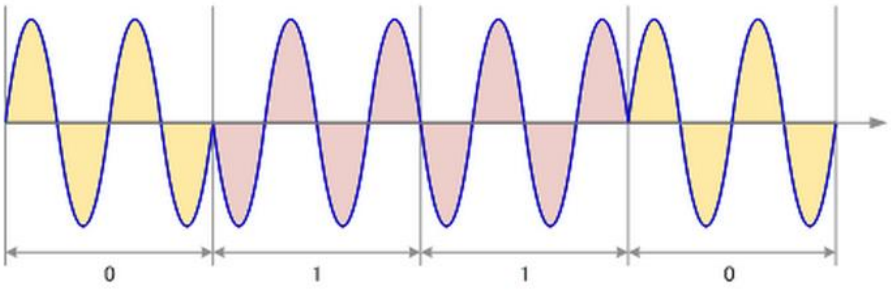


Figure 5: Phase Shift Keying

[11]

Amplitude Shift Keying (ASK)

3.2.1.3.3.3

Amplitude phase shift modulation is when the amplitude of the signal is changed in respect to a digital input. ASK modulation has a center carrier frequency and the modulation signal. The modulation signal and carrier frequency are summed together so that if the modulation is a binary 1, the amplitude will be higher than if it is a binary 0. Figure 7 depicts amplitude modulation very well, the data would be the modulator. ASK circuits are relatively cheap but not long range so they are used in cheap RF consumer goods such as keyless entry for a vehicle, and RFID systems. Since ASK is cheap, and easy to use, we are going to implement this modulation scheme for our PLC transmission since our analog chip supports it through two wire transmission interface.

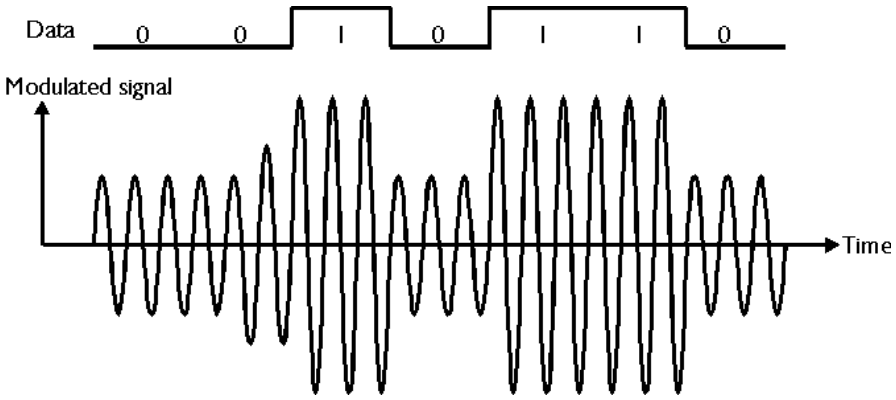


Figure 6: ASK Modulation

[12]

3.2.1.3.4 Broadband PLC

Broadband PLC is the high frequency, high data rate version of PLC in comparison to narrowband PLC. Broadband PLC, however, is limited in range. The data rates on broadband networks can be so high, they can transmit internet and stream HD videos over the power lines. Broadband PLC is used primarily as a “last mile” solution for ISPs to deliver internet services to their customers [13]. In older homes where Ethernet is not natively wired and the labor cost to install is prohibited, broadband PLC becomes a solution to deliver internet service form

the modem to any electrical plug in the house. One of the most common standards in broadband plc is known as HomePlug and is governed by the HomePlug Powerline Alliance. Broadband PLC uses multiple carrier frequencies to allow for faster data rates. Since broadband PLC has more complicated encoding schemes and high data rates than necessary for our application, we will be utilizing narrowband plc schemes.

HomePlug

Founded in 2000, and governed by the HomePlug Powerline Alliance which is composed of companies with background in communications, chip design, and consumer goods. The first rendition, Home Plug 1.0 had a data rate of 14 Mbps. Today, there are power line adapter modems for internet connections that have data rates up to 500 Mbps. HomePlug utilizes orthogonal frequency-division multiplexing (OFDM) [14].

3.2.1.3.4.1

3.2.1.3.5 DC PLC

All the previously discussed implementations of power line communications was the more traditional method that involves communications over the AC power lines. Power line communication is not limited to AC line and it has also been used in DC power systems in applications such as automobiles and cars. Main advantage with the using PLC for DC circuits is to reduce expensive wiring and wiring harnesses which adds a large labor and material cost. With fewer wires in a vehicle, it saves weight which in turn improves efficiency [13].

3.2.1.3.6 Difficulties

In communication systems, a best practice for superior performance is to impedance match all items on the communication line. The impedance matching is to avoid ringing and reflections. However since the network in a PLC system can theoretically be miles long and have hundreds of branches and devices, it is almost impossible to have every device on the network match impedance. The power grid is a typically a very noisy environment that is not conducive to signals.

3.2.1.4 I2C

I^2C stands for Inter-Integrated Circuit. This is both an electrical specification and a communication specification for digital electronics. The I^2C bus allows for devices to talk to each other with only two wires. One wire is the SDA which is the data line. The other wire is the SCL which is the clock line. Since the communication standard contains a clock line this is a synchronous interface which means multiple frequencies of communication can be achieved. What is special about the I^2C bus is that multiple devices can be put on the same bus by still only using two wires as long as each device on the line has a different address hardcoded to it. This is the reason why I^2C is being used on the glove sensor. The Accelerometer, Gyroscope, and Magnetometer used in this project have the ability to communicate through I^2C. This allows for all three devices to be talked by only using two pins of the ATMEGA328 microcontroller.

The I^2C communication standard is a half-duplex communication. This means that only one device in the communication line can be talking at any given moment. Otherwise this will cause messages to clash. I^2C is also a latching communication. This means that the devices in reality don't send a high voltage and a low voltage. Instead the bus line has to be tied high with a pull-up resistor to VCC. This means when the device is talking it either latches the line on to ground in order to send a zero or it lets go of the line and the pull-up resistor will latch it on to a one. This allows the devices to not ever be able to create a short circuit from VCC to ground. This is a great safety feature. The value of the pull-up resistor is usually set between 1k ohm and 10k ohm in order to give a strong pull without drawing too much current. The value of VCC can be between 3.3V and 5V. In our project this will be 5V due to the VCC of the ATMEGA328 being 5V.

The I^2C bus requires each device to have either a 7-bit address or a 10-bit address. For this project we will only be using the 7-bit address feature of I^2C. The I^2C communication also requires a master and a slave. There can be multiple masters and multiple slaves but at least one of each is needed to engage in communication. The master always initiates the communication. It initiates communication by pulling the data line low while the clock line is still high. This is a unique situation since the data line will change when the clock line is high only when a transaction has started or ended.

Once the start condition is given the master will drive the clock line and data line for 8 bits. The data line will only change when the clock line is low. The data line is sampled by the slave on the rising edge of the clock line. The first seven bits the master will send are the 7 address bits. This will be followed by the read or write bit. If the read or write bit is a 1 then the master is asking to read from the slave. If the read or write bit is low then the master is asking to write to the slave.

The next bit is the acknowledge bit. This bit is controlled by the slave instead of the master. So the master's bus will go into high impedance giving the slave control of the bus line for 1 bit. If the acknowledge bit is a zero then the master will know that the slave has received the message. This means that the master will continue its transaction. If the acknowledge bit is a 1 then the slave did not receive the message and it is up to the master to decide how to continue communication whether it's sending a stop condition or trying to send the message again. Some slave devices have the capability of holding the clock line low during the acknowledge bit until it gets the necessary data ready and then let go of the clock line. This is not a common feature among I^2C devices. Our sensors do not hold this capability.

After the master has sent the address and the slave has acknowledged the message the master will send a data message with 8 bits. This is a request for a specific type of data if it's a read command, or the data to write if it's a write command. Once again the slave will have to acknowledge the message at the

9th bit. Whichever device is listening whether it's the master of the slave will always have to acknowledge on the 9th bit.

3.2.1.5 SPI

SPI stands for Serial Peripheral Interface. This is an electrical standard and part of a communication standard for digital devices to communicate with each other. It is a synchronous serial interface and a master-slave interface. So the master has a clock line that goes to all slaves and only the master can control it. All the data will be synchronous to that clock. Along with the clock line there are two data line. MISO and MOSI which stand for master in slave out and master out slave in. So the master controls the MOSI line and the slave controls the MISO. This allows for a full duplex communication which means both devices can transmit data at the same time. The following is an example hookup of SPI.

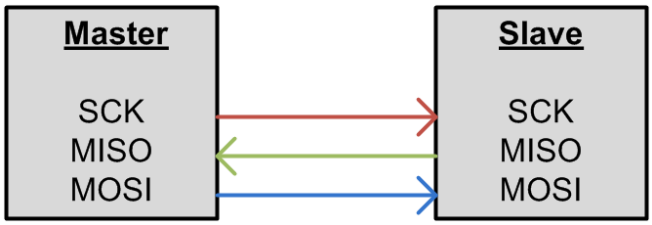


Figure 7: Master/Slave Protocol

Along with this a Chip Select (also known as Slave Select) line can be added. This is useful when having multiple slave devices on the same line. All the clock lines, MISO, and MOSI lines can be tied together and then every single device can have an individual CS line that allows the master to enable and disable the slave devices so that you can talk to multiple devices independently. This is opposed to having a separate clock, MISO, and MOSI line per device. The following is an example of this.

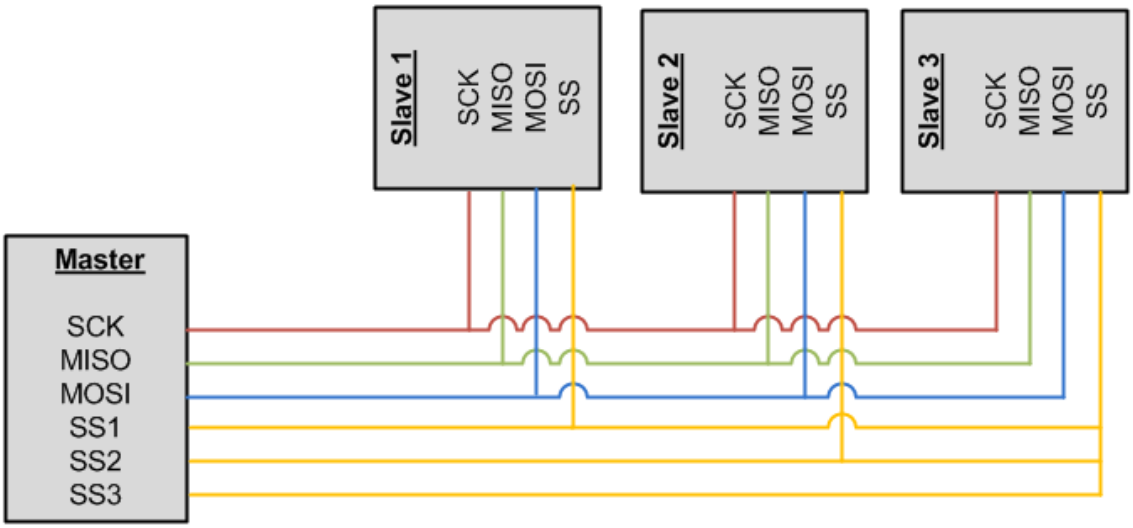


Figure 8: SPI with Chip Select

The communication standard for SPI is actually defined by the user. So in order to speak to a device with SPI you have to know how it wants the data. This can range to picking whether the data is latched on positive edges or negative edges, how many bits of data are sent in one transmission, and more. Therefore to speak to devices that use SPI we need to read their datasheet in order to understand how they want the data. Then we must program the master or slave to adhere to that protocol.

So you can tell that SPI is a very simple interface which is why it is so common to be used. It doesn't require much extra hardware or special hardware to implement this interface. Along with that it is synchronous which allows for high speed transmission with good signal integrity as opposed to asynchronous transmissions where a device needs to be able to over sample a signal.

3.2.1.6 HVAC Control

HVAC Control stands for heating, ventilation, and air conditioning. Every standard home has HVAC control which provides wire interfaces to controlling each of these devices. These devices are generally easily controlled with switches. Either the device is on when it is making full contact to its power source or it is off. So this can be done by using a switch. You can use an electro mechanical relay which would draw a descent amount of power. The most power efficient way would be to use MOSFETs since they require close to no current to turn them on and they can handle enough current to turn on the device.

Commonly thermostats work from a 24VAC source. This 24VAC source is usually created from a transformer on the actual device (heater, fan, or air conditioning) that converts 120VAC to 24VAC. Here is a sample schematic for a 2 wire thermostat that allows you to control just heat.

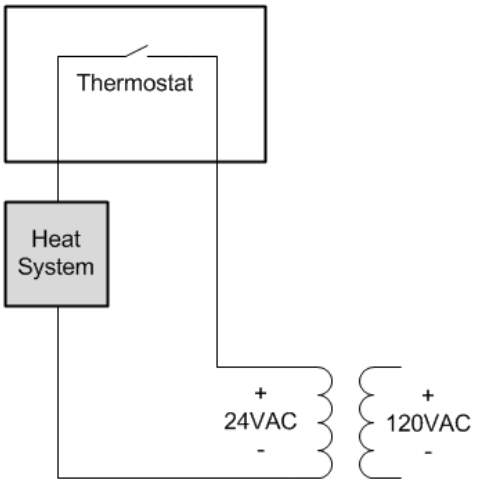


Figure 9: Two Wire Thermostat

The Rh is where the power comes from the transformer and the W wire is what to connect it to in order to turn on the system. A 4-wire interface can be used now to

add cooling into the system. This introduces Rc which is the power for the Cooling line and Y which is the line to turn the device on. Here is its schematic.

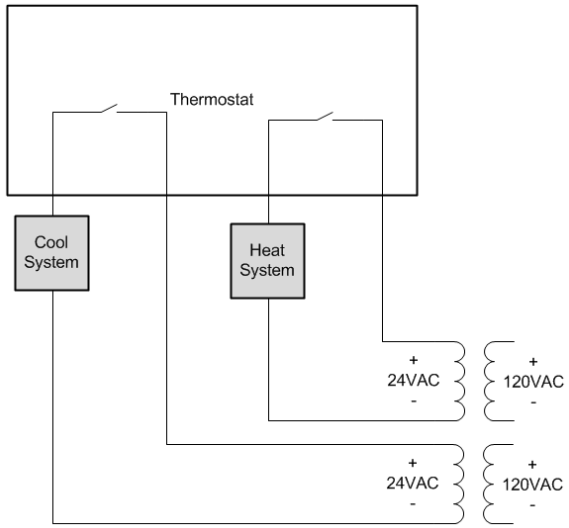


Figure 10: Four Wire Thermostat

It is important to note that in more modern systems the Rc and Rh might be combined into one wire. So you can use this one wire to power all the devices needed. Including adding ventilation which follows the exact same protocol of just using the line to power it on. Usually the fan is labeled G.

3.2.2 AC Mains Control

The standard 15 amp mains electricity circuit or the electricity from the normal three prong wall jack is 120 volts at 60 Hz in America and 240 volts at 50 Hz in Europe. This amount of power is significantly more than what digital circuitry can handle. If it was a lower power source, such as 12VDC or 24VDC, we could simply use a MOSFET transistor to turn it on and off. However the since we desire to have control over AC mains, we have a problem with control because of its large voltage and the nature of alternating current.

3.2.2.1 Electromechanical Relay

A relay is a great option for turning on heavy loads. Relays are electromechanical switches, which means that there is an actually mechanically portion making contact with another portion to create a closed circuit. A relay is composed of a frame, an armature, a coil, and contacts [15]. The frame supports the other parts of the assembly. The armature is the part that is moving and is pulled by the electromagnet on one side and on the other side is attached to a spring to return relay to idle state. The contacts are where the electricity flows through. The switch is controlled by an electromagnetic coil, which while energized changes the switches state. A relay, much like a switch comes in two configurations. Normally opened (NO) is when the coil is not energized, the relay does not allow current to flow, when energized, the circuit is complete and current flows. Normally closed (NC) is the opposite, with un-energized coil,

current flows and when energized flow ceases. Relays are found in many consumer electronics and white goods such as automobiles and ovens.

3.2.2.1.1 Pros

Relays are readily available, very common, and relatively cheap. A search on Digikey for relays produces over 30,000 options. Relays can be relatively cheap, starting at around $1.00. Relays are also a very easy concept to grasp and design with, put electricity in, switch is activated. Electromechanical relays can handle massive amounts of currents for example TE Connectivity's 4-1618002-5 can handle up to 750 amps!

3.2.2.1.2 Cons

With the simplicity of relays also leaves shortcomings. The first being that relays are mechanical, which when movement is involved, lifetime becomes an issue. A relay has a finite amount of cycles that it can open and close, which can be in the hundreds of thousands [15]. Since they are mechanical, they are relatively slow ≈10 ms as compared with a MOSFET which is in the nanosecond range. Since this product is supposed to be in the homes of consumers, we have to design to all the concerns than a consumer could have. One nuisance of electromechanical relays is that when they switch the make a "click" sound which can bother some people. To charge the coil, a large amount of current is required, much more that the solid state version. In addition, when the coil is released, as with any inductive load, a back EMF is produced that can be harmful to the digital circuitry on a microcontroller or transistor if not properly shunted by protection diodes. Since electromechanical relays rely on contacts, with large loads, electrical arching can occur at the contacts, which would make them unsuitable for an environment with possible explosive fumes and the arching wears the contacts down quicker, shorting lifespan.

3.2.2.2 Solid State Relay

Solid state relays are the silicon brother to the electromechanical version. Solid states do not have any moving parts, contacts or coils. Solid state relays use optical isolation in the form of an infrared LED and activate a triac with the help of a timing circuit [16]. Solid state relays are superior to mechanical relays because they can be triggered faster, at lower voltages, and with less current. Since there is no moving parts, they are not susceptible to the wear and tear of a mechanical relay which translates to a longer life span. The downside to solid state is the cost. On Digikey, a search for a 1 amp relay resulted in a mechanical version at $2.00 and a solid state version at $12.00. With the cost of the solid state and the disadvantages of a mechanical, we decided to go with no relay.

3.2.2.3 Triac Phase Control

Since we decided not to go with relays, we opted to have our circuits be dimmable as well. To achieve this we looked at the common AC dimmer switch. They use a type of silicon controlled rectifier called a bidirectional triode thyristor, or triac. A triac is a three pole device with a gate and two terminals A1 & A2. Triacs have a turn on voltage to the gate that must be met to allow current to pass through. Typically turn on currents are between 1-5V. Triacs are latching,

that is, once they are above their turn on voltage, they will stay enabled as long as there is sufficient current, called hold current. Since AC mains is 120V RMS or approx. 170VPP, it is at 0V twice during a period or 120 times a second. The triac will turn off once the voltage reaches near 0 and will not turn on again until the gate is subjected to the turn on voltage. With this in mind, a dimming effect can be accomplished by changing the time between the 0V level and the time to turn the triac back on. A very small time would mean near 100% power, a time of 4.167 ms, or a quarter of a period would relate to 50% power. The power is output is adjustable in a near continuous fashion from 0% to 100%. In standard dimmers, this timing is controlled by analog circuitry, however for our design it will be controlled digitally with a microcontroller. The problem with using a triac in this fashion is the sharp transients that arise at turn-on. These sharp rises can make light bulbs and other devices “hum” in an annoying fashion due a constant change in the magnetic field. This can be counteracted with large inductors and capacitors to smooth out the edges.

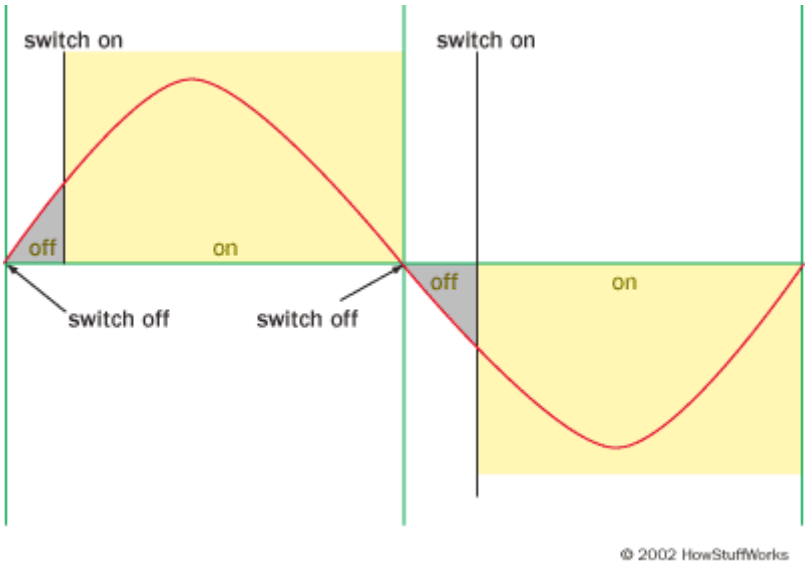


Figure 11: Triac Phase Control

3.2.3 Processor

The tasked designated as project goals and requirements will require significant processing power to be successfully implemented. These processing demands were evaluated, and in addition to the desired user interface, it was determined that our processing power would exceed the capabilities of most microcontrollers. Therefore, the group began research to evaluate the amount and type of processing power that was needed, and what type of hardware would meet these requirements. A brief discussion of the processing power options and each approaches positive and negative aspects is presented below.

Desktop Server: This approach would require the user/home owner to have a computer dedicated to running the home automation. This could be accomplished through networking to an existing home computer, but that would require that computer to be constantly running, which is a prospect that not many

consumers could maintain. The other option would be to sell a small computer that the home owner would leave running, which would be responsible for interpreting data from the sensors throughout the house. One positive of this aspect are that there would be an abundance of processing power, and the system would be safe and private because the data would not leave the house. The downside is primarily the power consumption of running a desktop computer all day, every day of the year. The computer would also be more expensive, which would make the system less accessible to the general public. The final downside would be the size, because the home owner would need to find a place that this computer could run [17].

Web Server: This approach would take the processing out of the house entirely. The main controller would simply gather data from the various sensors and send it to a web server that the home automation company owns. One positive for this approach includes reduced hardware complexity, and reduced power consumption for the consumer. The main controller would be smaller as well, but taking the processing power out of it would likely result in a lower quality user interface. The system would also be slower due to the delay between the house and its server. This would make certain features such as voice recognition less feasible. Another positive for the manufacturer is the potential of selling a subscription based home automation to cover the long term server costs. This would result in a massively profitable business model, since a user's expensive home automation system would become useless if they don't continue to pay the home automation company. However, having a networked house with sensors distributed everywhere presents huge security concerns, since potential burglars could hack entire neighborhoods to better choose house to break into. While security concerns can be addressed, they can rarely be entirely eliminated.

On-board processor: This approach uses a processor built into the home automation controller to handle the data processing and user interface. This would increase the hardware complexity and the power consumption in order to give a friendlier user interface and more predictive power of user behavioral habits. The positives of this approach is that the system can be made much smaller and more efficient than using a desktop server, while still making the system self-contained to eliminate the security concerns of using a web server. The challenges involved in this approach are also more significant, because in the other two examples the group would not have to build the processing board, but in this approach the group will be responsible for creating a board with processor and required peripherals. Ultimately, this approach was the one the group decided to continue to research.

3.2.3.1 Processor Architecture

Having considered the design constraints of size and power consumption, along with the system requirements of a graphic interface and the ability to perform algorithms on large amounts of data, the group evaluated several different types of processors that could accomplish this goal.

Microcontrollers were deemed to not have enough processing power to handle a nice graphic interface. These will likely find their way into other aspects of the project, but will not necessarily be performing the data analysis and processing as the main processor.

Intel processors were found to be too complicated to work with. In addition, the CISC architecture didn't adequately address the power consumption needs of our project.

ARM processors provided numerous benefits. First of all, they are still capable of running an operating system, and that additional layer of abstraction allows our programming to be done in more powerful environments, and will likely improve the overall quality of our user experience. Secondly, they are available in every variant from microcontroller to desktop or server processor. This allows the group to have more options in picking the exact processor. Given the relative simplicity (compared to other powerful architectures), the group decided to further pursue research into ARM processors.

3.2.3.1.1 ARM Processors

ARM offers three variants of the ARM architecture, being ARM Cortex-M, ARM Cortex-R, and ARM Cortex-A. A brief comparison of these architectures is given below. [18]

ARM Cortex-A: The A in ARM Cortex-A stands for application. This architecture is optimized for running operating systems, and therefore are capable of executing high level programs. These processors are well suited for mobile computing devices, and have many options in terms of selecting features that help with mobile computing devices. Some of the features that preliminary research turned up included memory controllers, touch screen interfaces, and a full architecture that is well suited to developing applications for. This is favorable for our project due to the large complexity of code required, which makes high level programming highly preferred over assembly programming. The architecture supports a more full featured instruction set than the other ARM cortex architectures. There are various versions of the Cortex-A processors, ranging from powerful processors used in database management and supercomputing to simpler models used in small portable electronics [18].

ARM Cortex-M: The ARM Cortex-M architecture is optimized for use in small microcontrollers and microprocessors. The M stands for microcontroller. This architecture is best used for handling mixed signals, input/output and motor controls. Processors using this architecture are generally small in size, power efficient, and rich in input/output options, as well as timers, PWM and analog interfaces that allow it to be useful for many small applications that don't require powerful processing. This architecture is more flexible than ARM Cortex-R, allowing it to be used for a variety of applications at the expense of error-proofing [18].

ARM Cortex-R: The ARM Cortex-R architecture is best used for applications that require high degrees of repeatability and reliability. R stands for real time. This is

primarily used for industrial control application, and for industries without any type of user interface, such as automotive industries. These chips are optimized for performance, reliability, and error proofing. However, they aren't well suited for user interfaces, graphics, or complicated input/output, which is something our main controller would need. A common application of the R series processors that is assisting a series processors with certain dedicated tasks. For example, R series processors are used in mobile handsets to assist with 3G, 4G and GPS processing to lighten the load on the application processor. This is worth considering if we find our primary processor struggling to do calculations and handle the user interface simultaneously [18].

3.2.3.2 Memory

Given that this project requires significant data storage and processing, memory for the processor was a primary concern. The main processing board will require two types of memory, being Random Access Memory (RAM) and storage memory. A discussion of the two types of memory required is given below.

RAM is important to the function of the processor since the access times for the memory locations in RAM are significantly less than the access times for mass storage. While still much slower than access times for registers and cache, the processor is able to get data from the RAM quickly enough for fluid application use. This is important to the functionality of the project because the user interface should not be slow to respond. RAM stands for Random Access Memory, meaning that the user is able access any memory location in the same amount of time, when both reading or writing to memory. There are two primary types of Random Access Memory, static random access memory (SRAM) and dynamic random access memory (DRAM) [17].

RAM is fundamentally different than mass storage, in the sense that it is closer to the processor in the memory hierarchy. This means that it has faster access times than mass storage. The RAM is where the application code for the user interface and the algorithm code for the data processing would be stored during the operation of the main controller.

As discussed above, there are two primary types of RAM, static and dynamic. Static RAM has the ability to retain code even when the power to the system is removed. Static RAM, or SRAM, stores data in a 6 transistor cell. SRAM is generally both faster and less power intensive than DRAM, and as such is generally used in speed sensitive application such as CPU registers and cache. However, it takes more room on the integrated circuit, meaning that its memory density is lower than DRAM. For this reason, as well as the increased complexity, it is much more expensive than DRAM, which is the reason that CPUs don't have mass amounts of register and cache memory. In general, most modern computing systems use both SRAM and DRAM, but for different uses. DRAM will lose its value whenever the power is no longer supplied [17]. DRAM technology works by using a large number of small capacitors, which store the data value by being either charged or uncharged. Since capacitors will gradually discharge, and more RAM is implemented by reducing the size of these

capacitors, the data values must be constantly refreshed through the use of a charge pump. This addition of charge corresponds to power consumption. The modern day growth of tertiary mass memory (discussed next) has allowed for the use of dynamic RAM instead of static RAM since the data can be easily stored in a dedicated section of tertiary mass memory whenever the power to DRAM must be removed.

There are various generations of DRAM to be considered. The earliest generation of RAM was SDR DRAM, or single data rate dynamic random access memory. The only thing that distinguishes this type of RAM is that the data is transferred only on one of the rising or falling edges. This type of RAM used a higher storage voltage and lower clock frequency, resulting in a slower type of memory. DDR RAM stands for double data rate RAM, meaning that data is transferred on both the rising and falling clock cycles. The advancement from Single Data Rate to Double Data Rate essentially doubled the RAM memory transfer rate, as the names suggest. DDR1 was the first generation of DDR memory, but became obsolete when DDR2 and DDR3 were released. The improvements made by DDR2 and DDR3 were lower storage voltages and higher operating frequencies. The reduced storage voltage reduced power consumption, while the increased frequency increased performance at the cost of additional power consumption.

Mass storage is important to our project because the various sensors throughout the house are constantly returning data to the main controller, and the controller needs to retain this information if it is to determine user habits. While RAM is the clear solution for application memory, there are many options for implementing mass storage, including a flash drive, SD cards, external hard drives, disk drives and web servers. These are discussed below.

Flash Drive: Given that the base station will be a Linux environment, one option is to include an extra USB port and use a large flash drive as the database storage. The obvious advantages of this method are the low cost of USB storage drives. This option also allows for larger, mass storage USB hard drives to be use in place of USB drives. In addition to allowing a much higher max data storage, this option is also convenient for debugging the household database, since nearly every consumer would have some type of desktop, laptop or tablet with a USB port that would allow them to look at the raw household data. With the data easily removable, it would be simple for the home owner to have some type of desktop application that would process the data and give feedback on usage habits. However, this benefit is achievable using internet connectivity. The downsides to using USB storage is that the drives are large, creating a challenge in making the device look sleek or mounting flush to the wall. Also, there are security concerns with having an externally removable storage device that is so obviously accessible. The transfer rate is faster than SD and internet transfer rate, but slower than a dedicated disk drive.

Another option for mass storage is the use of a SD card. This option would require the addition of a SD slot to the board. The chosen processor would

ideally have a set of pins pre-configured for communicating with SD cards, but a soft connection using general purpose input and output could also be used at the expense of processor time. The benefits of this option is primarily the storage density of the SD card. SD cards have excellent memory density, with sizes as large as 128 gigabytes available in the microSD package of 11mm by 15mm, although for considerable sums of money [19]. Additional benefits are the security benefits of having the data stored securely within the enclosure of the central controller, yet still being able to remove the data for the benefit of the design group debugging the database. The downsides are primarily the limited data transfer speed, which is faster only than average internet connections, with a minimum transfer rate of 10 megabytes per second [19]. Given that size constraints on the controller board are of a higher importance than transfer rate to the household database, this option proves to be a compelling choice.

Using an entirely external storage option is also an interesting option. This option could take one of two forms. One would be a pure cloud based solution, in which the data from the household wouldn't be stored in the house at all, but rather all data and processing would be done outside the household on corporate servers. The other option would have a separate server somewhere inside the house, which could be a separate machine sold as a part of the home automation system, or it could be an application installed on an existing consumer machine. The data could be transmitted either wirelessly or through Ethernet cables. This option has compelling positives, but also significant downsides from power consumption and security standpoints.

Using an external corporate web server has the benefits of drastically reducing the power consumption and size of the central controller, which essentially becomes a device for taking inputs, doing some compression, and sending that information to a server. The downsides to this are the need for constant internet connection, as well as the security concerns of letting the very personal information about home usage outside the residence. As stated before, if this information were compromised, then the home owner could be put at risk of attack or burglary. The total system power consumption from this approach stands to be reduced on a large scale, since the data centers doing the processing could be more efficient by doing computing on a large scale rather than on many small machines.

Using a household server has the same advantages of a web server in terms of reducing device complexity. It has the additional benefit of added security in the sense that data doesn't need to leave the house. However, the data could still be compromised easier than keeping the data local within the house controller. The disadvantages of this approach are the need for an additional computing machine, and the power consumption added by operating this machine.

The final mass storage option to be discussed is the use of a large disk hard drive or solid state drive. This would be connected via SATA connections, whereas the mass storage hard drives mentioned in the first USB approach would still be connected over USB. SATA is the fastest of the transfer methods

discussed in this section, so if the transfer rate was a huge concern for the functionality of the main controller then this option would be more appealing. One downside is the power consumption of running a disk drive. Another is the difficulty of retrieving specific data from a disk drive, whereas all the other options allow for the retrieval of specific locations in memory. The physical size of hard drives is also less favorable than SD cards or flash storage.

3.2.3.3 Processor Power Supply

Supplying power to the processing chip is critical to the successful operation of the chip. The processing chip requires several different voltages. Providing this power can be addressed in several different ways. One method is to provide the power to the separate board components with discrete, analog components. This method has the advantage that the student could demonstrate strong analog skills, but no other advantages. Using separate pieces leads to more overall board space used, and in general the separate power supply components each have disadvantages.

Linear voltage regulators are favorable for situations in which the voltage drop is small, since the voltage drop is lost as energy in heat. Using linear voltage regulators on a board is not an ideal situation since there are many different voltages required and the design would have to incorporate multiple different linear voltage regulators at different voltages [20].

Switching voltage regulators are better than linear voltage regulators for situations in which the regulated voltage is a large drop beneath the supply voltage. They achieve the regulated voltage by using a duty cycle, switching between the supply voltage and ground voltage to achieve a needed output voltage. While better on power consumption, this type of regulator is not good for PCB processor boards, because the rapid frequency switching creates lots of noise on the board, which is a concern for signal integrity of data signals on the PCB board [20]. This would require the switching voltage regulators to be placed a significant distance from any sensitive data signals, which would lead to a larger overall board area.

A better option for powering the processor and peripherals is to use an integrated circuit specifically designed for powering that processor. Given the importance of getting the power supply correct, the availability of reference designs is also an important factor to be considered. This option also has the benefit of being tried and proven to work, as well as a smaller footprint on the board and better signal integrity characteristics [20].

3.2.3.4 Processor Clock

The processor clock is an essential part of a working processor circuit, however it has applications in any dynamic logic circuit. The importance of the clock signal is that it allows multiple parts of the circuit to be synced together on a common signal, which allows for a more controlled transfer of data throughout the circuit.

The average clock signal consists of a fixed frequency, 50% duty cycle square wave oscillation. The 50 percent duty cycle means that the signal spends 50

percent of the time in the high state and 50 percent of the time in the low state. A square wave means that the signal rapidly switches between two states, high and low, and while the transition can never be instantaneous, the switch is fast enough that the circuit has sufficient time to use the clock signal to accomplish the dynamic logic function. In general, dynamic logic circuits use either a rising edge, falling edge, high level or low level, but as mentioned in the section on DDR RAM, some circuits use multiple edges of the clock signals in the circuit [21].

There are several approaches to clock cycles, including single phase, double phase or 4 phase clocks. Single phase clocks indicates that the clock signal can be carried on a single wire [21]. This is most common in modern processors. Double phase means that there are two clock signals, transmitted on separate wires with non-overlapping phases. A four phase is similar to the two phase except there are four wires and four signals instead of two.

Creating an actual clock circuit consists of only a few discrete components. First there is the crystal, which is generally a piece of quartz that is known to oscillate at a specific frequency when a voltage is applied. There are also several capacitors which help with the stability of the clock signal as well as filtering out noise from the clocking circuit. Without the capacitors, the signal will likely be too distorted for the processor to be able to recognize it [21].

3.2.3.4.1 Types of Clocking Circuits

This section will discuss the different approaches to providing a clock signal. With the understanding that all processor chips will have different specifications about what the incoming clock signal should look like, there are different ways to provide this signal. These methods include crystals, oscillators, resonators and RC circuits.

Crystals create an oscillation by applying a voltage to a lattice structure of atoms. The lattice structure is created in a manner that makes the frequency of oscillation very predictable. Crystals create a sine wave output, but for many processors this is good enough for clocking. Crystals have the benefit of being very cheap, which makes them popular for cost sensitive devices like digital watches [21]. They also have the benefit of being small enough to fit in devices of all sizes. In addition, the rate of oscillation is very predictable and quite stable across changes in temperature, which makes them reliable for high accuracy applications.

A crystal oscillator circuit does not include only a crystal. First of all, the crystal must have a voltage applied to it. In addition, the clocking circuit will generally have two capacitors, attached as shown in the picture below. The purpose of the capacitors is to both stabilize the output signal, as well as to filter out higher frequency noise produced from the crystal lattice.

Figure 12: Crystal Oscillator Circuit

Oscillators are similar to crystals in that the frequency is produced from applying voltage to a crystal lattice, and the resulting resonating frequency is used as the clock signal. However, an oscillator will output a more robust square wave that is used as the clock signal. Oscillators have crystals inside [22], but also have other analog elements to amplify the sine wave resulting from the crystal into a square wave.

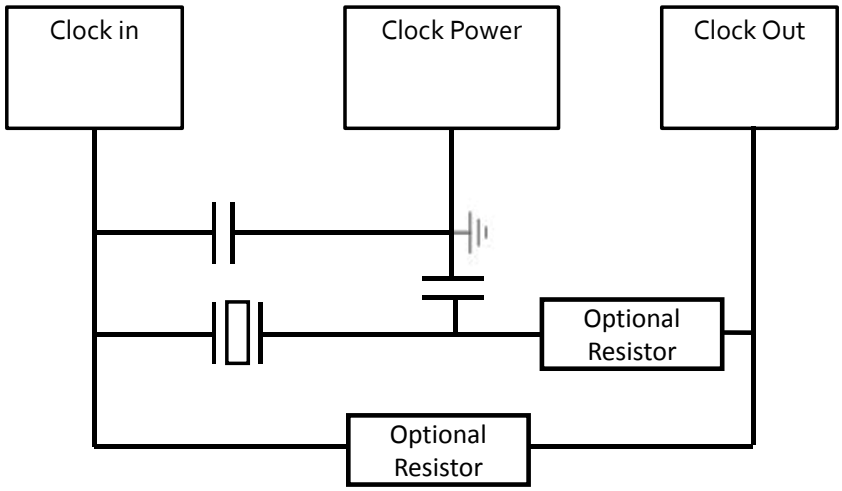


Figure 13: Crystal Oscillator Circuit

Oscillators have the advantage of being easier to use, since the designer doesn't have to choose the capacitor values for the circuit, since they are incorporated into the integrated circuit design. Oscillators also generally have adjustable output voltages, because the input voltage to the IC will determine the upper and lower output voltage. In addition, they are more stable because the capacitors are closer to the crystal that is oscillating. However, the downside to oscillators is their higher cost and the larger IC footprint, which makes them unappealing in low cost or small size devices.

A resonator is very similar to an oscillator, except the material is ceramic instead of crystal. The way in which the oscillation is created is the same in both approaches, but there are different positive and negative aspects. Resonators are cheaper than crystals, and slightly simpler to implement. However, they are more sensitive to temperature, vibration, electromagnetic interference and humidity [21].

Whereas the approaches listed above all rely on the physical structure of the resonating element to create the clock signal, an RC circuit can also be used to create a clock signal. RC circuits, or Resistor-Capacitor circuits, rely on the phase shifting properties of the capacitor to create an oscillation. If properly designed, an RC circuit can provide a stable clock signal, but the nature of the electrical components is that they are much more sensitive to temperature and environmental factors than the mechanical oscillators [21]. However, they do

have the advantage that they will achieve the oscillation frequency in a shorter amount of time than physical oscillators will, which generally require a significant amount of time to reach the steady state of oscillations.

3.2.3.5 Embedded Linux

Linux will be the operating system running on our processor because it is lightweight, easy to customize, and very versatile. Linux is also a free, open source operating system that has distributions specifically for ARM processors. Other operating system would provide us with similar results like VxWorks or WindowCE but they are very expensive and not feasible for this application.

3.2.3.5.1 Distribution Selection

The Linux operating system can come in many different types of packages, known as distributions. These can affect how much memory is required, what sort of graphics the board is capable of sustaining, and what sort of additional features are desired of the OS.

Ubuntu ARM is a very common distribution that can support ARMv7 and up. It targets the Thumb-2 instruction set which is the modern 32-bit ARM instruction set [22]. The uses for the Ubuntu ARM distribution are usually dealing with higher-powered devices that support good video, audio, networking, and decent processing power for running applications. It is based on Debian, it is great for two dimensional user interfaces, and also includes server packages in the installation.

Arch Linux ARM is a more lightweight and efficient option that supports ARMv5, ARMv6, and ARMv7 [23]. For our use of a Linux distribution we would prefer a more lightweight platform for speed and complexity reasons. One downside is that it needs to use outside server software to develop and run a server. In addition, Arch Linux has its own wiki and a lot of documentation available, making the learning curve very small.

Debian ARM is targeted for ARMv7 and normally used for larger and more complex Linux systems [24]. It enables an environment a lot like the normal desktop that can run on low power embedded platforms. Debian also offers a full GNU C++ development environment, many variations of servers and additional libraries/utilities. A large downfall is the small amount of documentation available for installation and customization.

3.2.3.6 JTAG

JTAG stands for Joint Test Action Group. This is not necessarily a communication method, but it is a commonly implemented as a debug port for integrated circuits. Almost all modern ICs that implement internal memory have a JTAG port, and implementing this port allows the user to both read and edit the internal memory registers. These registers are commonly used for settings, so being able to manually set them is a powerful tool. The downside of JTAG is that it is slow, and if there are significant changes to made or large amounts of data are needed, then the user could be stuck with really long transfer times.

3.2.3.7 Peripherals

The main controller will need a wealth of input and output abilities. This section will detail the various different methods of communication that the main controller could rely on to communicate with the different sections of the home controller, as well as how to take input from the user. The input taken could be in either a day-to-day control manner, or from a configuration perspective.

3.2.3.7.1 Serial Ports

Serial data connections transmit data one bit at a time. While there are different protocols, the general idea is that there is one wire for transmitting data from device 1 to device 2, and another wire to transfer data from device 2 to device 1. There may be additional signals for indicating complete transfers, device ready or device busy. Serial communication may seem inferior to parallel communication, but actually benefits from several signal integrity related benefits, including the ability to not worry about crosstalk between parallel wires, and the extra isolation of the wires that is provided due to the reduction of the number of data channels. Some of the popular protocols are RS232, I2C and SPI.

SPI

3.2.3.7.1.1 SPI, or Serial Peripheral Interface is another type of master slave communication method in which one master can communicate with multiple slave systems. It is sometimes referred to as synchronous serial interface or SSI. SPI is a four wire serial interface, in which there is serial clock, created by the master, a wire for the master to communicate to all the slaves, and a wire for all the slaves to communicate back to the master. Each slave also has a select line, which means that if the master must be able to communicate with multiple slaves then it must have multiple select lines leaving the master. The select line is active low, meaning that the line must be brought low in order to select a specific slave chip. The advantages over I2C is that it is simple and has higher throughput, and the disadvantages are that it cannot transfer long distances and it requires more pins than I2C. As the name implies this is commonly used to communicate with

3.2.3.7.1.2 peripheral devices such as screens, sensors and data storage.

I^2C

I2C, commonly pronounced I ‘squared’ C, stands for inter-integrated-circuit. As the name implies, this is used for communicated between various integrated circuits. For example, one might use I2C for communication between the power supply chip and the processor, which would allow them to communicate about error states, power needs and system readiness. I2C is a master-slave communication interface, which means there is one chip that is designated as the master, in most cases the CPU or microcontroller, and the rest of the integrated circuits are designated as slaves. This protocol is very simple in that there are only three wires. One wire is the power wire, and the other two are clock and data wires. I2C is best used in applications where simplicity and cost are more important than the data transfer speed, which is generally lower than other communication methods outlined in this section.

RS232

3.2.3.7.1.3

RS232 is commonly used in computer serial ports. It's generally known as a slower serial communication protocol. This protocol is not necessarily ideal for several reasons, including a large voltage swing, slow transfer speeds, and a generally large physical connector. The protocol specifies a high signal as +3 volts to +15 volts, and a low signal as -3 volts to -15 volts. The large voltage swing would specifically be a problem for a single PCB computer system, in which the swing would create signal integrity issues for other portions of the circuit. Large voltage swings also correspond to larger power consumptions, but do allow for longer transmission ranges. There is also no dedicated, declared method of transferring power along with data (the way that USB can), which makes devices connected by RS232 rely on either an external power source, or limits them to very small power consumption.

UART

3.2.3.7.1.4

UART stands for Universal Asynchronous Receiver Transmitter. This is the corresponding piece of hardware that communicates using some of the communication standards discussed in this section. They are commonly included on microcontrollers and processors. In modern machines, there are often multiple UARTs packaged on the chip, also known as a DUART or dual UART. A USART stands for Universal Serial Synchronous/Asynchronous Receiver/Transmitter, which is capable of doing both clocked and unclocked data transmission.

3.2.3.7.1.5 *RJ45*

3.2.3.7.1.6

RJ 45 is commonly referred to as Ethernet connection, but in fact RJ45 refers to the connector and not the data transfer protocol. RJ45 is an 8 pin/wire connection, in which one wire is power, one wire is ground and one wire is clock. In addition, there are four data wires. The eighth wire indicates command status. This type of connector is commonly used in networking. This connector will be useful in the described application because it is almost certain that a network connection will be required. The ability to communicate with the home owner when they are outside the household would be very useful in a number of situations.

USB

USB stands for universal serial bus, and is a very common communication method for modern computers and peripherals. USB has four pins, including a power pin and a ground pin, and a Data plus and Data minus pin. USB has several different connector standards, as well as several different generations. USB 1.x was the first release, and was not widely adapted until some of the major problems were fixed. It had a max data transfer rate of 12 Megabits/second. USB 2.0 was released later, and had a max data rate of 480 Megabits/second. The most widely used USB standard in modern machines is USB 3.0, which brings the usable data rate up to 4 gigabits/second. All USB standards are backwards compatible. There are multiple different USB port options, including Standard-A, Standard-B USB mini, and USB micro. The advantage of USB is the high standardization of the plugs, but the disadvantage

is that the connection is not industrial grade, and there is usually no locking method for the plug. The data rates are acceptable but in general not exceptionally fast. The plugs can, however, be made quite small, which is an advantage.

Parallel Ports

Parallel data communication is the complementary approach to serial data transfer, in which multiple bits of the same data are transmitted at once. For example, if 32 bit word needs to be transferred using parallel data transfer, then 32 wires would each transmit one bit of the word. Compared to serial data transfer, this is much faster, but also takes up more space and is more complicated. Parallel communication is becoming less widespread in modern computer systems.

3.2.3.7.1.7

3.2.4 Sensors

The use of various sensors is necessary for the successful operation of an intelligent household controller. In the application being discussed, the sensors included on the peripheral devices will be used to retrieve data regarding the environment throughout the house. This data must be accurate and repeatable, and cheap enough to be mass produced. The sensors themselves must fit the packaging requirements and be simple enough to interface with low powered microcontrollers. These various parameters will be discussed as they apply to the various types of sensors to be included on the peripheral sensory devices.

3.2.4.1 Accelerometers

An accelerometer is a device that measures force in perpendicular directions. So if the device is stationary and in the right orientation it can feel the force of gravity on it. If the device is moving it will measure the net force in that direction which can be directly used to find the acceleration through the following equation.

$$Force = Mass * Acceleration \qquad \textit{Equation 1}$$

The device proportionally converts the force into voltages. This can be used to calculate inertial movement or Euler angles of the device. This is extremely useful for a lot of applications especially virtual reality which is what it will be used for in this project. By having the acceleration of a device one can get the velocity and position of the device through integration.

$$velocity = \int acceleration * dt \qquad displacement = \int velocity * dt \qquad \textit{Equation 2}$$

But these are digital devices so we can't do straight integration. We have to sample the data so instead of integration we have to do summation. This unfortunately will lead to a certain amount of error.

$$velocity = \sum_{t=0}^{x} acceleration \qquad displacement = \sum_{t=0}^{x} velocity \qquad \textit{Equation 3}$$

In order to minimize the errors the device should be used with a gyroscope which will be covered in its respective part in the paper.

3.2.4.2 Magnetometer

A magnetometer is used to sense the orientation relative to the magnetic fields of the Earth. The tricky part of using a magnetometer is to make sure your design is not producing a lot of electromagnetic fields that will throw it off. A magnetometer is basically a digital compass. We are using a three axis accelerometer so you get the relative direction north in three dimensional angles. This is really useful if orientation is important. If ultimately all you want is Euler angles then it is not necessary.

3.2.4.3 Gyroscopes

A gyroscope is a device that measures radial acceleration. So it can measure the change in rotation with respect to time. It is important to note that it measures the change in rotation and not the absolute angle of rotation, as opposed to an accelerometer where you could measure the absolute tilt due to gravity. So in order to know the absolute rotation angle we need to know the initial position of the device. This is the reason why accelerometers and combined with gyroscopes. Due to the force of gravity we use the accelerometer to know the current tilt of the device while the gyroscope detects the change in tilt very precisely in the short run.

A downfall to the accelerometer is that it can only measure tilt when the device isn't moving. Otherwise it can only measure the rate of change of perpendicular movement. The gyroscope is only for measurement the tilt angles not the force of perpendicular movement. Regardless this device is absolutely crucial when you are talking about virtual reality. Since the device is always rotating we can't rely on just accelerometers. Once again the accelerometers can measure tilt when the device isn't moving and when it is moving it can measure perpendicular movement. The gyroscope helps get the last piece of information needed which is the tilt angles as the device moves.

3.2.4.4 Capacitive Touch

For our smart switch, we decided to go with capacitive touch instead of a traditional mechanical switch. Capacitive touch offers great benefits to its users instead of typical mechanical switching. First, since capacitive touch is electrically based and there are no moving parts, the life span is much longer than the mechanical counterpart that can wear out. Secondly, capacitive touch can fit in a much smaller package, all that is necessary is the sensor IC and the capacitive touch pad which can be a pad on a PCB. Since it is electrical, the sensitivity can be dynamically adjusted based on conditions or user preference. Capacitive touch switches can be more hygienic, because they are commonly covered by a single piece of plastic or glass which can be easily cleaned instead

of the nooks and crannies associated with mechanical switches where microbes can hide. The draw backs of capacitive touch switching is it requires contact with something that is going to affect the electromagnetic field around the sensor by changing capacitance. So it does not work with all mediums like a mechanical switch does. There also is not the haptic feedback of the “click” of a switch associated with capacitive touch, however there are techniques to add this feedback.

3.2.4.4.1 Capacitive Touch Functionality

Capacitive touch sensing works on the premise that a human body naturally has electrical capacitance. The capacitive touch sensor electrode, usually a pad on a circuit board has a capacitance with respect to a nearby ground plane. When a finger, or other body part enters the electromagnetic field of the electrode, the human body acts as another capacitor in parallel with the capacitance of the electrode. Detecting this change in capacitance is what triggers a switching event. There are two techniques to measure this change in capacitance.

Resistor-Capacitor Method

3.2.4.4.1.1 The resistor capacitor method of capacitive touch sensing includes a simple RC circuit. There is a capacitor with a known value in parallel with the capacitive touch pad and a resistor in series with the capacitor. The capacitor is charged and is then allowed to discharge through the resistor. The capacitor voltage is monitored via a comparator. When the capacitor is fully charged and allowed to discharge a timer is begun. When the capacitor reaches the voltage to trip the comparator, the time is stopped. When there is no extra capacitance on the line, this is considered the baseline measurement. When there is extra capacitance on the line, it affects the RC constant and changes the discharge time indicating that the sensor has been activated [25].

3.2.4.4.1.2

Oscillator Method

Another method for determining capacitive touch is by using an oscillator. A waveform is injected into the sensor electrode and a baseline frequency is established. When extra capacitance is added to the line (touch sensor activated) the frequency of the oscillation changes. The change in the frequency can be easily determine with a comparator and a timer via a microcontroller.

3.2.5 PCB

A PCB, or printed circuit board, is the physical product that results from a schematic design. The printed circuit board will not function without the integrated circuits and discrete components soldered onto it. Printed circuit boards can be very simple, or can become very complex. A simple PCB will have only a few layers, through hold mounted components and the connections will be visible to the user, resulting in an intuitive circuit representation. Through hole components are simpler to solder, making this type of board more accessible to students learning about PCBs and to hobbyists. A complex printed circuit board will have many layers of conductors through which signals or power can be transmitted. The various layers are separated by insulators, and the signals or

power can only be transmitted through layers in Vias, or metal conducting posts that allow signals or power to move from one layer to another. Complex PCBs generally require powerful computer aided drafting (CAD) software to create. A PCB is rarely drawn completely by hand. Rather, an engineer will draw a schematic, with all the symbols linked to files that detail the actual physical size and connector dimensions, and then a powerful software will automatically route the power and signals to realize the circuit. This design would then be checked to ensure that no poor PCB design practices were implemented by the software. The user generally has the ability to specify desired characteristics such as PCB trace width, spacing and turning characteristics.

3.2.5.1 Multi-Layer

In modern electronic PCB cards, it's very uncommon to see single layer printed circuit boards. In contrast, almost all boards are multiple layers. Many integrated circuits, especially ball grid array type integrated circuits, require at least 4 layers. This is due to the density of pins on the bottom of the ball grid array integrated circuits, which makes it difficult to fit the printed circuit board traces out from underneath the integrated circuit unless there are multiple layers with which to route them through [26].

The general structure of a multiple layer printed circuit board is alternating layers of insulators and conductors. In general, entire layers are committed to power, signal or ground. Sometimes, the power and ground signals will share a layer, but avoid each other by leaving sections of insulator between them. Designers must be careful to avoid unnecessary capacitances caused by large sections of conductors separated by thin layers of insulator. This could cause significant signal integrity issues, as well as numerous other circuit problems.

When there are multiple layers of a printed circuit board, it will become necessary to transmit data or power between the layers. To do this, a via is necessary. A via is an interconnection between layers of a printed circuit board. The designer must pay attention to the current requirements of the signal going through the via in order to avoid the possibility of heat damage from an overcurrent condition.

3.2.5.2 Through hole packaging

Through hole packaging indicates that the part needs to have a hole drilled all the way through the PCB in order for it to be soldered onto the board. This method is easier to solder, but much less space efficient because the holes interrupt the signals in every layer of the board. In addition, the same integrated circuits can generally be made much smaller if they are packaged for surface mount instead of through hole. One significant advantage of using through hole packaging is that they offer improved thermal dissipation through the leads of the package. Since there is more metal to metal contact between the pins of the through hole package and the mounting holes drilled in the PCB, there will be more heat dissipated into the board. The fact that the leads extend from the bottom also improves thermal characteristics, since each pin becomes a thermal via to the backside of the board.

3.2.5.3 Surface mount packaging

In general, surface mount packaging is used for finished products, which are optimized for speed, size and power consumptions. Surface mount packaging indicates that the part is held onto the printed circuit board by the solder than connects its electrical contacts to the top layer of the PCB. The soldering techniques needed to mount surface mount parts are more advanced than soldering techniques for mounting through hole parts. Ball grid array is one type of surface mount packaging in particular that is difficult to design for and difficult to mount to boards. Ball grid array integrated circuit packaging has the pins spread out in an array on the underside of the integrated circuit that contacts the top layer of the PCB. Each pin is an enclosed in a small ball of solder, which when heated to the proper temperature in a reflow oven will melt and make electrical contact with the pads on the PCB below.

The challenges involving ball grid array packaging include more than just the difficulty of soldering them on. These challenges include the problem of getting the signals out from underneath the board, because the density of pins is too high to fit all the signals out on one layer of PCB [26]. Therefore, for larger ball grid array integrated circuits it's important to have a PCB with enough layers for the signals to drop underneath and be routed to the appropriate parts. In addition, debugging board problems is much harder when the actual pins cannot be tested with lab equipment, since they are hidden beneath the integrated circuit. An important parameter for ball grid array parts is the pitch width. This tells how much space is left between the balls of solder on the bottom of the part, and must be considered carefully once the minimum trace width and pad size of the PCB manufacturing method is known, as choosing a pitch size too small for the manufacturing process will make soldering the part onto the board impossible [26]. Overall, the benefits of reduced size and increased pin count justify the hassle of designing for a ball grid array integrated circuit.

3.2.5.4 PCB Thermal considerations

An additional challenge of designing printed circuit boards is the thermal dissipation required for various parts. Generally, the center of a surface mount part will be a solid piece of copper that gets soldered down to the ground plane of a board. To properly dissipate the heat generated by some types of ICs, like processors and integrated circuits for providing power, thermal vias should be included. A thermal via is an electrical via, or vertical connection through layers, that is intended to allow heat to travel to the underside of a PCB. The underside of a PCB can be used in the same manner as the top side, which is to say that one can solder parts to it or use spaces of copper to dissipate heat. For a chip running close to its operating parameters, a good approach would be to provide thermal vias to an open space of copper directly beneath the chip, and then solder a heat sink onto the copper pad. A heat sink is a piece of metal with good thermal characteristics and an intentionally large surface area. A properly designed heat sink can allow a device to operate close to its maximum operating parameters [27].

3.2.6 Microcontroller

A microcontroller is a programmable logic device that is used for embedded applications. It is used to control other digital devices by either general purpose input/output pins or peripherals to common communication protocols like UART, SPI, I^2C, etc. A microcontroller contains the processor, the memory, and the programmable peripherals all in the same integrated circuit. This makes it very small and makes it very easy to integrate as well. Some of the simpler microcontrollers only require power, and a clock in order to have it fully working.

When picking out a microcontroller there are a lot of things to consider. One of the most important being what peripherals it comes with. This includes what communication protocol hardware is already built in like UART, I^2C, SPI, and even CAN. There is also things like build in Analog to Digital Converters that vary in accuracy and channels. Some microcontrollers also come with built in op-amps and comparators. Some come with built in RC oscillators. Also you have to watch out for how many timers and what kind it has. Also watch out for the number of interrupts that it can handle.

Other than peripherals you also have to look at the memory. How much memory do you have to store your code, which is usually done in flash. How much memory do you have real time use, also known as RAM. Some microcontrollers allow you to connect and external memory source to its main memory bus in order to expand its memory. Along with this how many general purpose registers do you have to use while writing code.

You also need to look at the processor types. Most microcontrollers today will be RISC processors versus some older microcontrollers which used to CISC processors. Along with this you need to look at how many bits the processor is. This will determine how many instructions come standard to its assembly language. This will also determine how accurate the math applications of it can be.

Speed also needs to be considered. Typical commercial microcontrollers then to be between 1MHz to 100MHz for max clocking speed. But its max clocking speed doesn't necessarily determine the speed your instructions get ran at. That could be determined by the efficiency of the architecture. Like mentioned earlier some microcontrollers come with built in RC oscillators but these aren't the most accurate. A lot of microcontrollers will allow you to add in an external crystal oscillators.

Something very important to consider is how many general purpose input/output lines are available. You need to make sure that the microcontroller you get has enough for your system since every company makes microcontrollers with varying GPIO but with the same architecture. Along with this they will also change how many communication peripherals come with it. Some might come with multiple UARTs and some of them might only come with one due to the limits in pins.

One more topic to consider are the development boards available for that microcontroller. This is important since this is the easiest way to find out if that microcontroller's hardware can handle your system, along with if your code for that microcontroller can handle its algorithms, speed, etc. In addition to this the development board has a lot of accessories that allow for testing of other hardware. For example the Arduino and MSP430 are two very common hobby boards that come with tons of add on "shields" which allow easy proof of concept and proof of hardware for systems.

3.2.7 Motor Control

Motor control is not a main focus of the smart controller, but is necessary to impact the various mechanical systems that control the environment of the home. Motor control could be used to open and close vents, blinds, or fans. Understanding what types of motors are best suited for certain tasks is important, as is understanding the challenges involved in controlling the various types of motors.

Motors are defined as having a rotor and a stator. The rotor is the section that rotates, and the stator is the stationary portion [28].

3.2.7.1 DC Motors

DC motors are the oldest type of motor. They are also the least efficient, but the easiest to control. DC motors generally have fixed magnets that create a magnetic field and a coil of wire in the rotor that creates a secondary magnetic field at an angle to the fixed field. Torque is generated through the interaction between the magnetic field and the fixed permanent pole magnets. The torque causes the rotor to move.

The inefficiencies of DC motors come from two primary sources of energy loss. The first is the friction of the brushes on the slip rings. In order to continue to rotate, there must be a continuous contact between the applied voltage and the rotor, which requires a brush to make contact with a slip ring. This brush is not the most efficient way to transmit energy to the rotor, and the physical friction also causes losses. In addition, the torque generated is a function of the angel between the rotor and the stator, meaning that there will be points during the rotation that have zero applied torque. This makes DC motors ill-suited for point to point movement, but DC motors work well for low to medium speed variable speed operation. DC motors are controlled by applying a voltage to the terminal leads, and the direction can be changed by changing the polarity of the applied voltage. This is often accomplished through an H-Bridge. In addition, PWM is an effective way to control the speed of a DC motor [28].

3.2.7.2 Servo Motors

Servo motor are a type of AC motor. They have the advantage of being more efficient than DC motors, and having good torque at high speeds. They are best suited for controlled torque and speed output applications, but are not suited for high precision point to point movement unless fitted with a rotary encoder and feedback system. Servo motors are structured with windings in the rotor and the

stator, and the interactions between the windings creates a difference in the angle of the magnetic field [28]. The rotor will move to attempt to align the magnetic fields, and the change in magnitude of the AC signal will continue to rotate the magnetic field generated by the stator.

Servo motors are generally controlled by motor controllers, because the feedback loops and control methods are complicated. The additional complexity of controlling AC signals is worthwhile in applications in which the highest motor power efficiency is needed, since servo motors have a higher efficiency rating than DC or stepper motors.

3.2.7.3 Stepper Motors

Stepper motors are DC motors capable of moving in small incremental steps, making them best suited for precision point to point movement. The structure of a stepper motor is to have wide poles with many small magnetic ‘teeth’. The teeth correspond to fixed pole magnets on the rotor, with a slightly offset tooth pitch. The offset tooth pitch ensures that the rotor can only line up with one pole at a time. By alternating the energized pole of the stepper motor, the teeth of the rotor will continue to step forward one increment at a time to best align the permanent pole magnets with the currently energized teeth of the stator. Many stepper motors have hundreds of increments, or steps, per revolution [28].

There are two categories of stepper motors, unipolar and bipolar. Unipolar are less powerful, bigger and less efficient, but draw less energy overall. Bipolar are more powerful for the given energy input and smaller relative to their rated output, but they tend to take more energy overall than unipolar motors [28].

Stepper motors are generally controlled by integrated circuits specifically made for controlling them. These driver ICs will often have an H-bridge for each winding of the stepper motor, and this bridge can be controlled by a PWM signal or an indexer. An indexer allows the user to simply specify the direction and send a voltage pulse, and the motor will increment a step.

The disadvantages of stepper motors are a high holding current and bad characteristics at any medium or fast rotational rate. However, stepper motors are capable of applying large amounts of torque at low speeds or while stationary. In addition, one must be careful to not overload a stepper motor or the motor will miss steps, meaning that the user will no longer be sure of the angular location of the motor [28].

3.2.8 Server/Database Management System

A database management system is the means of storing, modifying, and extracting data from a server. There are multiple levels involved as seen in Figure 14.

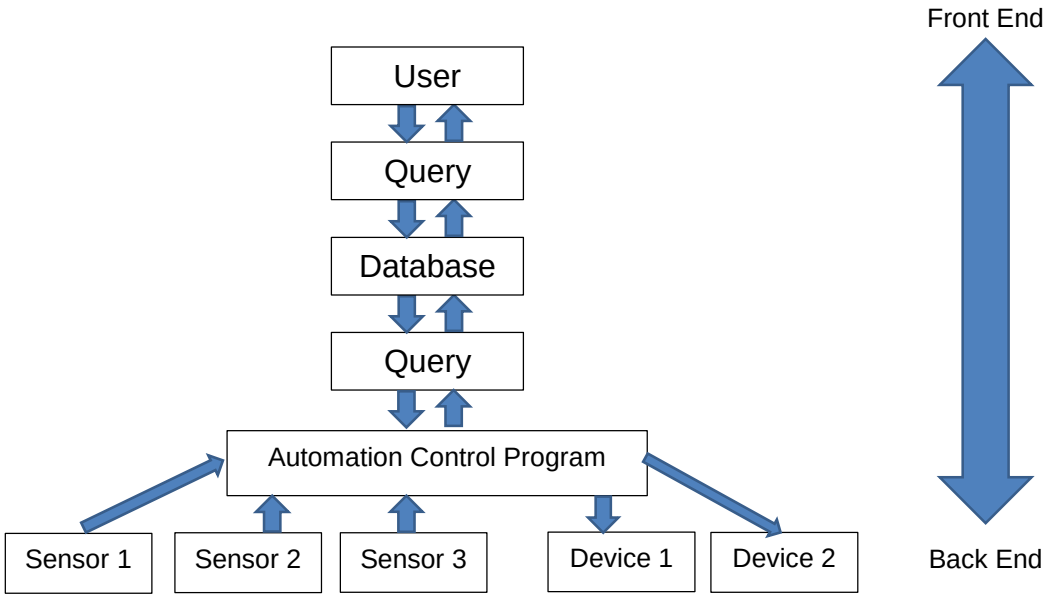


Figure 14

The user application may either be an Android application, Apple application, web application, desktop application, or any combination of those. Refer to the Graphical User Interface section for further research.

The query manager includes both the creation of queries based off of user application inputs along with query execution. For server scripting we are considering using PHP Hypertext Preprocessor or Active Server Pages (ASP). The goal of a server scripting is to be able to dynamically create queries to obtain, edit, or add the necessary information in the database. PHP is a very popular scripting tool [29] that is used in a wide variety of server support systems including in very popular websites such as Facebook. It is very similar to C++ and there is a lot of documentation for PHP which makes the learning curve for this scripting language to be very minimal. On Linux it is very simple to install a PHP package at no cost. ASP is another server scripting tool created by Microsoft that can be programmed in any .NET supported language. It is preferred among .NET developers but due to its price and larger learning curve involved, it isn't ideal for our system.

There are many different types of databases but one of the most utilized database types for pure information and textual data is the relational database. A relational database is based off of the use of tables to represent entities and keys in those tables to help establish relationships between entities. The rows of a given table are the separate records of information while the columns are each attribute for an entity. MySQL is a very common open source relational database software. Development can occur via command line or on a special IDE called MySQL Workbench. MySQL works on many platforms including our target OS, Linux and it is written in C/C++. MySQL is commonly used in an open source

software stack called Linux, Apache, MySQL, and PHP (LAMP) for databases so it works very well with our target OS and our target server scripting language.

There are many choices available for server hosting software. Ubuntu has a specific Ubuntu Server for ARM server system that includes the LAMP software stack so all components would be automatically incorporated together to create the server and DMBS. The goal of this server package is to provide equivalent performance to servers deployed on the x86 architecture. Ubuntu is a very big partner with Linaro which has led to collaboration and the creation of many tool chains, kernel upgrades, and other tools to allow for the most efficient use of an ARM processor as a server. Lighttpd is another open source server hosting option that is known for its speed, security, and flexibility. It also has a very small memory footprint. It is known for being used in very high traffic environment which proves its durability for taking in queries and processing them in a real time environment. Apache is another very common server, used as a part of the LAMP software stack and is known for its abundance of features and robustness [30]. It is known for playing a large part in the initial growth of the World Wide Web. It allows for use on process-based, process and thread based, or event-hybrid mode for different architectures. Because of this high level of customization Apache is able to reduce latency and increase throughput to allow for the handling of more requests in a timely manner.

3.2.9 AI Adaptive Control Algorithms

For the automated residency portion of our system a method for collecting pertinent information, processing that information, making decisions, and commanding external hardware for the system must be designed. Given the correct dataset, the adaptive control algorithms must be able to create commands.

Data Set

The following adaptive control methodologies can be implemented assuming our dataset is complete from the collection of sensor data at a quick enough rate.

Pattern Recognition

The use of pattern recognition in artificial intelligence is usually derived from a multitude of statistical and probabilistic expressions. Pattern recognition is most commonly used for applications such as image processing, speech analysis, machine diagnostics, character recognition, and personal identification.

Clustering

The process of clustering of data points to help to determine patterns is prevalent in many pattern recognition algorithms and can help to classify certain household behaviors at habitual. As a very important part of exploratory data mining and statistical data analysis, clustering allows for a summarization of a data set. Density-based clustering attempts to detect areas with a very dense population and rejects objects in sparse areas. DBSCAN is a very popular density based clustering algorithm that requires the use of two parameters, the minimum

number of points required to compose a dense region and epsilon [31]. Points are evaluated and a search occurs for any neighbors in their epsilon neighborhood. If enough points have close enough neighborhoods then they are determined to be a dense region. In addition, all other points in the scope of their epsilon neighborhoods are also in the dense region. The identification of dense regions can help to display strong user or environmental habits in our system.

Regression Analysis

As a form of predictive analytics and often used for prediction and forecasting, regression analysis is a statistical way to explore the relationship of a dependent variable to an independent variable. One of the main goals of regression analysis is to identify causal relationships. This feature is directly applicable to our system's goal of providing adaptive suggestions to the user based on previous habits and environmental trends. Performing extrapolation can allow us to make assumptions about future behaviors of the automated residency to create action events automatically without user interference.

In order to use this analysis we must be sure that the following assumptions are met:

- The sample is representative of the population
- Error is a random variable with a mean of zero and uncorrelated
- The independent variables are measured with no error
- Linearly independent predictors
- Variance of the error is constant across observations

3.2.10 Human Interface Device Control

A human interface device (HID) takes input from a human, interacts with a processing device, and delivers output to humans. The HID protocol was created to provide a standard means of communication between devices and allow for less need to install device drivers [32]. This protocol was originally created for devices such as mice, keyboards, and joysticks. The benefit to using the HID protocol is that only one HID driver needs to be installed for a computer, not for each HID separately. HID protocol can be used via USB or Bluetooth, with the Bluetooth method being a lightweight wrapper for the USB HID protocol that boasts low latency and low power.

In the protocol there are two parts, the host and the device itself. The host's responsibility is to take input information from the HID and to output certain information to the device. In our system the host is going to be the Central Manager and Smart Switches placed around the house. Alternatively, a computer will also act as host for additional glove functionality like game and computer control. The device is the glove which will define its data packets by using an HID Descriptor. The mode using the HID Descriptor is "report protocol" where the descriptor is a group of bytes that help to define what the content of each packet is going to be, how it will be structured, and how many packets the device supports. After the descriptor in the packet of data comes the data itself to define commanding information for any external device control.

There is a more simplified “boot protocol” that can be used by more hosts but is only able outline the keyboard and mouse information. The keyboard usually has support for all 104 standard US keyboard keys and the mouse has support for the x axis, the y axis, and the first three mouse buttons. This simplified protocol can be used for hosts that cannot parse HID Descriptors.

3.2.11 Graphic User Interfaces

3.2.11.1 Visual Studio

As part of Microsoft’s software suite, Visual Studio is only usable on a Windows operating system but it provides a tremendous amount of tools and libraries. This makes interacting with external devices, setting up ports and sockets, using graphical design components, and dealing with servers very simple. The downside of Visual Studio is its cost and that it only runs on Windows.

3.2.11.2 ADT Plugin for Eclipse IDE

Useful for creating applications for Android devices in Java, this plugin creates an easy means of developing Android applications within your Eclipse IDE. There are specific libraries that can be used to deal with the sensor interface for device accelerometer, touch, and proximity sensors. The ADT Plugin also contains many graphical features that are easy to add into any project such as tabs and other organizational structures, widgets, animation, and more.

3.2.11.3 Qt Project

Provides application and UI framework for developing in C++ or QML. The benefit to using Qt is that it is cross platform, open source, and usable in desktop, embedded and mobile platforms. There is also a very small learning curve involved with development in Qt. Kits can be used to assist with cross platform build and run settings to allow for different back ends to support the same front end of a UI. They continue environmental values for specific devices and tool chains.

Development in Qt achieves almost all of the goals of our project, the most important one being that we need an application that is cross platform. The main GUI will be running on an embedded Linux environment in which none of the previously mentioned developmental strategies can do. We would also like to port this GUI onto Android devices as a remote version of control of the automation system. For design and implementation, using one GUI for multiple platforms provides us with the most time efficient decision.

4 PROJECT HARDWARE AND SOFTWARE DESIGN DETAILS

4.1 INITIAL DESIGN ARCHITECTURE AND RELATED DIAGRAMS

This section will detail the design architecture that was determined as a result of the research completed as a group. The separate sections of the home controller system will each be described in detail as they have been designed, including the manner of communication between the sections of the home controller. The reasoning behind the design choices will be mentioned, but explained in more detail in the subsequent sections. In addition, the specific pieces selected will be explained and justified in subsequent design sections.

4.1.1 Main controller

The main controller was decided to follow the approach of an on board processing unit. This was deemed satisfactory compared to the options of a cloud based processing system, or a separate processing computer elsewhere in the house primarily by balancing the need for power consumption with the security concerns of a cloud based processor. Therefore a central controller would need to be designed that integrated both processing power with moderate storage capabilities, as well as extensive input/output option for communicating with the smart switches, load controllers and the HVAC air flow controllers throughout the house.

To address the need for processing power, a compact mobile system-on-a-chip was selected. This selection was made over desktop or microprocessors in order to balance the need for size and power efficiency of a microcontroller with the processing power of a traditional desktop processor. A module was selected that provided an architecture and instruction set that allowed for a Linux operating system to be installed on it, because the complexity of the tasks and desired user interface quality justified the use of a higher level programming language and operating system.

To address the user interface desired from the main controller, it was decided to include a touch screen interface. This would allow for simple graphical information feedback to the user, as well as simple touch based inputs that can be made on the fly as the homeowner is coming and going throughout the house. Additional user interface options will be presented in the software design section.

To address the extensive input/output points on the main controller, it was decided to use power line communication to communicate with the smart switches, load controllers and the HVAC airflow controllers. This would allow the central controller to simply send data to one PLC communication terminal, which would then send the data to all of the peripheral devices. This both simplifies the data transmission from the main controller, but also allows for the potential of a large number of peripheral devices. This allows the system to be applied to both small single family residences and large commercial buildings.

The design of the main controller was started but not finished, due to the loss of funding from Duke Energy. The design progressed to the point of an integrated power supply circuit, clocking circuits, RAM memory management, I2C communication to speak with the peripherals through power line communication, USB connections, and the beginnings of HDMI and Ethernet connectivity. The estimated PCB board dimensions would be around 25 square inches and 6 layers thick to accommodate the TI AM3358 Sitara processor chosen. This board size, combined with the required integrated circuits, connectors and components, would have cost in excess of $500 to build the first board and $300 for each revision needed. Given the likelihood of a perfectly successful board on the first try was low, and the decreased funding available, it was decided to instead purchase a Beaglebone Black and use the available funding to create better peripherals. The Beaglebone Black met all of the required input and output requirements, as well as the requirement for a processor capable of running the Linux operating system. In addition, it meets the requirements for the needed memory and storage that our designed board would have. The design decisions made prior to the cancellation of this facet will be discussed regardless as they impacted the choice of the Beaglebone Black over other development board options.

4.1.2 Smart Switch

The smart switch was designed to include several input methods from the homeowner, and include several sensors to provide real-time detailed information about the current state of the house. The switch was designed to communicate with the central controller via power line communication. This was chosen because it greatly reduces the installation complexity and cost of this system. The smart switch has been designed to fit the form of an existing outlet, meaning that it can be installed without modifying the home power distribution system.

The user input methods were decided to be a touch based interface and a microphone. The touch interface will simply provide a polished, minimalist approach to controlling the environment of the room. This method of input will allow the homeowner to adjust the temperature and lighting conditions in that specific room. The microphone will not be interpreting speech, due to the difficulty of sending audio data through power line communication to the main controller. Rather, the microphone will listen for a programmed noise such as a clap, snap or whistle.

The sensor inputs were chosen to be a temperature sensor, humidity sensor, ambient light and passive infrared motion sensor. The temperature sensor provides and obvious utility of knowing the environmental temperature in each room, which is the obvious feedback method for HVAC control. The humidity sensor is useful for detecting open windows in a household. Ambient light sensors contribute to the overall energy consumption reduction because any light generated inside the household both consumes electricity and generates heat. By having a method of detecting either the ambient light from open windows, or being generated from auxiliary light sources, the overhead lights on the load

controllers can generate only enough light to meet the user's brightness requirements. The passive infrared sensors allows the system to turn off lights in rooms that have nobody in them.

4.1.3 Load controller

In the intelligently controlled house imagined by the group, the household needed a method to directly impact the energy consumption in a manner other than HVAC controller. The load controller is essentially a digitally controlled dimmer switch installed on either outlets or light fixtures. The load controllers will communicate with the main controller via power line communication.

The load controller will utilize a triac, or triode for alternating current, to control the alternating current of the household power. This will allow for both complete power transmission and zero power transmission, as well as anything in-between.

4.1.4 HVAC airflow controller

In order to most intelligently use the heat or air conditioning being generated in the house, it was determined that an airflow controller was needed. This is to be achieved by using off the shelf motorized vent valve, and adding hardware to communicate and control it with power line communication. The microcontroller will be the TI MSP430G2955 from the MSP430 value line. The motor controller will be the TI DRV8818, a stepper motor controller intended for simple control of small motors. Also included on the airflow controller will be an air flow temperature sensor.

4.1.5 Gesture Interface

In addition to the several interface options described above, it was decided to include an entirely new method of household control. This new method is to be a gesture glove, which will include three types of sensors. These will include an accelerometer to determine user movements, and a gyroscope to determine user hand positioning. The final input will be a flex sensor, which will provide a simple analog input to tell the overall positioning of the fingers relative to the hand. This sensor data will be processed by an onboard microcontroller, and the resulting hand position data will be packaged and sent to a PC. The PC will then complete analysis on the hand data to recognize gestures as commands. The PC will then pass the information onto the home controller via the same power line communication protocol used for the smart switch and the load controller.
Hardware Design

4.1.6 Glove Hardware

4.1.6.1 Microcontroller (ATMEGA328)

The ATMEGA328 is a very popular microcontroller made by Atmel. It is known mostly for being used in the Arduino devices which are a popular hobby electronics board. This makes it perfect for prototyping since we get a board that already has a lot of useable hardware, easy to program, and a lot of reusable libraries.

The ATMEGA328 is an 8-bit RISC CPU. It has 32kB of ISP flash memory which is the memory to hold in programs, 1kB EEPROM for use as RAM, 23 general purpose input/output lines, 32 general purpose registers, two SPI peripherals, one I^2C peripheral, programmable UART peripheral, and a 10-bit 6-channel analog to digital converter. [33] It has 6 hardware pulse width modulation pins. It has 3 built in timers. Along with this it has a 5V TTL interface.

The ATMEGA328 will be used as the main controller for the glove. The I^2C protocol is going to be used to talk to the gyroscope, accelerometer, and magnetometer. The UART peripheral will be used to talk to the Bluetooth module, the ADC will be used to get the data from the flex sensors.

The reason to use the ATMEGA328 is its popularity. This team already has experience with it and has some previously written codes and libraries that allow the use of the communication peripherals like I^2C and UART, as well as using the analog to digital converter. Therefore the ATMEGA328 is a great place to start for the glove. Along with this if the Arduino bootloader is loaded on it then writing code becomes a lot simpler thanks to the pre-written functions of Arduino.

4.1.6.2 Accelerometer

The accelerometer being used in this project is the LSM303DLHC. It is a 3 axis accelerometer and a 3 axis magnetometer. [34] The magnetometer will be covered in its respective section on the paper. This accelerometer has selectable scales of ±2g/±4g/±8g/±16g. The accelerometer uses an I^2C interface to communicate with digital devices. It needs power between 2.16V and 3.6V. The following is a table of the pins for the LSM303DLHC.

Pin#	Name	Function
1	Vdd_IO	Power supply for I/O pins
2	SCL	I2C Clock
3	SDA	I2C Data
4	INT2	Interrupt 2
5	INT1	Interrupt 1
6	C1	Reserved Capacitor
7	GND	0V
8	Reserved	Leave Unconnected
9	DRDY	Data Ready
10	Reserved	Connect to ground
11	Reserved	Connect to ground
12	SETP	S/R Capacitor
13	SETC	S/R Capacitor
14	VDD	Power Supply

Table 2: Accelerometer Pins

Here is a picture of the schematic with the components needed to get the device working.

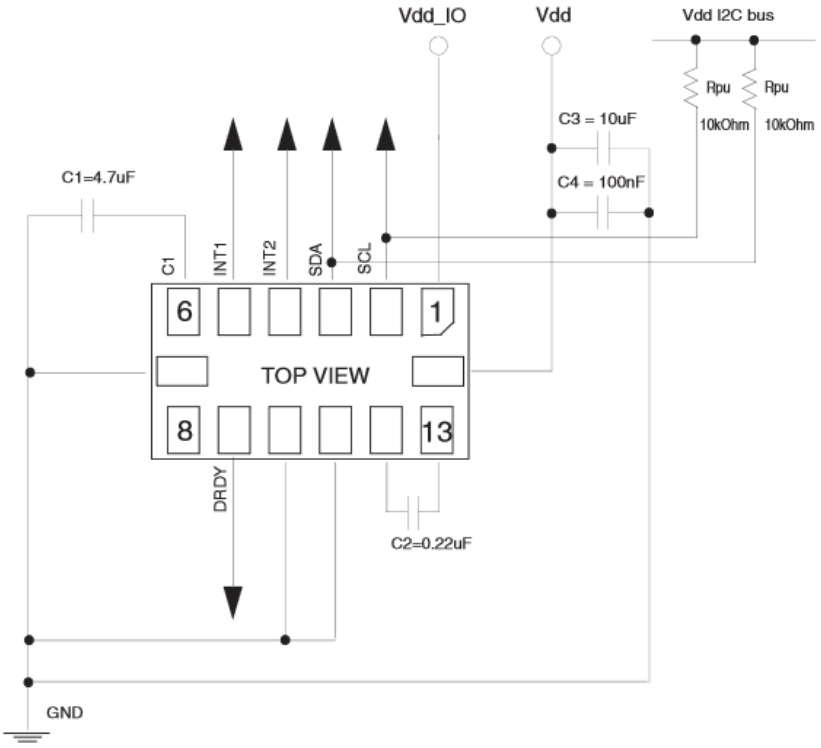


Figure 15: LSM303DLHC Standard Setup from STMicroelectronics (permission requested)

As you can notice the device needs a couple of external capacitors and resistors. First of all are the decoupling capacitors for the power supply. This helps stabilize Vdd so it doesn't have any sudden changes and helps filter out noise. Next are capacitors needed to set the sensitivity of the device. These capacitors should be ceramic capacitors at the values specified in the schematic to have an acceptable sensitivity for our project. These capacitors MUST be present otherwise the device will not work properly. Next are the two pull-up resistors. This is just part of the electrical characteristic of the I^2C protocol.

This device has a built in I^2C address of 0011001b. The device can be configured through changing a lot of the control registers inside of it. But for this project all the registers will be left at their default value therefore we will not go over all the registers. The ones we do need to know though are the output registers which are read-only. These would be six 8-bit registers. 2 registers for each axis. The X-axis, Y-axis, and Z-axis. OUT_X_L_A stands for output, x-axis, low byte, and accelerometer. Here are the addresses in hex.

Name of Register	Address
OUT_X_L_A	28h
OUT_X_H_A	29h
OUT_Y_L_A	30h
OUT_Y_H_A	31h
OUT_Z_L_A	32h

OUT_Z_H_A	33h

Table 3

So to get this data the master needs to send a read command to this device. The master being the ATMEGA328 and the slave being the LSM303DLHC.

4.1.6.3 Gyroscope

The Gyroscope being used in this project is the L3GD20. This is a three axis gyroscope. [35] This device can use either I^2C or SPI in order to communicate. In order to reduce the amount of wires or copper traces we are using the I^2C that way we only need two wires to talk to both the accelerometer and the gyroscope rather than have four wires go to each device. The device outputs radial acceleration with 16 bit precision. Along with this the device also outputs temperature with 8 bit precision. The device can be powered from 2.4V to 3.6V.

The following diagrams are the pin layout, description, and the reference schematic for this device. The capacitors going from VDD to ground are for filtering out noise and stabilizing the DC voltage. The other capacitor coming out of pin 14 is needed for internal purposes.

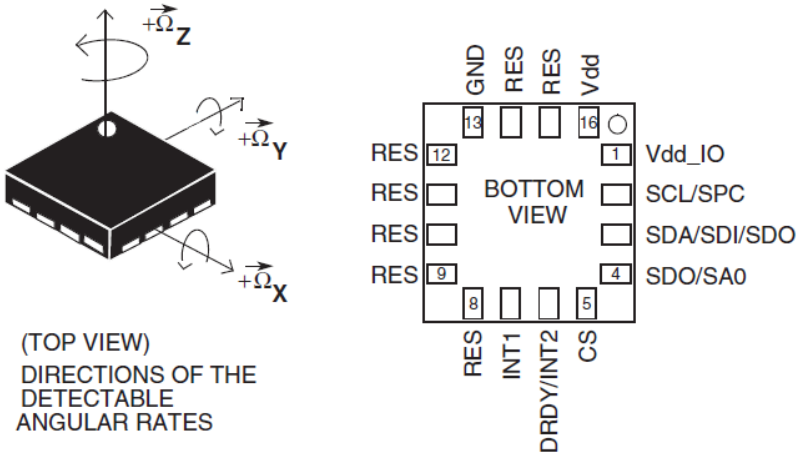


Figure 16: L3GD20 Pin Connections from STMicroelectronics (permission requested)

Pin#	Name	Function
1	Vdd_IO	Power supply for I/O pins
2	SCL	I2C Clock
3	SDA	I2C Data
4	SA0	Interrupt 2
5	CS	SPI Chip Select
6	DRDY/INT2	Data Ready/Interrupt
7	INT1	Programmable Interrupt
8	Reserved	Connect to ground
9	Reserved	Connect to ground
10	Reserved	Connect to ground

11	Reserved	Connect to ground
12	Reserved	Connect to ground
13	GND	0V
14	Reserved	Connect to ground with Ceramic Capacitor
15	Reserved	Connect to VDD
16	Vdd	Power Supply

Table 4

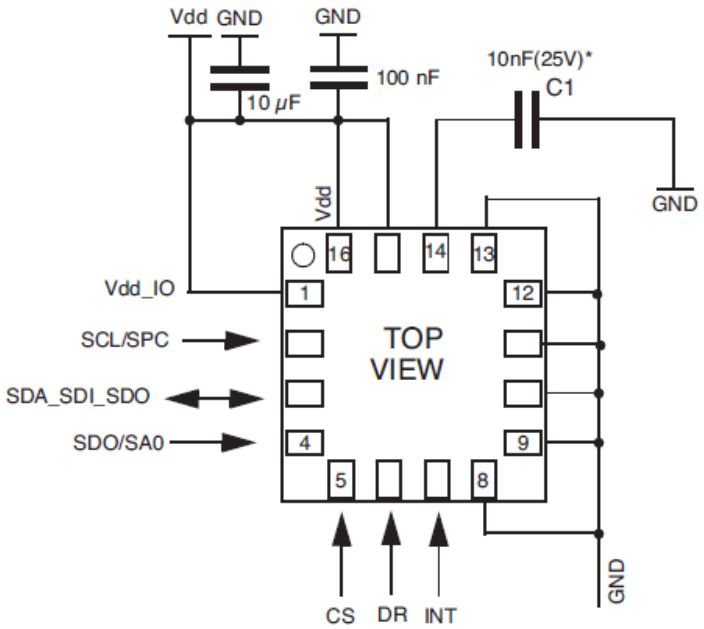


Figure 17: L3GD20 electrical connections and external component values from STMicroelectronics (permission requested)

L3GD20 Standard Setup from STMicroelectronics (permission requested)

The following is a table of the register addresses containing all the information we need for getting the output data. This includes OUT_TEMP for the temperature and all the other output registers for the gyroscope data which is divided into high bytes and low bytes. Since the data is 16 bits and I2C registers only contain 8 bits the data has to be divided.

Name of Register	Address
OUT_X_L	28h
OUT_X_H	29h
OUT_Y_L	30h
OUT_Y_H	31h
OUT_Z_L	32h
OUT_Z_H	33h

Table 5

4.1.6.4 Magnetometer

The LSM303DLHC chip has an accelerometer and a magnetometer. So our design does not need the magnetometer but we are using it there for more data for future features. The pinout of the chip has already been given in the accelerometer part of the paper.

Name of Register	Address
OUT_X_L_A	34h
OUT_X_H_A	35h
OUT_Y_L_A	36h
OUT_Y_H_A	37h
OUT_Z_L_A	38h
OUT_Z_H_A	39h

Table 6

4.1.6.5 Flex Sensors

The flex sensor is a flexible analog resistor that changes resistance relative to the amount the device is bent. In this project the flex will be used to measure the bend angle of the fingers. A flex sensor will be connected to each finger and will span through the length of the finger.

In order to measure the bend angle of each finger the sensor will be used as a voltage divider. On one end of the sensor it will be connected to VCC. VCC in this case will be 5 volts since this is the VCC of the ATMEGA328 microcontroller. The other end of the flex sensor will be tied to a fixed value resistor. The other side of that resistor will be going to ground. This creates a voltage divider. The node where the flex sensor and the fixed resistor value meet will go to an analog pin of the ATMEGA328. Here is the schematic for it.

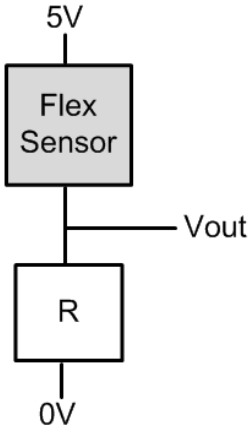


Figure 18 Flex Sensor Schematic

The voltage going into the analog pin will be equal to

$$V_{out} = \frac{VCC * R_{fixed}}{R_{fixed} + R_{sensor}} \quad \textit{Equation 4}$$

In order to pick the value of R_{fixed} we need to know the range of the flex sensor. For the flex sensor in this project the resistance value of the flex sensor ranges from 25k ohms when it is unbent to 150k ohms to when it is fully bent. From testing it was seen that the flex sensor bend resistance was actually 50k which is much lower than the specified range in the datasheet. This is probably due to increasing resistance in lifetime. Which means over time the resistance value of the flex sensor on the glove will probably increase. In order to battle this obstacle the glove will have a calibration feature in the glove's software. This can be referred to in the respective part in the document. From this data R_{fixed} will be chosen to be 50k ohms.

In order to give a more dynamic range from the sensor another resistor can be added in parallel to the flex sensor. This creates a voltage reference for the analog pin of

$$V_{ref} = \frac{VCC * R_{fixed1}}{R_{fixed1} + (R_{sensor} || R_{fixed2})} \quad \textit{Equation 5}$$

This option will be explored more in depth after more testing. The reason to consider this approach is because the flex sensor doesn't start at 0 ohms. If this was the case then you could match R_{fixed} to be the max value of the flex sensor and have a big range for the voltage divider. Since this isn't the case and the initial resistance value of the flex sensor is so high this approach can help increase the range of the voltage reference.

The flex sensor also has a maximum power rating of 1 Watt. [36]Since the flex sensor itself is already such a high value of resistance this is not a concern for this design. Given by the power equation:

$$P = V^2/R$$

The high resistance value results in a very low power consumption. The current going through the sensor will be very low. The ATMEGA238 has an input current limit of 40mA. Given by the worst case scenario of VCC/R_{min}, where R_{min} is the minimum resistance of the sensor which would lead to the highest current possible it can be seen the current will not reach anywhere close to 40mA.

Flex sensors usually come in two standard sizes. 2.2" and 4.5". In order to get the bend angle of the whole finger the 4.5" sensor will be used.

4.1.6.6 Bluetooth

The devices that are going to be used for Bluetooth communication between the Wireless Glove, the API, and the Main Controller is going to be the Bluetooth bee. These are Bluetooth integrated into the XBEE PCB format. [37]Along with

this will be a UartSBee shield by Seeedstudio which is a PCB that lets you put in UART message to control XBEE devices. The combination of this and xbee devices is an extremely easy way to do multiple kinds of wireless communication while only needing UART.

The Bluetooth Bee goes on directly on the UartSBee shield. It is a 4 wire interface. Power, ground, Rx and TX. Power for the BT Bee is 3.3V but can be powered by 5V since the UartSBee shield has a voltage regulator on it. The BT Bee is set to 38400 baudrate by default for UART. It will be left at that in order to not run into compatibility issues between the other BT Bees.

Here is the example block diagram for our system.

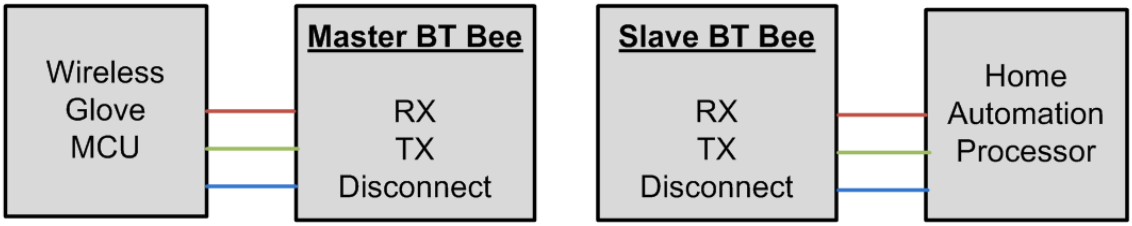


Figure 19 Block Diagram/Schematic

4.1.6.7 LiPo Battery

A Lithium-Ion Polymer battery is a very standard kind of battery used today mainly for the fact that it's rechargeable. The wireless glove will be powered by a LiPo battery for its convenience. It will be on a connector that will be removable from the system in order to put the battery on a standard charger. Putting a charging system on the glove would be hard to do considering the small area that's already used by most of the hardware on the glove.

The things to consider when picking these batteries is the voltage and current in amps per hour. You can add multiple batteries in series to stack voltage or in parallel to get have a larger current capacity. The goal right now is to only use one battery but if after testing the current capacity is not enough we will fall back to this method.

The battery selected is a 063048 cell. It has a nominal voltage of 3.7V and a rating of 900mAh. [38]The 3.7V is great for the ATMEGA328 since it can be powered anywhere from 3.3V to 5V. It will be going through a linear voltage regular for 3.3V. The battery selected also has built in protection for over-voltage, over-current, and minimum-voltage.

4.1.7 Home Automation Hardware

4.1.7.1 Main Controller

The main controller design, as outlined in the introduction to the design section (4.1), includes a processor and its essential external components, RAM, LCD screen with touch interface, USB port, SD storage port, Ethernet interface, UART interface, I2C interface and SPI interface. This section will detail the important

information regarding this specific designs integrations of the various system components listed above. Each section will explain first the design approach taken by the student design group, then how these decisions impacted the decision to instead use a development board, as discussed in the following paragraph.

The initial design description included plans for creating the central controller from the ground up, including plans to embed the processor on a custom PCB containing RAM, a clock signal, mass memory and support for various types of peripherals and data connections. This plan was pursued vigorously throughout the first 2 months of the first semester of design, and was continued until the group was informed that the anticipated funding had decreased significantly. At this point the decision was made to focus the available funding and energy into creating a more complete set of peripheral devices for the central controller to communicate to and use an off the shelf computer device for the central controller. This decision was motivated primarily by the expensive manufacturing costs of the designed board once all factors were taken into consideration. These factors include the cost of manufacturing the board, the cost of high performance integrated circuits in small batch size, the difficulty level and associated risks with soldering these high performance chips to the manufactured board and the likelihood of a redesign being required.

The cost of the manufactured board was going to be high. The ball grid array processor, TI AM3358, required a minimum of 6 layers to successfully route all signals from underneath the device at the available trace width [39]. The keep-out region for the RAM and HDMI processor chips was large enough that when considering the external connections required and the desire to isolate analog from digital traces, the group estimated that a conservative PCB size estimation exceeded 25 square inches (the group never proceeded to PCB design, as the schematic was unfinished at the time the decision to cut was made). The only recommended PCB manufactures that support 6 layer boards are PCB pool and 4PCB, and only PCB pool pricing was available for our requirements. The price for a run of 3 boards exceed $500 [40], and the bill of materials at the time the main controller custom PCB was cancelled approached $100 per board, and would exceed $300 to buy all required parts for three PCBs. This would have been ill advised with the full anticipated budget, but seemed particularly unwise with the reduced budget available. The alternative of a $30 Beaglebone with $60 touchscreen seemed to be a much more prudent allocation of the available funds [41].

For each of the following sections relating to the main controller, the design for the custom PCB will be discussed, then applied to the Beaglebone Black that will be used in the final prototype. An additional design discussion of a required custom Beaglebone cape (expansion board) is included.

4.1.7.1.1 Processor

The processor chosen for the main controller is the Texas instruments AM3358, a Cortex A8 processor from their Sitara line of processors. [42]The process of

selecting this specific processor began by selecting the brand of processor, Texas Instruments. This was chosen due to the wealth of resources regarding Texas Instruments parts, including design tools available on their website and extensive yet understandable datasheets. Considering the group had no prior experience designing processor boards, the availability of reference materials was a major factor in choosing a Texas Instruments chip. Next we had to decide the level of performance needed. The Cortex-A series was selected because it provided the ability to run an operating system on it, which we deemed a major benefit for creating a graphical user interface. Next the level of Cortex-A processor was determined. In this decision, it was deemed that the newest, highest performance chips would be far more graphics power than was needed for the relatively static graphic user interface that was planned. Also, it was determined that choosing a processor that had been used for development boards with similar levels of performance to what was needed for the main controller would allow us to judge the relative performance of the finished main controller board. It was finally decided to choose a Sitara Cortex-A8 because it best matched the graphics performance needed, yet offered a large variety of input and output methods, including an integrated LCD controller and support for a touch interface, both of which were important to our main controller.

The AM3358 is a system on a chip, meaning that it includes nearly all of the necessary components for a processor to perform tasks [39]. It includes both a CPU and a graphics driver, as well as numerous separate processing sections for controlling external interfaces. These sections include an LCD controller, external memory interfaces for both RAM and mass storage, a real time clock system, a network controller, 6 UART communication ports, two audio serial ports, I2C master/slave ports, general purpose input/output pins, debug support and security focused systems. Having all of these tools integrated into one chip makes for a powerful system, but it also greatly increases the complexity of designing for the chip and increases the density of the pins located on the physical system.

The physical integrate circuit package for the AM3358 comes in two varieties, a 298 pin package and 324 pin package [39]. The package best suited for the needs of this project was the 324 pin package, because the ball pitch was larger. This increases the physical size of the integrated circuit package, but due to the limitations of the PCB manufacturing companies within our budget the larger pitch size was required.

Once the processor was picked, the general schematic to power and clock the processor was created. There are essentially three things that a processor needs to perform any type of processing task. It needs power, memory and a clock signal. Each of these essential support systems to a processor is detailed in following subsections.

> *Note: While the group designed board containing this processor will not be manufactured, the processor on the Beaglebone Black is also manufactured by TI and is a member of the same product family as this chip. The two chips could*

be interchanged with only minimal changes made to the external circuitry, and being members of the same product family indicates that all of the desirable features of the AM3358 are available on the Beaglebone development board.

Powering the Processor

The design for powering the processor uses an integrated circuit specifically designed for powering the AM3358 processor [43]. This chip provides the correct voltages with the needed currents available, and also includes an I2C communication channel so that the processor can demand certain power supply conditions. In addition to providing the power needed to operate the CPU, GPU and RAM, it also provides several DCDC converters to provide system DC power needs. These can be configured to various different voltages as needed. It also provides a LED driver, which eliminates the need to create an analog circuit to drive the status LEDs that our board will include.

4.1.7.1.1.1

The specific chip used is the TPS65217C. The C indicates that this chip is optimized for powering DDR3 RAM, which will be discussed in a later section. It can take its power input from either a battery, a DC or AC source. The DC source is generally a USB power supply, as would be found on a cellular device. The processor board will take its power from an off the shelf ACDC converter configured for 5V. The system first processes the input voltage to create a general system 5 volts supply, which is then fed back into the power supply DCDC converters to provide the needed DC voltage levels for the various processor internal components [44].

In general, the center pins of the processor are the hardest to reach on a PCB design, but they also generally include the power and ground pins. Since they all can be connected together, this greatly simplifies the challenge of routing the data from so many pins out from underneath the processor IC to the power management IC.

Note: The Beaglebone Black powers the processor using a similar approach of an integrated circuit specifically designed for that processor. This is a benefit to this design in that it protects from variations in the external power supply to the Beaglebone since all the voltage reaching the processor is coming from a power supply chip. In addition, the chip provides the correct voltage for the type of RAM on the Beaglebone. Providing the required voltages from one chip rather than multiple voltage regulators contributes to a smaller overall size [45].

4.1.7.1.1.2

Clocking the processor

The various clocking circuits for processors were explored in the research section, and from the options presented an oscillator integrated circuit was chosen. Given the proximity to the alternating current of the power line communication, it was determined that the protection from EMI was the most important factor, and the enclosed design of oscillator integrated circuits provided the best protection against frequency instability. In addition, for a circuit that is the backbone of the processing board, it was determined that the extra cost and board space was justified by the simplicity of design.

For the AM3358, two frequency signals need to be provided. These two frequencies are 32.768 kHz and 24 MHz. The MHz range value has multiple options, but the middle option of 24MHz was chosen to provide the biggest margin in case of frequency shift due to EMI from the alternating current or other environmental factors.

> *Note: The Beaglebone Black operates in a very similar manner in that it requires two different frequencies clock input to operate with all features enabled. Whereas we would have used integrated oscillators for the simplicity and stability, the Beaglebone uses crystals to reduce the cost of each board* [45].

DDR3 RAM

The Beaglebone Black comes with DDR3 SDRAM. This RAM is the same generation that was designed on the custom controller board schematic, but a different chip. The chip found on the Beaglebone has a capacity of 4GB, making it by far large enough to run a graphic user interface and compete processing at the same time [46].

4.1.7.1.3

Ethernet

Ethernet is a commonly used networking interface for Local Area Networks, or LAN. It is also used for larger networks, including internet connectivity. Relating specifically to this project, Ethernet connectivity is important for allowing remote control of the house, as well as if the remote processing option is chosen as a viable option. Ethernet has the advantages of being the best way to network a computer system, but if an Ethernet connection is to be used over long distances, a repeater might be needed.

4.1.7.1.4

> *Note: The Beaglebone has an Ethernet switch and RJ45 port, making it ideal for our networking needs* [45]. *This allows the Beaglebone to communicate with the internet and local area networks in the same manner as any other computer. This feature will be utilized to network the main controller with a webserver and mobile application.*

4.1.7.1.1.5

HDMI

The custom main controller was initially planned to include an HDMI interface, and the complexity and expense of the connections and the data buffer was one of the contributing factors to the decision to discontinue work on the custom main controller. The Beaglebone has an HDMI mini port, and this connection will be implemented to send display data to the LCD screen. HDMI is ideal because it conserves the IO pins that would have to be used to send the video data if using another data transfer method.

4.1.7.1.1.6

USB

The custom main controller contains a USB connection in the schematic. It was determined to use a USB-mini port to reduce the size of the board. An important consideration when using USB is the risk of electro-static discharge, which was addressed through the use of a specialized USB ESD protection “groundstrap” IC, which protects the processor from the discharge when the shield of the cable first touches the shield of the port [47].

Another important consideration when designing a microcontroller with USB connection is to protect analog signals from the noisy high speed electromagnetic interference that USB data connections create [48]. This can be addressed by isolating the PCB traces from other data wires, and to use care to not cross the USB high speed data signals over breaks in the ground/power planes, because the high speed digital data will create interference on the ground plane, which will in turn affect everything in the ground signal return path.

The benefit of USB is two-fold, since it allows the group to debug the code in real time, and also is the simplest method of returning touchscreen data from the selected screen module to the Beaglebone. The Beaglebone has a single standard USB 2.0 port, but this can be split using a traditional USB splitter during the prototyping stage.

4.1.7.1.2 LCD

The LCD screen to be used during both programming and on the final design is the SainSmart 7" LCD touchscreen. This screen is ideal for several reasons. First of all, the LCD screen is a complete package. LCD screen design was not a topic that the group was planning on researching and designing, so the decision was made to purchase a complete solution for both the LCD driver and touchscreen interface. This screen interfaces well with the Beaglebone since the video input for the screen can be provided through HDMI, which is included on the Beaglebone. The touch data is processed on the board's driver, and this data is returned to the processor through a USB connection, which is also supported by the Beaglebone. The decision to purchase and use this screen was also motivated by the necessity to begin programming as soon as possible due to the amount of coding that needs to be completed for this project to be successful. By paring the SainSmart display with a Beaglebone, the software development was able to begin before the hardware was completed for less than $100.

The size of 7" was also carefully considered. This size was chosen over smaller or larger sizes primary due to the type of user interface the design group imagined. A screen large enough to be comfortably read at arm's length was desired in order to facilitate fast user input as the home owner was entering or leaving the house. However a larger screen would be unnecessary and would require an even more significant installation. The power draw of a screen larger than needed also influenced the choice of 7". Any smaller than 7" would require the user interface to not convey enough information quickly and could be easily ignored if there is important information the homeowner needs to see when near the device.

4.1.7.1.3 Estimated Load

The power conversion includes an exterior high current 120VAC-24VAC transformer, not mounted on the board. This transformer must be able to provide power for the Beaglebone, the LCD screen module, the PLC amplifiers and the Bluetooth Bee. The Beaglebone draws a maximum of 500mA at 5V, for 2.5W [41]. The LCD screen was measured to draw a maximum 900mA at 9V for 8.1W. The PLC amplifiers draw up to 2A at 15V for 30W at a small duty cycle [49]. The

Bluetooth Bee power consumption is minimal at 3V and 100mA for .3W. The total is 41W.

4.1.7.1.4 Transformer

A safe margin of error would suggest a transformer capable of supplying at least 60W. Transformers are rated by VA, a unit of power considering both real and reactive power, so to ensure the transformer can supply 60W of real current a higher VA rating should be selected.

The first component required is a transformer to bring the voltage of the household main to a more manageable level. 24 volts was chosen for this design because 120V to 24V transformers are readily available and cheap. The resulting 24V DC will need to be further regulated to lower DC voltages.

The selected 120V to 24V AC transformer is TCT50-03E07AE by Triad magnetics. This transformer is capable of handling 6A, more than enough for the application needs [50]. This particular part is large due to the power ratings, and is therefore a chassis mount part. This dictates that it will need to be mounted to the exterior of the main controller rather than connected to the PCB. In a practical household application, this part will be fixed to existing household support studs near where the main controller is installed, and the 24V power output would be fed into the rest of the power supply.

4.1.7.1.5 Rectification

Once the household main voltage has been reduced to the desired level, the alternating current is transformed to direct current using a rectifier. A full bridge rectifier is desirable over half bridge rectifier. The resulting voltage is not suitable for DC use in most cases, as there will be significant voltage ripple from where the AC voltage approaches zero.

The bridge rectifier selected is the MCC GBU6A, a module capable of rectifying 24V at 6A with at 1.1V forward diode voltage drop [44]. A large 470uF filter cap is placed after the rectifier. This component will be mounted on the custom Beaglebone Cape, and will take 24V AC input from the previously mentioned transformer.

4.1.7.1.6 Analog Filtering

Analog filtering will be completed by a pair of large electrolytic capacitors. Electrolytic is preferred due to the larger capacitances available. The purpose of the capacitors is to reduce the ripple to a more acceptable level. One capacitor will be placed between the output voltage of the bridge rectifier and ground prior to the DC voltage regulator, and the other will be placed after the DC voltage regulator. The purpose of the second capacitor after the voltage regulator is to supply instantaneous power to the circuit during periods of rapid power consumption rise, since voltage regulators aren’t well suited for supplying more power instantaneously.

4.1.7.1.7 Voltage Regulators

The unregulated voltage is fed into a switching voltage regulator, the ACT4533 by Active Semi. The large drop from 22V to 15V dictated the use of a switching voltage regulator. The switching frequency is 125 kHz, which is fast enough to give excellent efficiency of 91% at our usage parameters. However, the large frequency requires the use of a large current filter inductor on the output, with an inductance of 220uH. The selected inductor, the RL2446 (recommended by TI in datasheet for given supply parameters, has a diameter of almost an inch, meaning that much of the cape will be taken up by this inductor. This 15V output is then further regulated by two linear voltage regulators to 9V and 5V for the screen and microcontroller. The linear was deemed acceptable for the 6V drop to the screen given the lower power needs, and the smaller current being supplied to the Beaglebone at 5V doesn't justify the board space required for another switching voltage regulator.

A voltage divider sets the reference voltage to determine the output voltage as compared to an internal reference voltage of .808V. The low side resistor is determined to be 10kΩ to limit current flow, and the high side resistor is found to be 175kΩ. Other external components include a bootstrap capacitor with a datasheet provided value of 1uF, and output surge supply capacitors of 470uF discussed previously. A schottky diode is also recommended by the datasheet to protect against high initial voltage spikes common in switching voltage regulators. It is rated to a reverse breakdown voltage of 40V, since this is the upper supply voltage of the linear voltage regulators following the switching regulator.

4.1.7.1.8 HVAC interface

Interfacing with the heating ventilation air condition system is an important part of creating a home automation system, but unfortunately there is no single solution that allows the main controller to interface with all HVAC systems. Rather than try to design a system that can interface with all of the possible HVAC systems, the decision was made to instead interface using one of the most common household industrial standards.

This standard is very simple. The heater/air conditioner supplies a 24V signal to the thermostat, or main controller within the scope of this project. The heater/air conditioner also has a 24 logical input, and when the 24V signal is passed back to the input on the heater/air conditioner, the selected feature is enabled (either heater or air conditioner). This communication method will be implemented by having a screw terminal on the Beaglebone Cape, and two high voltage MOSFETs. The gates of the MOSFETs will be connected to general purpose IO pins on the Beaglebone, which can then control the continuity of the path back to the HVAC digital input.

4.1.7.1.9 Schematics

A screenshot of the main controller schematic as designed before the funding cut is shown below. To better organize the drawing, it was done on 3 separate pages. Each page is shown with a brief description of what is shown. For clarity, this design was not finished and will not be manufactured, but significant project

knowledge was gained from the time invested and will therefore be discussed, including how this knowledge contributed to the selection for the Beaglebone Black.

Mainboard Schematic Page 1

Mainboard Schematic Page 2

Mainboard Schematic Page 3

Beaglebone Cape

4.1.7.1.9.1

After the decision was made to use the Beaglebone Black Dev board instead of a custom board, it was determined that there would still need to be significant analog circuitry and connection hardware included to power and interface with the Beaglebone. The options for creating this interface were to create a separate board with power supply and connections for Bluetooth and PLC communication, or to create a stacked PCB to connect directly to the Beaglebone. Different development platforms call stacked add-on boards different things: In the Arduino development platform they are called shields, the MSP development platform refers to them as booster packs, and the Beaglebone development platform refers to them as Capes. Therefore, a Beaglebone Cape schematic was created to power the Beaglebone off of the 120VAC power supply, and provide connection points for the Bluetooth interface to the glove (through Bluetooth Bee by ZigBee), and the Custom PLC communication card designed as part of the smart switch.

The other purpose of the custom cape is to provide good mounting points for the Power Line communication interface and a Bluetooth communication module. The interface pin patters are shown below.

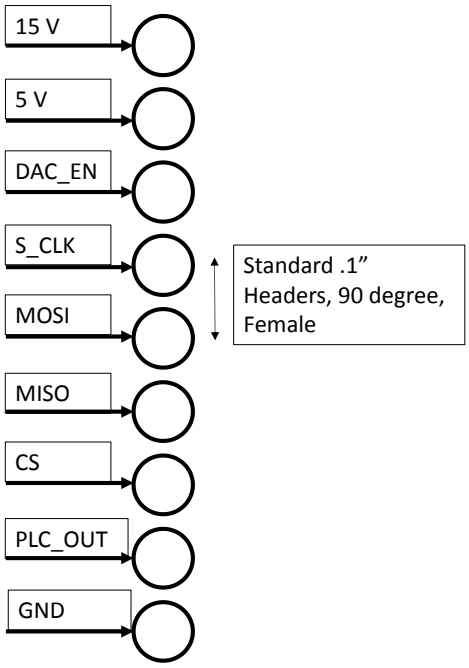


Figure 20: PLC Communication Pin Connection

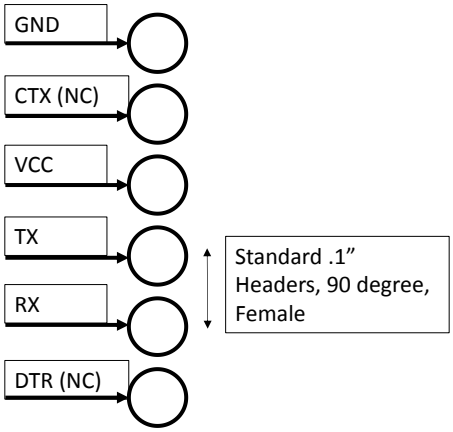


Figure 21: Bluetooth Bee Communication Pin Connection

The initial schematic for the Beaglebone Cape with power supply is shown on the next two pages. Missing functionality of the Cape at the stage shown include the ability to interface with the household 24V logic generally used to control the HVAC system, which will simply be a set of high voltage, logic level MOSFETs connected to GPIO pins of the Beaglebone.

Cape Schematic Page 1

Cape Schematic Page 2

4.1.7.1.10 Power Line Communication

Analog Front End

We are not analog experts or claim to be, with this in mind we decided to go with a commercial solution to the analog side of power line communication. To achieve this we came across a Texas Instruments product, the AFE031. It is an analog front end that handles most of the signal conditioning and amplification. At the heart is a large power amp designed to drive signals in the low impedance environment of the mains electrical grid.

The AFE031 power line communication chip can be communicated with in multiple ways. The first way is to use the SPI interface in order to control its registers, and more importantly the Digital to Analog Converter Register. The registers controlled by SPI can be changed in order to change the program the AFE031 TX gain and its frequency response, the Rx gain and its frequency response, and various power saving mode [49].

Along with the SPI pins there are 5 additional pins allowed to be used. The DAC pin, SD pin, TX Flag, RX Flag, and INT pin. The DAC pin is used in order to put the AFE031 pin into DAC mode allowing the SPI interface to write directly to the DAC register. This allows for the output of the AFE031 chip to be controlled directly by this register. The SD pin is used to enable or disable the whole AFE031 chip including the SPI registers. This puts the chip into sleep mode allowing it to go into its lowest power consumption mode.

The TX flag is used to know whether the chip is ready for transmission. If the TX PGA, and the TX filter are configured correctly and the device isn't busy already transmitting then the TX flag will be low. Otherwise it will go high to signal that it is busy or not configured. The RX Flag is used to know whether the chip is ready to receive. The same parameters are checked for the RX configuration as are for the TX configuration. The INT pin is used to detect interrupt conditions. The chip can be programmed to interrupt on Current Overload or Thermal Overload.

The user can program the maximum output current of the chip's power amplifier with the external resistor (R_{set}) on pin 46. The thermal overload of the chip can be triggered whenever the junction temperature exceeds 150 °C. If this happens then the output stage of the power amplifier is disabled.

The transmission line of this chip goes through 4 different stages. The DAC, the TX PGA, the TX filter, and the power amplifier. So the first step is to go through the 10-bit digital to analog converter. Then the output of that goes through the TX PGA which is an amplifier which can be set to -12db, -6db, -3db, and 0db. Then the output goes through the TX filter. Finally the signal goes through the power amplifier which is capable of generating 26V at 1.5Amps. Here is the sample diagram for the transmitting stages.

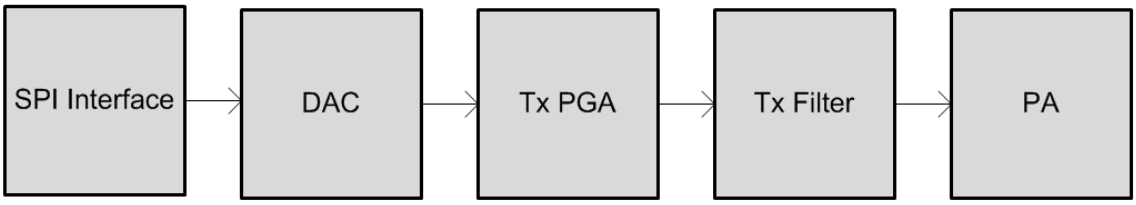


Figure 22 DAC Data Path

The PLC will interface with the main controller as described in the next paragraphs. The PLC interface is arguably the most important aspect of the design project. Since this technology is the foundation of the distributed sensor design, it was decided to create one PLC board that works well, then communicate with it using a serial interface. This board will then connect to the main controller, as well as each of the peripheral devices.

PLC was chosen for the communication method between the controller and the peripheral devices for several critical reasons. First, it was important to have a communication standard that didn't require lots of renovations to existing households. PLC allows this because the AC electric wires are already installed in the house, and therefore the data connection and power connections for the peripheral devices already exist. Second, the use of a single wire connection between the master device and the slave devices is very desirable due to the difficulty of connecting one wire for every peripheral device. In a full scale implementation, there could be hundreds of smart switches, load controllers and air flow controllers, making it impractical for any type of communication bus requiring more than one line. The disadvantages of data rate and bus traffic were mitigated by the fact that there is no need for large amounts of data to be transferred, but rather only occasional updates regarding temperature and lighting conditions.

4.1.7.1.9.2

Mains Transformer

Since the mains electricity is 120VAC, our sensitive digital electronics do not easily interface with it. That is why we need a coupling transformer that couples into the mains electricity to send and receive signals. From the AFE031 datasheet, Texas Instruments recommended Coilcraft's DA2032 transformer. It is a small sized, SMD, flyback transformer with a winding ratio of .013:1.6 primary to secondary. We will put the secondary on the high voltage side which will give an output on the primary of 1.38V when the mains peaks at 170V.

4.1.7.1.9.3

Circuit Protection

As previously stated, the mains electrical grid is not a very electrically clean. It is subject to voltage changes and line transients. We must protect our circuit against voltage spikes. We are basing our design off of the documentation found in TI's *Analog Front-End Design for a Narrowband Power-Line Communications Modem Using the AFE031.* On the mains side, we use a metal oxide varistor, which is a type of resistor that has high resistance until a certain voltage level is reached, then the resistance decreases. When the resistance decreases, it

allows the MOV to absorb a pulse. The MOV suggested in the datasheet is the Littlefuse TMOV20RP140E. On the low voltage side we use a transient voltage suppressor (TVS) which is useful to quickly clamp any over voltages to ground. The TVS we use is P6SMB18CA by Littlefuse. It will shunt away any transient's great than 15.3V. Schottky diodes, are used to steer current away from the analog front end because of their low voltage drop. A Zener diode is on the supply to further prevent over voltages.

High Voltage Capacitor

The high voltage capacitor is used to block the low frequency mains voltage. With the windings of the transformer, the capacitor forms a voltage divider.

4.1.7.1.10.4

$$HV_{CAP} = \frac{VA_{Limit}}{V_{AC}^2 \cdot 2\pi \cdot f}$$ *Equation 6*

According to the equation, a 1.84µF is calculated. However the datasheet recommends a .47µF capacitor.

PLC AFE Schematic 1

PLC AFE Schematic 2

4.1.7.2 Smart Switch

4.1.7.2.1 Microcontroller (MSP430)

The MSP430 is a microcontroller made by Texas Instruments that is advertised for its “ultra-low” power consumption. On sleep modes it can draw less than 1 uA. [51] It is a 16-bit RISC CPU. It can be programmed in C or in its own assembly language.

The MSP430 is perfect for the smart switch in this system. Some of the MSP430 devices include hardware made for capacitive touch. Along with this some of the MSP430 come with built in analog to digital converter which is very useful for monitoring the analog sensors on the smart switch. These include temperature, microphone, and infrared.

The chip chosen for the smart switch is the MSP430FR5739. The MSP430FR series feature a new feature from TI known as FRAM. This is ferroelectric RAM. What is special about FRAM is its extremely low power consumption. Lower than flash. Since the goal for this project is to reduce the amount of power consumed by a house we have to make sure that the components added in like the smart switch draw very little power. Along with this it has extremely fast writes and high endurance (can be written too many times) but these two features aren’t necessary. The low power consumption is the main focus.

Any of the value line MSP430 chips would work for the smart switch but once again the goal is to have the least power consumptions. The chosen chip has 14-channel 10-bit analog to digital converters. This is important since some of the data being read like temperature, light, and microphone is going to come in as an analog value.

4.1.7.2.2 Humidity Sensor

The humidity sensor used in this project will be the HIH-4030 by Honeywell. This is an extremely easy to use analog sensor. Only three traces are required and they are power, ground, and output voltage. The output voltage needs to have a pulldown resistor since it needs a minimum load. The following figure is the schematic for it.

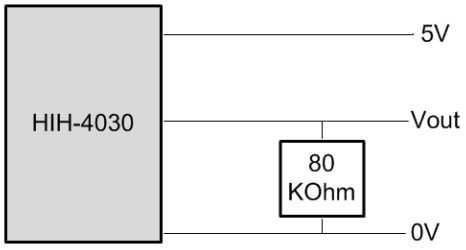


Figure 23 Humidity Sensor Schematic

The output voltage has an equation of

$$V_{OUT} = V_{Supply} \cdot (.0062 \cdot (\text{Relative Humidity}) + 0.16) \qquad \textit{Equation 7 [52]}$$

This gives us a nice linear voltage directly proportional to the supply voltage we put in which is going to be 5V.

4.1.7.2.3 Microphone

The microphone will be used in this system in order to pick up sound cues like clapping, or even voice in order to control the house. This is just to add more convenience and versatility for the user. The microphone data will go straight into the MSP430 and the MSP430 will be responsible for deciding whether certain data from it is useful or not. Whether it crosses a certain threshold.

The microphone used for this project will be the SPU0410LR5H-QB microphone from Knowles Electronics. This is a mems microphone. This microphone works from 2.7V to 5V [53]. In order to use this microphone with the analog to digital converter from the MSP430 though it needs to be amplified 100 times in order to better interpret the data. For this we will be using Texas Instrument's TL972ID operational amplifier. This is a simple supply, dual rail to rail amplifier [54] so we can supply power to it with the 5V and ground from the MSP430 making it extremely convenient in our design.

First thing we need to do is have an offset voltage of 2.5V in order to correctly get the positives and negatives of the fluctuation. Since sound travels as waves that both increase and decrease the voltage we can't have it at 0V or it will go below and the MSP430 does not support reading negative voltages. So we will create a difference amplifier where we will have a voltage divider between two of the same value resistors going into the positive terminal. This will offset the output up by 2.5V. On the negative terminal we will simply have an inverting amplifier with a gain of 100. We use high resistor values like 1M ohm in order to limit the high current spikes that microphones like these can have. We have a trimmer potentiometer to be able to adjust the gain as necessary.

Along with this we will put filtering capacitors and resistors to correctly draw out the frequencies needed from audible noise (which is from 20Hz to 20kHz) in order to filter out electrical noise coming in. Another capacitor will be put in series with the resistor holding the 2.5V into the positive terminal in order to have a stable, non-oscillating DC voltage. We will have a two stage filter/ amplifier to achieve the best performance possible.

4.1.7.2.4

4.1.7.2.4 Capacitive Touch

Chip Selection

Capacitive touch has gained popularity do to smart phones and other devices with touch interfaces, so our choices on how to power the capacitive touch is tremendous.

The first option we came across is using the TI MSP430 Value line microcontrollers. The value line come with peripherals built in for the oscillator method of capacitive touch. The MSP430 even has example code with it to show exactly how to implement capacitive touch elements. However given the

complexity of this project any extra code is something else to go wrong and will take valuable time to debug.

Looking for an off the shelf capacitive touch solution, we came upon Atmel's QTouch series. They are a line of complete capacitive touch controllers with a wide range of touch channels, sensor configurations, led drivers and even haptic drivers. We decided to go with Atmel's AT42QT1085 because it has eight touch channels, 12 GPIOs and a haptic engine [55]. We plan on having two sliders, which each take 3 channels and a proximity sensor which is a single channel. In addition, behind each sensor we plan to install LEDs, so we can take advantage of the GPIO pins. To make the smart switch more realistic, we are going to include haptic feedback, which with the haptic engine on the chip, it should be easier to deliver the correct pulses. Each of the channels has an individually adjustable sensitivity level via the SPI interface.

Design

4.1.7.2.4.2 Even though we are using an off the shelf IC dedicated to capacitive touch, there are many factors we have to take into account when we design our smart switch. Since the electrodes are going to be pads on our PCB, we need to take into careful consideration how we lay out our design and PCB. Since our design is to replace an existing light switch, we must stay within the physical bounds and standards already set of a decorator style toggle light switch. We want to incorporate two sliders, one vertical, and one horizontal. The generic layout is in Figure 24.

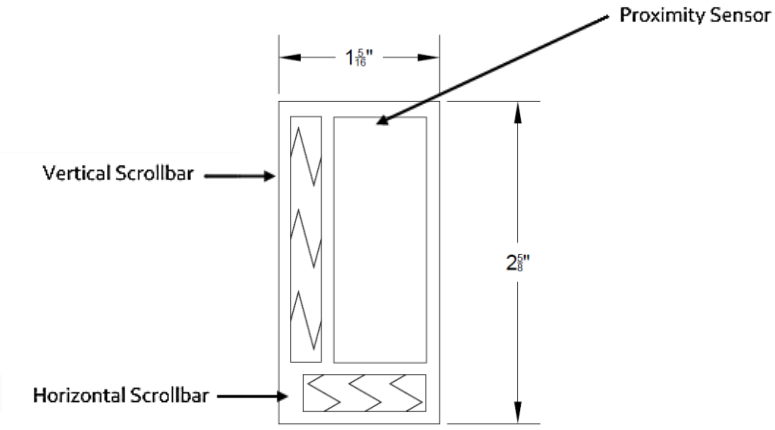


Figure 24: Touch Pad Layout for Decorator Switch Size

4.1.7.2.4.2.1 PCB Pad Design

There are two types of capacitive touch sensors, self-capacitance and mutual-capacitance. Self-capacitance which involves one electrode and mutual-capacitance which involves two electrodes. Since the datasheet recommended self-capacitance sensors, that is what we are using. When designing the capacitive touch sensor the formula for capacitance is extremely helpful:

$$C = \frac{\epsilon_0 \times \epsilon_r \times A}{T}$$ *Equation 8*

Where ϵ_0 is permittivity of free space, known to be 8.85×10^{-12} F/m. ϵ_r is the permittivity of the dielectric, which in our case is the PCB board, FR4 which has a dielectric constant of 4.2. The A is the area of the pad and T is thickness of dielectric, which is the PCB and our boards will be 63 mils (1.6mm). A pad size is a delicate battle between having a large area to sense and small enough making the capacitance low enough for it to be sensitive [56]. The grounding for a capacitive touch sensor is another trade off situation. Grounding planes provide noise resistance, but at the same time they add a lot of parasitic capacitance, lowering the sensitivity of the sensor. According to the data sheet, it is recommended to use a 50% to 75% ground hatch behind the sensors in a noisy environment. Since this touch panel is going to be a part of our smart switch and will be shoved in a junction box with noisy AC lines, we are going to use the 75% ground hatch.

4.1.7.2.4.2.2 Proximity Sensor Design

We would like to use a proximity sensor with our switch so that when a user is close and it is low lighting conditions, the switch will illuminate. With the Atmel QTouch chip, it is easy to implement a proximity sensor. According to the data sheet and Atmel's Proximity Design Guide [57], we make a large plane object and increase the sensitivity of the touch channel. Atmel claims that with a large enough sensor, it is possible to obtain a proximity detection of 250mm (9.8 in). The proximity sensor, by default will detect 360°, which can give false reading if installed on a thin wall which would detect a presence on the other side. To ensure the proximity sensor is omnidirectional we included a ground plane with a 30% fill on the opposite layer of the touch sensor.

4.1.7.2.5 LEDs

We want our Smart Switch to cater to both form and function. With that in mind, we want to include many LEDs so the user can have the switch have colored indications, or used to match a room or setting. The plan is to have red LEDs behind the keys or sensors of the touch panel to illuminate as the user slides and selects. To do this, we are going to use a specialized type of LED package known as a backfire LED. A backfire LED is mounted on the reverse side of the PCB and shines through a hole in the PCB; this is shown in Figure 25. The backfire LED allows for there not to be any components on the sensor side allowing a perfectly flat mate with the plastic panel.

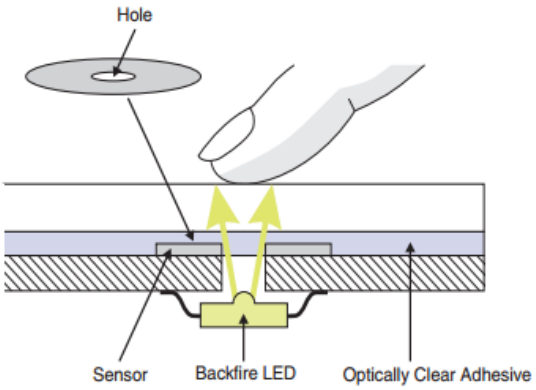


Figure 25: Backfire LED (Atmel QTAN0079)

Since the front panel of the touch control is to be made of clear acrylic that is engraved, we plan to side illuminate the acrylic with RGB LEDs. The edge illumination will activate when the proximity sensor triggers, that is when a human is close to interaction with the touch panel. The side illumination will provide two benefits, first, it will highlight the engravings in a selectable color, and second it will provide a soft ambient light when the user is near. An example layout of the RGB LEDs is seen below in **Error! Reference source not found.**.

Figure 26: Smart Switch Edge Light LED Layout

The backfire LEDs of that are behind the touch panel will be driven with the GPIO of the QTouch controller since we plan on having them having discrete on off states. The side light LEDs are another obstacle because we want to be able to dim and mix colors. To drive the RGB LEDS, we are going to use a TI LED 24 Channel PWM Driver, specifically the TLC5951.

The TLC5951 is a 38 pin device that takes in values for each output via serial input. The chip has three groups of eight channels, or one group per color of RGB LED. Each channel is individually adjustable with 4096 steps (12 bit) or PWM control, and 7 bit current control. Each group has an 8 bit brightness control. The driver also has feedback for open LED detection as well as thermal overload, though we are not going to utilize these functions.

The PWM grayscale data for each channel is shifted into a 288 bit (12 bit * 24 channels) shift register MSB first. The data has is structured in a way shown in **Error! Reference source not found.**.

LSB									MSB
0-11	12-23	24-35	36-47	48-59	60-71	…	252-263	264-275	279-287
R0	G0	B0	R1	G1	B1		R7	G7	B7

Table 7: TLC5951 Shift Register Data Structure

The register data is output when the grayscale latch (GSLAT) makes a low to high transition.

To current limit the LEDs, the chip has an IREF pin which is connected to the ground via a resistor. The formula for the resistor from the datasheet [58] is:

$$R_{IREF}(k\Omega) = \frac{V_{IREF}(V)}{I_{OLCMax(mA)}} * 40 \qquad \textit{Equation 9}$$

Where V_{IREF} is the internal reference voltage at 1.20V. I_{OLCMax} is the maximum current for any of the LEDs, which we are setting to 30mA. This gives us a R_{IREF} of 1.6kΩ. The current limiting resistors for each color can be found in **Error! Reference source not found.**.

Color	Forward Voltage	Forward Current	Limiting Resistor
Red	2.2V	30mA	55Ω
Green	2.2V	30mA	55Ω
Blue	3.3V	20mA	10Ω

Table 8: LED Current Limiting Resistor Calculations

4.1.7.2.6 Panel Design

For both esthetics and usability, the front of the switch is a clear acrylic plastic sheet. It will be side lit with right angle LEDs and rear lite behind each button with the rear backfire LEDs. It shall be composed of two layers, the first being the layer the LEDs shine into and will be larger than the second. The second will be the size of a decorator switch and shall fit into existing outlet covers. The back of the first layer will be tinted with a white spray tint. The top panel (#2) will be engraved via a laser cutter, and the right angle LEDs will illuminate the engraving. Each of the two acrylic panels will be cut to exact size using the laser cutter in the Texas Instruments Innovation Lab.

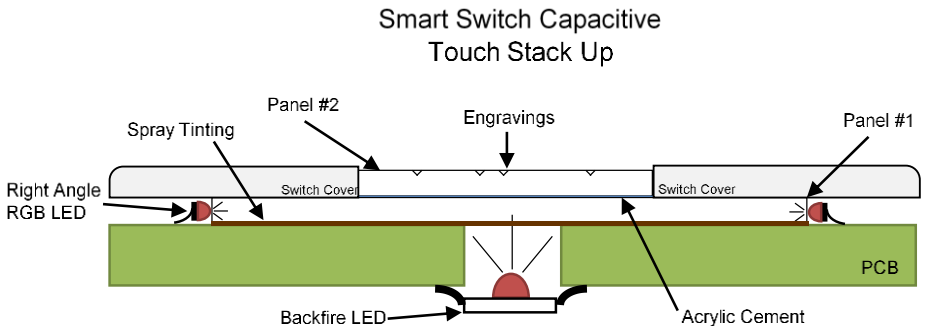


Figure 27

Acrylic

4.1.7.2.6.1 Acrylic, or Plexiglass as it is commercially known is a clear plastic that is commonly found and relatively easy to work with. Acrylic is lighter than glass, scratch resistant and will not yellow or fade in extended UV exposure. For our panels, we are going to use two pieces of 1/8" acrylic on top of one another for a total 1/4" thickness.

Bonding

4.1.7.2.6.2 For ultimate reliability and performance, the external acrylic panels must be uniformly flush with the capacitive touch PCB. To do this, we must bond the acrylic to the PCB. We are going to use an adhesive type called Liquid Optical Clear Adhesive (LOCA). LOCA is commonly used in bonding touch digitizers to LCD screens in smartphones. LOCA is UV curable, so we will apply it, put the panels under compression and subject the stack up to a UV source. We could use either a black light or natural sunlight as our UV source. To glue the two pieces of acrylic together, we are going to use acrylic cement. Instead of being an adhesive, acrylic cement softens the acrylic it is applied to, and effectively

4.1.7.2.6.3 makes the two pieces form into one, this is called plastic welding.

Engraving

The top acrylic piece is engraved on the back side with a mirror image of our desired graphics. The mirror image of the graphics is so from the top, the graphics look normal. We chose to engrave on the backside so that the outside surface the user interacts with is completely flat and does not have the texture changes of engraving. To engrave, we are using the Epilog Laser cutter in the Texas Instruments Innovation Laboratory.

4.1.7.2.7 Temperature

One of the most important information needed to collect from the house is temperature. One key difference between this system and other systems is that this device will be measuring temperature from multiple rooms rather than just one. There was a lot of things to be considered when picking temperature sensors like accuracy, power consumption, but the most important was interface.

Sensors usually come in either an analog interface or a digital interface. A digital interface means that you need to have some device that can send and receive serial communication like SPI and I^2C to talk to it. Usually these devices come with a multitude of options to change accuracy, put it in stand-by mode for low power consumption, etc. The problem with these is that they might require more extra hardware then desired. Along with this it is harder to integrate them because they require more space on your PCB and more code in your software.

With these reasons said the chosen interface was analog. Analog sensors are the easiest to integrate while also taking up little area. The smart switch device has a lot of things going in it with a very small area to work with. Analog sensors will make the PCB design easier giving more space for other traces. The only external hardware required for analog sensors is a pull-up/pull-down resistor.

The device chosen for temperature is the TMP36. It is a temperature sensor that outputs a voltage linearly proportional to the temperature. It can be powered from 2.7V to 5.5V. [59] The output voltage follows the following equation.

$$^oC = 100 * V - 50$$

Here is the schematic for it

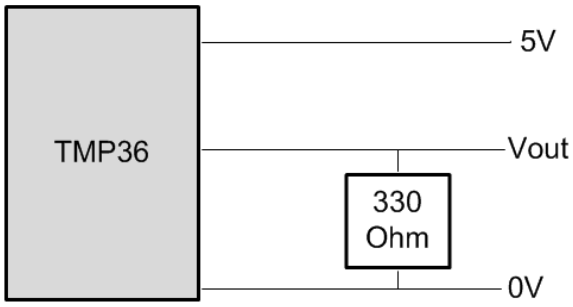


Figure 28 Temperature Sensor Schematic

4.1.7.2.8 Passive Infrared

Passive infrared is a sensor meant to detect motion. It does this by measuring the amount of infrared radiation coming out of the objects in its field of view. This is a very convenient sensor to have when you are talking about home automation. Pre-built PIR modules usually come in a digital interface where there is an interrupt signal knowing when the information changed. There are more advanced PIR systems where you can get a lot more data from it where it tells you the location of the objects and where they are going. This won't be necessary for this project.

The PIR module used in this project will be the SE-10 from Sparkfun. This is a simple PIR that only needs 3 wires; power, ground, and Alarm. The alarm signal is a digital interrupt that pulses when there is change. This module can be powered anywhere from 5 to 12V. So it will be powered by 5V to match the voltage of the MSP430. It has 120 degrees of view. The only external hardware needed is a pull-up resistor for the alarm signal since it is an open collector.

Smart Switch Capacitive Touch Schematic

Smart Switch LED Driver Schematic

Smart Switch Sensor Schematic

4.1.7.2.9 AC to DC Power Conversion

The goal of our project is for all of our modules to be plug and play. Since they all interface with US mains electricity, we have to convert the 120VAC to voltages for our digital components. We need 15VDC for our power amp on the PLC chip, 5V and 3.3V for different digital logic.

Design

We went with a traditional transformer and full bridge rectifier circuit. The design we have done is a test design because none of us have experience with power supply design. We used a Coilcraft JA4429-AL transformer which has a turn ratio of 1:0.24 [60]. This outputs 24VAC at a maximum of 1A. Instead of using discrete diodes in our full bridge, we opted for a single, contained package design and use a DF02S, a 1.5A full bridge rectifier by Fairchild Semiconductor. We use a 24V Zener diode because our initial simulation showed a large voltage spike on startup. The 24V Zener will clamp any voltages over 24V to ground. We will probably have to increase the Zener voltage for power dissipation reasons, however that will be determined empirically.

4.1.7.2.9.1

4.1.7.2.9.1.1 Regulators

We opted for switching regulators instead of linear because we have the potential to need large amounts of power and we are translating between large voltage gaps. Since switching regulators are more efficient, there is less worry about heat dissipation. For our initial design we decided to use three LT8610 switching regulators for Linear Technology [61]. We opted for three for easier debugging, which is if one of the regulators is damaged we can remove it and the other two should be un-affected (applies only to 5V and 3.3V regulator.) The steps we take to transform the 120VAC to our digital voltages can be seen below in Figure 29.

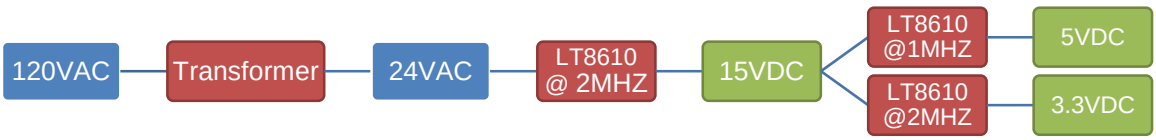


Figure 29: AC to DC Conversion Process

The LT8610 has an enable pin to turn the chip on whenever high, we tied it to V_{CC} because we have no reason to take up extra pins on our microcontroller by turning the regulators on and off. The regulator has a power good pin to indicate when the power output is stable, however this is another unnecessary function for our purposes, so we just tie it to a test point encase we need it for debugging purposes. The frequency is adjustable between 200kHz and 2.2 MHz by changing a resistor between the R_T pin and ground. The higher the frequency, the lower the efficiency, but also the smaller the components. We opted to try and make or test board a $10 board with Osh Park which means we

were limited to a 1”x2” for our circuit so we opted for smaller components and less efficiency. The output voltage is programed with a voltage divider between the feedback, output and ground. To find resistance values for the divider we used a 1MΩ resistor for R_1 and the equation below from the data sheet to find R_2:

$$R_1 = R_2\left(\frac{V_{out}}{0.970V} - 1\right) \qquad \text{Equation 10}$$

For most of the design, we copied reference designs from the LT8610 datasheet.

Simulation

We used LTSpice to simulate the rectification part of the circuit to plan what diodes, and capacitors we may need. Since LTSpice is a free program, the usability is limited, however it gives a good ball park estimate on what to expect from the circuit.

4.1.7.2.98

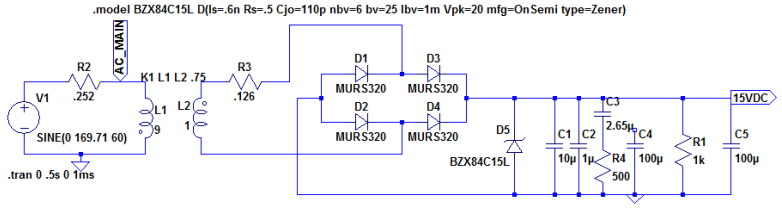


Figure 30: Basic Rectifier Circuit Schematic in LTSpice

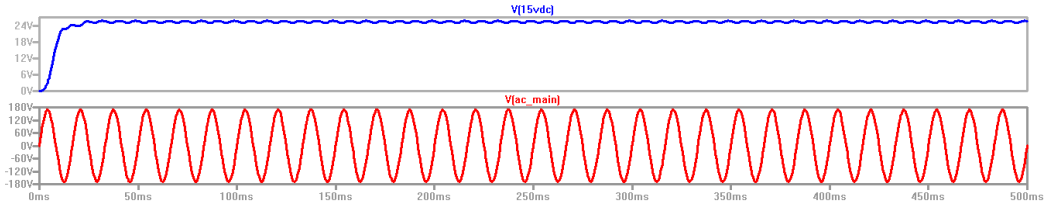


Figure 31: Simulation showing AC Mains (Red) and rectified 15VDC (Blue)

PCB

This is the first circuit board that we have had successfully sent to fabrication for this thus far. We designed it to be a $10 board by Osh Park. At $5 per sq/in, we decided on a simple 1”x2”. With all the large parts, especially the transformer, it was hard to fit and route everything on a two layer board. The component density is fairly high. Since it was mostly power, we decided to use large traces and large copper pours. This does two things, first it can handle more power, and second it helps dissipate heat from the three switching regulators. The board simply takes in 120VAC via two terminal blocks and then will output 15, 5, and 3.3V on the other side. If this board works, we will use it as a power module for our remaining designs. It should simply stack behind our other boards with its .1” headers. One thing we had to take into account was the live mains voltage on the board, we had to have larger isolation between all AC Mains nets/ vias and the copper ground pours because they have a higher chance of arching/ being shorted.

PCB Layout (Actual Size)

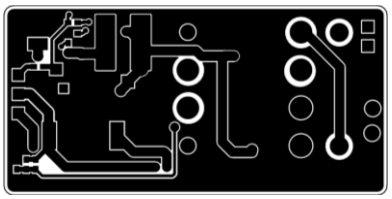


Figure 32: ACDC Top Copper Layer

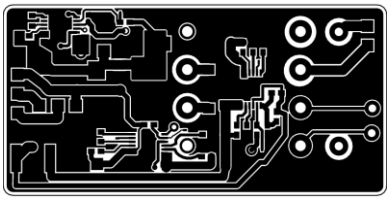


Figure 33: ACDC Bottom Copper Layer

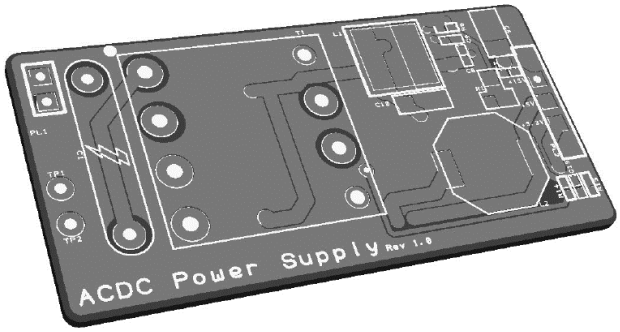


Figure 34:3D Approximation of PCB (Top Shown)

ACDC Schematic

4.1.7.3 Airflow Control

In a residence with the main controller and smart switches installed, the controller is able to collect detailed information about the environmental conditions of various rooms of the house, but is unable to directly impact the environment of individual rooms. The air flow controller allows the system to direct the air of the desired temperature to the rooms where it is needed most. This is accomplished by using a motor to open or close a vent that is installed in the ducts leading to each room. This piece of the home control system will need to be installed by a contractor or during initial construction.

4.1.7.3.1 Microcontroller

The microcontroller chosen is the MSP430G2955 [62]. This particular MSP430 is from the value line, meaning that it is small in size, low power and cheap. Given that this microcontroller will not be performing any complicated or processing intensive tasks, choosing a simple, small microcontroller is ideal.

The choice of the MSP430 line of microcontrollers was primarily influenced by the familiarity with the microcontroller from the coursework completed. The particular microcontroller comes in two packages, being a VQFM 40 pin package and a TSSOP 38 pin package. We chose the 38 pin TSSOP package because all the pins were on two sides of the chip, leading to easier soldering.

The 38 pins of the microcontroller selected include two power pins, two ground pins, a test (JTAG) enable pin, a reset pin, and four (4) ports of eight (8) multipurpose pins [62]. All of the 32 pins in the ports are general purpose IO pins, but they each have at least one other function, such as serial communication, timer, analog-digital communication or JTAG support.

The air controller functions required are two analog inputs for the temperature sensor, debug buttons, JTAG interface, SPI communication to the PLC communication card, an analog input for reading the PLC chip data, timers for putting steps into the motor driver chip and numerous digital IO pins for configuring the motor driver chip's settings. Debug LEDs will be added prior to manufacturing if there are available pins.

An important consideration in the use of this chip is the overlap of pin functions. The MSP430 chosen, as can be seen in the schematic (page 1) at the end of this section, have many functions assigned to each pin, but not every pin has the same features [63]. For example, port 1 pins 4, 5, 6, and 7 are the only pins that are capable of using JTAG interface, but are also IO pins connected to timer A. Therefore, it's not possible to use the JTAG and timer A compare at the same time. Other important overlaps for this chip is the serial communication port in port 3 and the external crystal inputs in port 2.

4.1.7.3.2 Power supply

The airflow controller will require multiple different voltages to operate all of the separate functions correctly. Luckily, the voltage and corresponding current requirements are very similar to the voltage and current requirements of the main

controller Beaglebone cape schematic. A brief breakdown of required power terminals is described as follows.

Identical to the LCD screen, the stepper motor will operate at 9V, 1A [64]. The MSP430 will operate at 5V, which is also the same as the Beaglebone. Both the main controller and the airflow controller will communicate using the same PLC board, which requires 15V and 2A.Therefore there are no appreciable differences in the overall power requirements of the main controller and the airflow controller. However, when considering overall power use the main controller will consume more because the LCD screen will have a much higher duty cycle than the stepper motor, which might open and close several times a day as opposed to once or twice an hour for the LCD screen. The PLC communication will similarly have a lower duty cycle because the vent will only return status or temperature data when queried, as opposed to the main controller that is sending signals much more often.

Overall, the power supply circuit will be very similar to the main controller power supply in terms of components used. Both will draw power from the 120VAC household main and transform, rectify and regulate that voltage to usable DC levels. However, the regulator supplying 5V will be much smaller because the MSP430 draws much less power than the Beaglebone. The 15V supply will be identical. The only difference between the 9V supplies will be that the airflow controller 9V regulator will have a much larger output capacitor in order to handle the rapid power demands of driving a motor.

4.1.7.3.3 Clock Source

The MSP430 has an internal clock source, and our design intends to use this internal source. However, the use of a real time clock may be required during the course of the prototyping or during testing. Therefore a clocking circuit was created and included on the schematic. The clock circuit consists of a 32kHz crystal, two load capacitors and two zero ohm resistors in case the impedance needs to be tuned. The values were selected according to a MSP430 clock circuit reference provided by Texas Instruments [65].

4.1.7.3.4 Motor

Stepper motors were the right choice because the movement required was point to point rather than torque controlled or speed controlled, which would have favored servo motors or DC motors, respectively [28]. The high starting torque of stepper motors was also ideal to overcome the mechanical frictions of the air damper’s gearbox.

Estimating the required torque was difficult given that the air damper had not arrived in the mail at the time of designing. Subjectively, the torque required is likely very low because the vent is 6” diameter and the rotational axis has bearings, and the motor will be driving through a gearbox.

The stepper motor chosen is a small motor made by Mercury motors [64]. It is rated to 12V and .33A, but good motor design practices allows for much higher

starting currents to ensure the motor will be able to begin movement. It is a bipolar stepper motor with a four wire interface. The bipolar was chosen because it offers better power per unit area than unipolar motors. Each step angle is 1.8 degrees, meaning that there are 200 total steps available in a full rotation. The precision of movement is not a critical requirement so the relatively large step size is acceptable. The maximum detent torque is .016 Newton-Meters, which is comparable to a light twist with the fingers. This should be more than enough to move the vent damper. A picture of the motor is shown below.

4.1.7.3.5 Motor Driver

The motor driver IC is the TI DRV8818, a motor driver intended for small, low current stepper motors [66]. This integrated circuit has the H-bridge transistors inside the chip, whereas most high current motor drivers have the transistors outside the integrated circuit.

This motor driver has an indexer built in, which means that the movement is controlled by two signals, direction and step. To move the motor, one simply sets a directions with a digital logic level, and then whenever the step pin receives a rising edge the motor is advanced one step [67]. The direction pin is attached to a digital IO pin on the MSP430, and the step pin is attached to an output pin on an MSP430 timer pin. This allows the MSP430 to control movement though a pulse width modulated signal rather than having the code set a digital IO pin high and low over and over, but either approach is possible with the MSP430 connected as shown in the schematic.

The motor driver also has various digital inputs for selecting different features. These include a general enable pin, a reset pin, a sleep pin, and two pins for selecting the level of micro-stepping [66]. These digital inputs are wired to general purpose IO of the MSP430, but it is unlikely that the application will require their use.

Other features include the ability to use a current “Chopper”, which monitors the overall current into the motor and limits it by modulating it over time. The mechanical inertial of the motor makes the pulsing of power unnoticed. The current limit is set via external resistors and capacitors. Pads for potentiometers were included for those pins to allow for easier tuning during prototyping, then once the ideal resistance value is known they can be removed and replaced with traditional surface mount resistors. Similar, the decay mode can be selected to either decay fast, slow or mixed [66]. H-Bridge decay controls how quickly the winding currents can drain to ground when the motor movement is arrested. Slow decay acts a brake in some configurations [68].

Connections to the motor will be done with .1” pitch screwed wire terminals. While not included in the schematic at this time, there will also be terminals and a header bridge switch to allow for external bench voltage to drive the motor.

4.1.7.3.6 Temperature Sensor

The airflow controller will use an analog temperature sensor, the TMP36 from analog devices, in the TO-92 package. The temperature sensor works as a voltage divider, with an analog input voltage, a ground voltage and a power voltage. The temperature sensor will be mounted in the air vent in order to return the temperature of the air going into a room to the main controller. The temperature sensor to be used is the TMP36 precision analog temperature sensor, which is also being used on the smart switch. The interface is simple, and all that is required is a single ADC input pin [69].

Terminals for two temperature sensors were included, so that if needed a temperature sensor can monitor the heat on the motor as well. This would allow the microcontroller to identify a fault condition in which the motor is stuck and cannot move, and cut power to the motor to prevent damage or a fire.

4.1.7.3.7 Communication and Integration

The airflow controller communicates to the main controller through power line communication, using the same device as the smart switch. Similar to the smart switch, the airflow controller cannot initiate communication with the main controller. An example of the communication process would be an addressed command sent from the main controller to the airflow controller telling it to close the vent. The main controller would then wait for a message back from the controller confirming that the message is received. The main controller would then send a query asking if the vent has finished closing or if there were any errors, and upon receiving that query the airflow controller will return a message stating the operation was either successful or unsuccessful.

Similar to the smart switch, the airflow controller will have a temperature sensor. This sensor will be not be mounted on the PCB, but rather on the inside of the air duct. This will allow the airflow controller to measure the temperature of the air as it arrives in the destination room. The communication protocol for the temperature query will be the same command as the temperature query to the smart switches. The main controller will send a single command asking for temperature, and the airflow controller will either send a temperature as response or a message indicating some type of error.

4.1.7.3.8 Packaging and Mechanics

The airflow controller will be the only aspect of the project to interact with the residence in a mechanical manner, making the packaging and mechanics a larger concern than in other aspect of the project. In order to reduce the difficulties involved in designing a mechanical enclosure, it was decided to purchase an existing product similar in function to the envisioned functionality of the airflow controller, then modify this product to interact with rest of the system in the desired manner.

The product purchased is the SunCourt ZO106 normally-open 6” air duct damper [70]. This product is shown below.

Figure 35: SunCourt ZO106 6" air duct damper (permission pending)

This product was selected because it has the needed mechanical stops and bearings to rotate the airflow disrupting element, and is mounted inside a standard size of HVAC duct. In addition, this duct comes with a reduction gearing system that allows a lower power motor to move the element regardless of airflow speed. In addition, the product comes with a 120V-24V step down transformer, which can be used in the power supply circuit for the motor driver [70].

However, several modifications will need to be made to the SunCourt damper. First of all, the damper was designed with a spring to return the damper to an open position whenever power isn't supplied to the motor. This is not ideal because if the goal is direct air to one room of the house, this would require holding all of the other airflow dampers shut, requiring a lot of power. Choosing the normally closed option wouldn't help either, since power would be required to cool or heat multiple rooms at once. To address this issue, the return spring will be removed, and the servo motor will be replaced with the previously mentioned stepper motor. The stepper motor is ideal because the structure of a stepper motor creates a higher non-powered holding torque [28]. Combine this holding torque with the frictional resistance of the reducing gearbox, and the system will be able to hold its closed position without requiring power. If this proves to be an issue, the speed-torque characteristics of a stepper motor will provide for another option for holding shut. A small rubber wedge can be installed at the closed end of motion, so that when the vent is closed and powered down there will be a mechanical hold. Since stepper motors have the highest starting torque per current of any

4.1.7.3.9 Schematics

Schematics for the PCB section of the airflow control are shown in this section. The schematic details the connections between the MSP430 microcontroller and the stepper driver integrated circuit, as well as the connections between the PLC communication card and the microcontroller. Also included on the schematic is

the initial power supply schematic that will provide the correct voltages to the PLC chip, microcontroller, motor driver and motor. This design is correct with regard to the microcontroller and motor driver, but needs to be checked and redesigned for the power supply circuits and the pin connections to the PLC chip, since the standard pin connection diagram shown in the main controller section was created after this design.

The first page shows the microcontroller and its pin connections. On this page there are several features worth highlighting. First, there are three external buttons included. The first is for resetting the microcontroller manually, the second and third are included to give more debug options and to assist early in the programming process. There are also terminals for wires leading to the temperature sensor to be soldered. Also, a set of headers are placed between the TEST pin (pulled low) and the 5 volt supply. The TEST pin determines if the JTAG interface is active or not, and the use of headers allows the user to place a header bridge between the two pins to enable JTAG at any time. The JTAG interface was given exclusive access to the pins defined as JTAG communication on the MSP430 to prevent any type of debug errors due to improper JTAG connections [62].

Airflow Controller Schematic Page 1

Airflow Controller Schematic Page 2

Airflow Controller Schematic Page 3

4.1.7.4 Load Control

4.1.7.4.1 Triac
For the full power of a 15 amp 120VAC circuit, a typical household circuit, we have to use a suitable triac. We came across a triac by NXP Semiconductors the BT139-600, it's a 600V, 16A triac [71] in a TO-220 package. The TO-220 package is beneficial because we can easily attach a heat sink and thermal paste to it to help dissipate any excess heat associated with large loads. The trigger voltage is 1.5V at 35mA which can easily be provided via a microcontroller pin and a discrete transistor.

4.1.7.4.2 Zero Crossing Detector
To determine when to fire the triac, we need to know when the zero crossing of the mains electricity occurs. To do this we use a zero crossing detector circuit which is composed of an AC optocoupler and a transistor. The zero crossing detector will send an interrupt to a microcontroller which will then change the fire the triac based off a timer module.

4.1.7.4.3 Current Transducer
To monitor loads we are going implementing a current meter, from which we can derive power consumption and from there, cost of operation. Since we are trying to measure the current through the mains lines, there is an inherent changing magnetic field. We can measure this changing magnetic field with a Hall Effect sensor. However instead of designing a loop, figuring out the magnetics, and normalizing to known current we opted for an off the shelf unit. We are going to use a Hall Effect current transducer by Allegro Microsystems, the ACS711. It is a QFN chip that is in series with the AC Line. The line resistance for is .6mΩ, so that with the maximum load it can handle (15.5A) [72], its power dissipation is only 144 mW. The chip converts the current nicely to a linear output voltage that can be read by an ADC pin on a microcontroller. The chip also features a pin for indicating an over current condition which we can tie in as an interrupt to our load controller and immediately shut down to prevent any further damage.

4.2 SOFTWARE DESIGN

4.2.1 Glove Control Software Architecture
The software for the physical glove is being housed inside the ATMEGA328 microcontroller. The ATMEGA328's goal is to collect the raw data from the accelerometer, gyroscope, magnetometer, and flex sensor as fast as possible. Then it has to package it in a 64-bit package and send it serially to the UART to Bluetooth converter so the Application Programming Interface can receive it, interpret it and process the data. The only goal of the physical hardware of the glove is to get the data at the desired speed and send it. Due to the physical limitations of the microcontroller (which was picked due to the little area since we want the glove to be compact) the raw data is processed through algorithms in the API.

Upon start up there are a couple of things that need to be done. Initializing the accelerometer, gyroscope, and magnetometer, and connecting through Bluetooth to the API. The very first thing that will be done is connecting through Bluetooth. If the Inertial Measurement Unit (IMU) devices aren't initialized then they are kept in a low power state therefore saving energy. If they were initialized before we were connected through Bluetooth then power would be wasted for data that is not going to be used at all.

When the Bluetooth Bees are trying to pair together a series of commands must be given. The master must first send ascii "\r\n+INQ=1\r\n" to start inquiring. When you inquire a device you will receive a message like "/r/n+RTINQ=18,E4,1B,63,D6,00\r\n". You will then send a message like "\r\n+CONN=18,E4,1B,63,D6,00\r\n". This will start connection to 18,E4,1B,63,D6,00 device. You will then receive a request for input which will be "\r\n+INPUT\r\n". This is requesting for the pin code. You can then send the pin code as "\r\n+RTPIN=0000\r\n" replacing 0000 with the actual pin code. If this is successful then the two devices will be paired. For the slave all you need is two messages. One to put it into pairing mode which is "\r\n+INQ=1\r\n". Then after receiving the right pairing code you send back "\r\n+RTPIN=0000\r\n".

In order to initialize the accelerometer, gyroscope, and magnetometer you need to write through I2C to a couple of registers to put them in the right mode. For the gyroscope you need to write 0x0F to the CTRL_REG1 register and 0x20 in the CTRL_REG4 register. This will put it in normal power mode, enable all axes, put it in 100Hz mode, and put it in 2000 degrees per second accuracy. For the accelerometer you need to write 0x47 to the CTRL_REG1 register and 0x28 to the CTRL_REG4 register. This will put it in normal power mode, enable all axes, run it at 50Hz, and put it in 8g mode which is the max amount of acceleration this chip can read. To initialize the magnetometer you need to write 0x00 to the MR_REG register. This will put it in continuous conversion mode and run it at 15Hz.

Once the device is initialized the data from all the sensors need to be constantly read. The flex sensors is just read through the analog pins of the ATMEGA328. The rest of the data is read by I2C messages. A message will be compiled every 10ms (at 100Hz) since that is the speed the gyroscope refreshes at and is one of the most important and sensitive data.

4.2.1.1 Application Programming Interface

The Glove Manager Application Programming Interface (API) needs to allow for the gesture recognition glove to communicate with an external device. It also must take the communicated information and process it to create a set of commands for a given application based off of gesture recognition. The functional requirements for this API are outlined in the following section.

4.2.1.1.1 Functional Requirements

Receiver

- Takes in raw IMU and flex sensor data via Bluetooth from the glove

Processing

- Converts raw IMU and flex sensor data into gesture recognition information
- Interprets gesture information into command data for mouse and keyboard control
- Interprets gesture information into command data for alternate interfaces
    - Home automation system
    - Google Glass
    - Oculus Rift

Transmitter

- Sends mouse commands to the computer for mouse control via HID protocols
- Sends keyboard commands to the computer for keyboard control via HID

4.2.1.1.2 Interface Control

The Glove Manager Application Programming Interface will contain an HID manager to deal with the HID protocols for the device the API is housed on. This will allow for the glove to directly control mouse movements, clicking, and keyboard inputs. In addition to this, the Glove Manager Application Programming Interface will have an Output Manager to deal with formatting data to send to other applications or devices such as the Oculus Rift.

4.2.1.1.3 Processing

The Glove Manager Application Programming Interface is responsible for taking raw sensor data from the glove and translating it into gestures which are further translated into control commands for external devices.

It will translate the raw data from the sensors into 3-axis acceleration, 3-axis radial acceleration, 3-axis relative magnetic field, and flex sensor data from 3 fingers. The Flex sensor data will be used to know the position of the fingers. The accelerometer and gyroscope data will be used to tell in what direction you are moving or rotating. The magnetometer data will be used to tell where you are facing relative to the North. With these gestures, commands can be created based on how they would translate into control of a computer or device.

4.2.2 Home Automation Software Architecture

The home automation software has a few main components that will be explained in detail—the backend Central Manager processing system, the server/DBMS, and the graphic user interface.

4.2.2.1 Central Manager Backend

The Central Manager backend is responsible for doing all of the processing of data in terms of commanding and controlling the system. It will take input from the user via the glove and the Qt GUI application in addition to remote input from the Smart Switches and Ethernet. Once given status and command inputs the Central Manager backend will be able to keep track of the system's health and report status data. If it is in Smart Decision Mode then it will also query the database for past user information and formulate adaptive commands to control

the pertinent devices. The follow diagram outlines the class structure of the Central Manager background and the interfaces it interacts with.

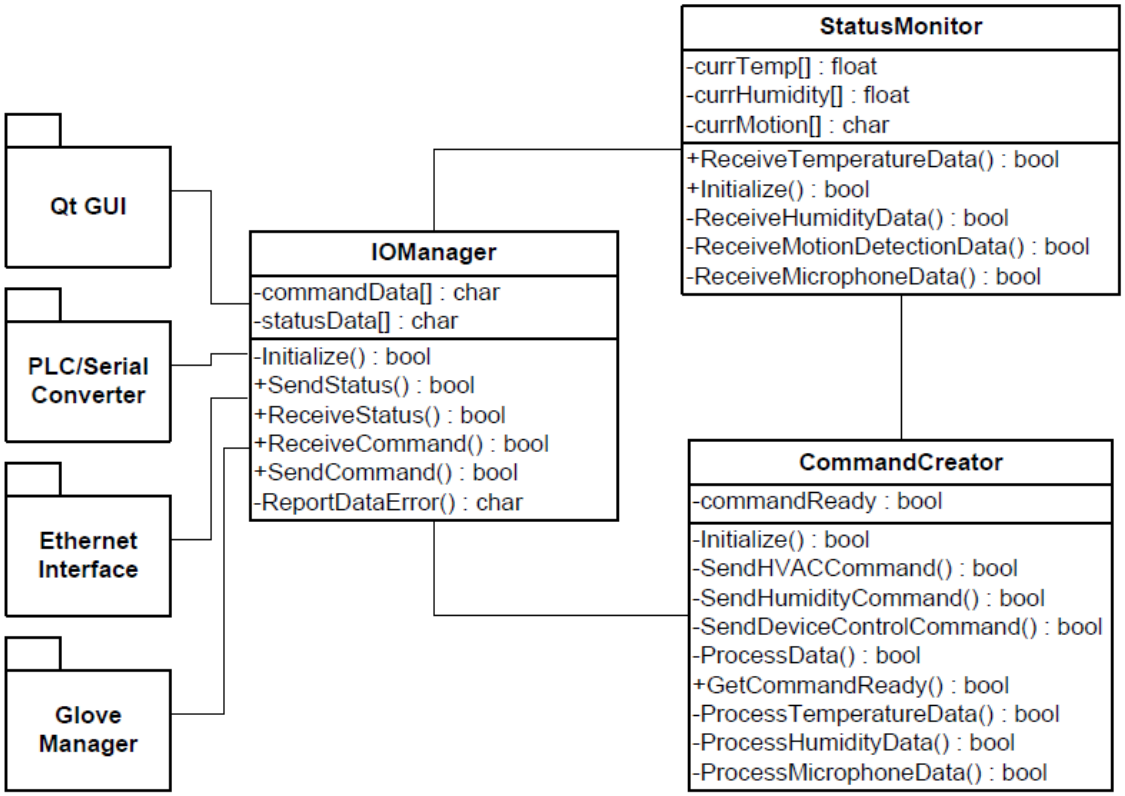


Figure 36: Central Manager Class Diagram

4.2.2.2 AI Adaptive Control Adaptation Parameters

The internal software system will have built in adaptation parameters to deal with fringe conditions and error handling to prevent the user from doing inefficient or incorrect commanding of devices. The following are parameters that will be hard coded in to maximize efficiency.

Parameter	Restraint
HVAC Ventilation Command	If commanded on, must remain on for a minimum of 30 minutes. If commanded off, must remain off for a minimum of 30 minutes.
Home Lighting Device Command	If commanded on, must remain on for a minimum of 5 seconds. If commanded off, must remain off for a minimum of 5 seconds.
Humidity Status	If humidity reaches dangerously high conditions for extended amounts of time, turn on ventilation system.

Temperature Status	If temperature reaches user defined maximum and minimum thresholds, automatically turn on/off AC/heater

Table 9: Adaptation Parameters

4.2.2.3 I/O Manager

4.2.2.3.1 Requirements

The I/O manager is a software component tasked with both sending and receiving command and status data. It shall sends sensor and usage status data to the LCD display in the form of a QT application, it shall send commands to the external devices and it shall send sensor and usage information to the database. The I/O manager shall receive command data from the Command Creator software component and it shall receive status data from the Smart Switches.

4.2.2.3.2 Interface Control

The main external interfaces to the Central Manager that the I/O Manager will communicate with is the PLC/serial converter. For the purposes of this project the database and LCD screen are considered parts of the Central Manager and therefore are internal interfaces. The following is the expected data transfer between the two entities.

TX Entity	RX Entity	Message	Size	Composition
I/O Manager	PLC/Serial Converter	Device Address	16 bits	House Unique ID (8 bits) Device Address (8 bits)
I/O Manager	PLC/Serial Converter	Command	16 bits	House Unique ID (8 bits) Command Bitfield (8 bits)
I/O Manager	PLC/Serial Converter	Data	16 bits	House Unique ID (8 bits) Data (8 bits)
PLC/Serial Converter	I/O Manager	Smart Switch Data Package	50 bits	House Unique ID (8 bits) Temperature (10 bits) PIR (10 bits) Ambient Light (10 bits) Humidity (10 bits) Capacitive Touch (2 bits)

Table 10: Interface Messaging

4.2.2.3.3 Processing

The purpose of the I/O Manager is to collect and distributed pertinent data from all internal and external interfaces. It will encode any command messages to be sent to external devices and assign device addresses accordingly. It will also be responsible for deserializing the Smart Switch data package and making all data

reachable by other software components in the Central Manager via a simple getter/setter protocol.

Furthermore, the I/O Manager will provide a level of security for home device control by appending a House Unique ID to every message sent out to make it more difficult for an outside source to tamper with device control. Every house will have its own 8 bit Unique ID that must be found at the beginning of every device control message before the information inside the message can be used. It is infeasible for our system to use an encryption mechanism due to time and processing constraints.

4.2.2.4 Commanding and Statusing

Commanding and statusing is made up of two main software entities, the Status Monitor and the Command Creator.

4.2.2.4.1 Functional Requirements

The Status Monitor shall take in data from each temperature sensor once per second. Every five minutes it shall calculate the average temperature to make smart decisions about turning on the AC/heater accordingly. Every hour it shall calculate the average temperature and send it to the I/O manager to be stored in the database. This hourly average data shall be stored in the database for two years for pattern recognition and adaptive commanding. The Status Monitor is also responsible for recording data from every humidity sensor once per second and every five minutes it will calculate the average. Lastly, the Status Monitor must take in motion detection data at any triggered even and save all motion detection data for two years in the database.

The Command Creator shall accept status data to control the HVAC system when temperature reaches a certain threshold by turning the AC/heater on/off. It shall also use pattern recognition to create commands for the AC/heater from long term data analysis. For the humidity sensor the Command Creator shall record data from every humidity sensor once per second and alert the user if humidity is at dangerous conditions. It shall take the average humidity for every five minute interval to make smart decisions about turning on the AC to combat high humidity. The Command Creator shall accept motion detection data to create commands related to turning on or off devices accordingly. It shall also use the long term data for pattern recognition and adaptive commanding. For event driven capacitive touch the Command Creator shall turn on and off devices at capacitive touch commanding.

4.2.2.5 Processing

The purpose of the Status Monitor is to receive status information from the I/O Manager and calculate averages for each sensor. The averages calculated every five minutes will be used for immediate feedback and command which means it will be sent to the Command Creator. The averages calculated every hour are used for long term pattern recognition and they will be sent back to the I/O manager to be stored in the database. The PIR and capacitive touch sensors

are the only sensors to not follow this protocol. They are both even driven and all event entries will be saved in the database for further analysis.

The Command Creator is in charge of receiving status data from the Status Monitor and checking for patterns that would warrant commands to be sent. This includes the monitoring of adaptation parameters and acting on fringe cases for those parameters, dealing with direct user commanding from the capacitive touch sensor in a Smart Switch or direct commanding from the user interface, and commanding based on pattern recognition.

4.2.2.6 Server Hosting

The embedded ARM processor on the Central Manager will host the server containing archived sensor data. The decision was made not to host the server remotely in case the system lost connectivity with the internet. Lighttpd will be used as the server because it is lightweight and there is a lot of documentation and tutorials available on how to use it. With the Lighttpd server the relational SQL database is hosted and the QtSQL module will be used in Qt for querying and interfacing with the database.

4.2.2.7 Database Management System Design

Business Rules

- Database must be reachable by the Central Manager itself as well as a web based application
- Statuses will be compressed after being received from the status monitor to save space
- Each Smart Switch will have zero, one, or multiple Sensor Devices incorporated in it
- A Smart Switch will have an ID and location
- Each Sensor Device will have a name, a type, an enabled flag, and a health status
- Each Sensor Device may be of the Temperature, Humidity, PIR, Ambient Light, or Capacitive Touch type
- A Temperature Sensor Device can have zero, one, or many Temperature Data Entries
- A Temperature Data Entry will have date, time, and temperature reading attributes
- A PIR Sensor Device can have zero, one, or many Motion Detection Data Entries
- A Motion Detection Data Entry will have date, time, and motion status attributes

Entity Relationship Diagram

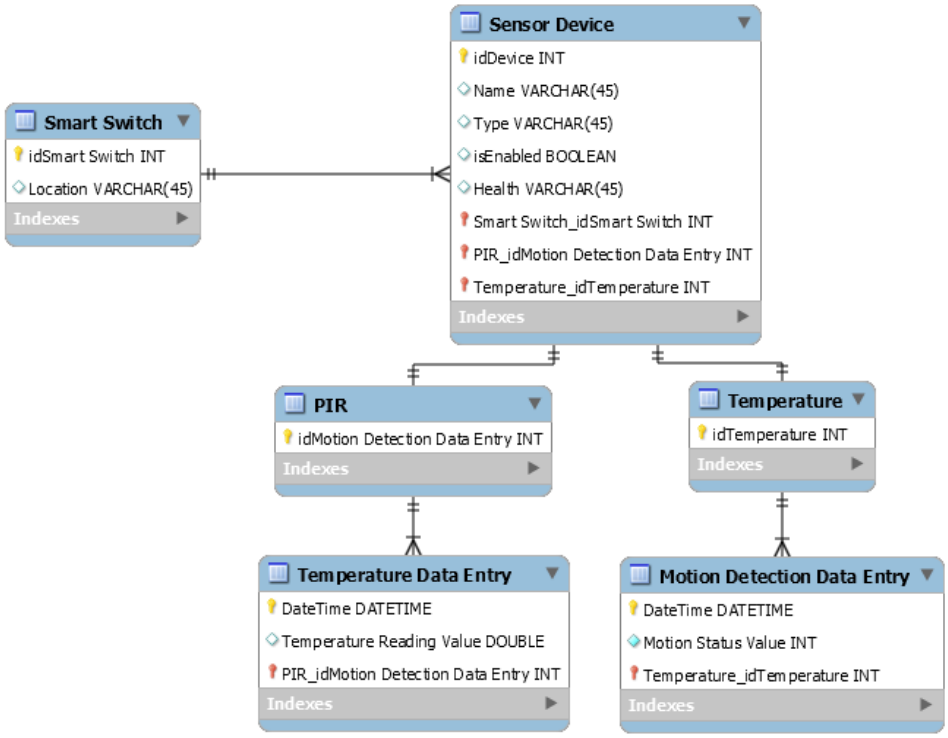


Figure 37: DBMS ER Diagram

4.2.2.8 Graphic User Interface Design

The user will be able to graphically interact with the home automation system via a graphical user interface housed on a 7 inch LCD display for the Central Manager. In addition to this, an application will be available to remotely control the system from the user's personal device.

Based on its ability to be cross platform to allow for the same functionality across many different types of personal devices including Android, Windows, and Linux operating systems on phones, tablets, and personal computers Qt will be used in the creation of our GUI front end and back end. The back end software for the Central Manager will be in C++ so using Qt in C++ will allow for a smooth transition between interfaces.

4.2.2.8.1 Functional Requirements

- Smart Decision Mode
  - Allow user to set smart decision mode to on/off
  - While smart decision mode is on, allows user to customize to turn on/off certain features and set thresholds for temperature, ambient light, and humidity event triggers
- Commanding
  - Allows for manual commanding of all devices in all rooms
- Statusing
  - Provide status data on all rooms, and all sensors in each room
- Setup

- Allows for initial setup of rooms, addition of new devices, moding of automation system, and glove setup

4.2.2.8.2 Organization

Most graphic user interfaces have a means of navigation in the application and an active window where a user can interact with the interface. The Central Manager graphic user interface will utilize Qt's QML Pathview as a means of navigation between active screens. This structure allows for the user to intuitively swipe on the screen until the desired index is selected to be in the active window. This is an alternative to tabs that aligns best with our goals of utilizing a touch screen and a gesture recognition glove to control the automation system. If the glove is in proximity to the Central Manager and a gesture is deteced of a horizontal swipe, it can also control this Pathview interface. The following figure shows a prototype of the Pathview element in the user interface where the active window will be the dining room but that can change with a clockwise or counterclockwise swipe.

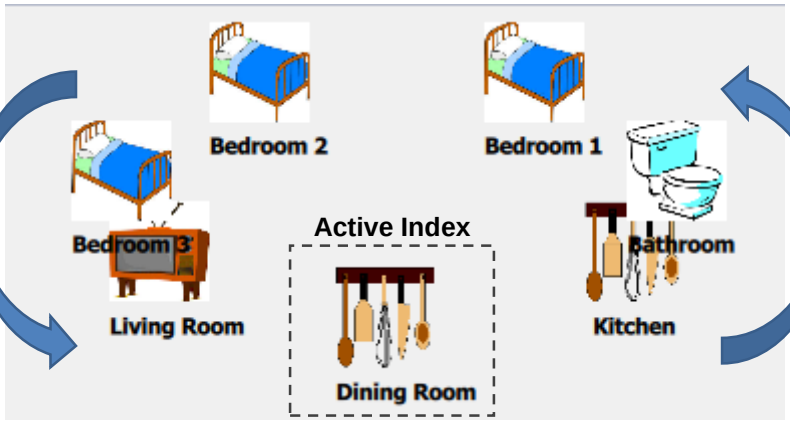


Figure 38: Pathview Element Example

The corresponding Qt code for this Pathview element is in the following figure.

```
21    PathView {
22        id: pathView1
23        x: 72
24        y: -36
25        width: 427
26        height: 254
27        z: 0
28        rotation: 0
29        scale: 1
30        delegate: Component { ...}
47        model: ListModel {
48            ListElement {
49                name: "Living Room"
50                sourceCode: "livingroom.jpg"
51            }
52
53            ListElement { ...}
57
58            ListElement { ...}
62
63            ListElement { ...}
67
68            ListElement { ...}
72
73            ListElement { ...}
77
78        }
79        path: Path {
80            startY: 200
81            startX: 240
82            PathQuad {
83                x: 240
84                y: 50
85                controlY: 150
86                controlX: 520
87            }
88
89            PathQuad { ...}
95        }
```

Figure 39: Pathview Element Code Example

The active window will be one of the following—homepage, setup and configuration page, room manager page, and help page.

The homepage is the default that will display whenever the user needs to access the Central Manager. It will have the current date and time, general statuses for average sensor data for entire house in terms of temperature, humidity, and lighting, and it will have any error report information to notify the user at first interaction with the graphic user interface. The homepage will not have any means for the user to control anything other than to switch to a different page via the Pathview element.

4.2.2.8.3 State Machine

In order to model the expected behavior of the user interface a state machine has been derived. This state machine keeps track of event changes and state transitions that can be compared against the actual application to ensure functionality is correct.

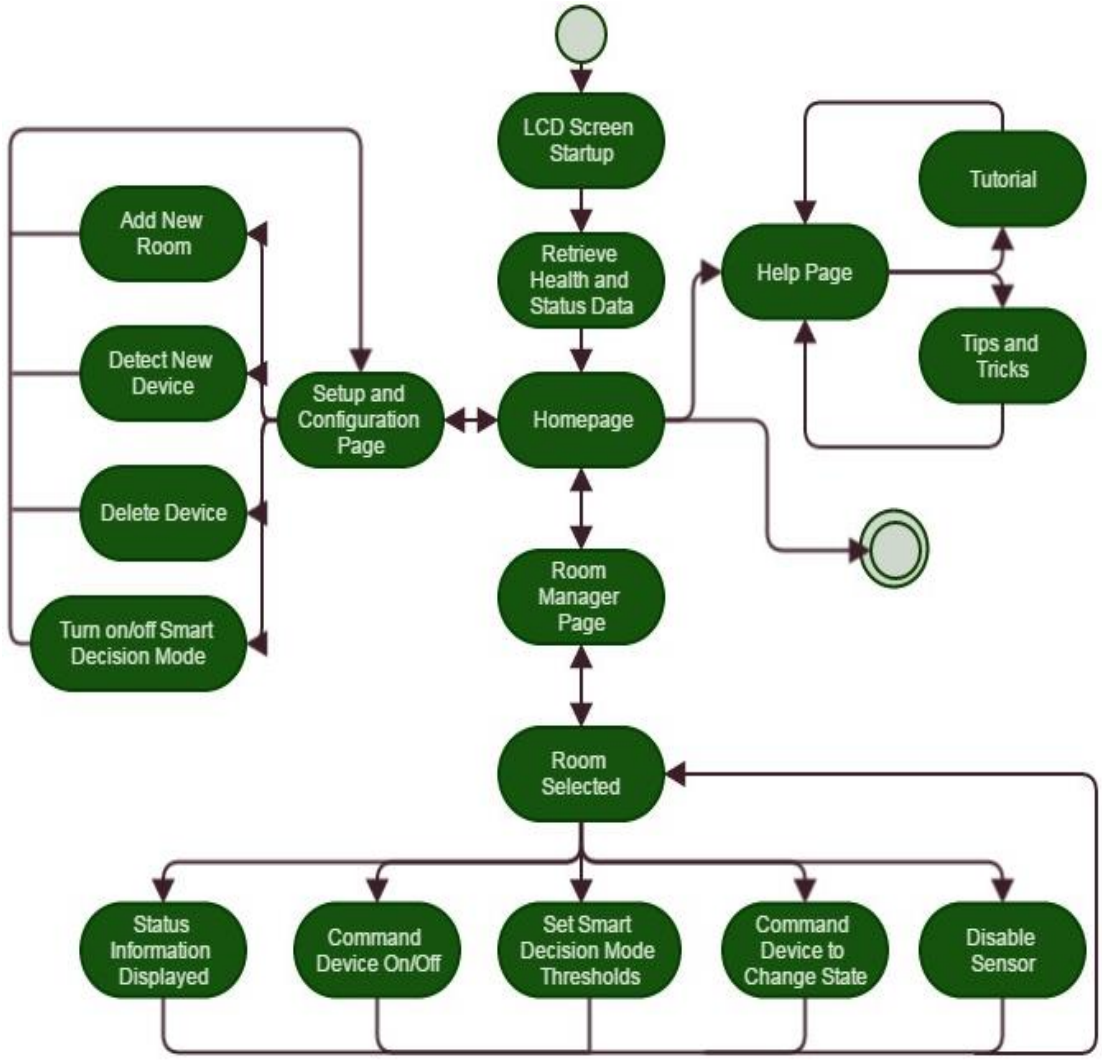


Figure 40: Qt GUI State Machine

4.2.2.9 Smart Switch

The smart switch software is separated into 4 systems. The Data Acquisition System, Powerline Communication Controller, LED controller, and Load Controller. These are done as such since technically these systems are independent of each other aside from passing through data from the main controller. Therefore these systems will be asynchronous to each other as they have their own timing to meet in order to pass data through.

4.2.2.9.1 Data Acquisition System

The Data Acquisition system is in charge of constantly updating the raw data from the sensors on the smart switch. This includes temperature, light, capacitive touch, sound, passive infrared, and humidity. It will constantly get this data at a rate of 100Hz with the exception of sound. Sound will be tied to an interrupt since it can change anywhere from 20Hz to 20 KHz. Therefore to keep the system

running without such a fast refresh rate to save power it will interrupt based. This will only allow us to get sharp spikes in noise like clapping.

All analog sensors come in as a value from 0 to 1023. Capacitive touch comes in as 10 digital values. Therefore in order to pack all this data to send it to the main controller we will need 42 bits. It will need 10 bits for each analog sensor which puts us at 30 bits, 10 more bits for the capacitive touch, 1 bit for whether sound is active or not, and 1 bit for whether the passive infrared had an interrupt based on change.

4.2.2.9.2 Powerline Communication Controller

The next system is the Powerline Communication Controller. Once we have the data needed from the sensors and we want to send it, we need to pack it up in SPI messages to send to the Digital to Analog Converter of the Powerline Communication chip. This is only to be done in an interrupt based manner though. So after the data is processed and an event happens such as that "temperature changed by 2 degrees", or "sound value went above the threshold", or "light sensor changed by this value" then we will pack the data and control the DAC accordingly to send the data.

This has to be its own system because it's not as easy as just sending digital data. We have to control the DAC to act like a sine wave and correctly modulate that sine wave to represent our data. Therefore a translator system needs to be in between the data from the Data Acquisition System and the DAC register from the PLC chip. That's the main focus of the Powerline Communication Controller.

Along with this the Powerline Communication Controller is in charge of interpreting the data received from other systems. This is done through 1 analog pin. Therefore the system needs to be constantly polling this pin and interpreting the 10-bit analog value to represent the data being sent from other systems. This is how the smart switch and load controller get commands from the main controller.

4.2.2.9.3 LED Controller

The LED controller is fairly simple. It will depend on the capacitive touch, passive infrared, and the light data. As a person approaches the switch the light sensor value should be going down and/or the passive infrared should be triggering. As it senses this pattern then you will start controlling the outside LEDs to fade on in order to give the user clear visibility of the switch.

Once the user is in front of the switch and starts touching the capacitive touch pads then the LEDs under the pads will light up accordingly to let the user know that his input is registering correctly. As the user moves to other capacitive touch pads then old ones need to shut off as the new ones turn on. For this reason the capacitive touch LEDs will be interrupt based. This will give the quickest response from the user's touch to the LED lighting.

To control the LEDs a serial message needs to be packed with which LEDs to turn on/off. This serial message is then sent to the qtouch chip by Atmel. First an

address of the LED must be sent. Then the value of brightness needs to be sent right after. Data rate has to be quick in order to fade in and our LEDs in real time of the user physically using the capacitive touch pads.

4.2.2.9.4 Load Controller

The Load Controller will have direct commands from the main controller. The load controller is in charge of controlling the switches and triacs that physically control the items in the room like the light and fan. So it needs to know what switch or triac is responsible for which item. That means that it also has to understand setup messages from the main controller to tell it which of it switches and triacs are hooked up to what.

Along with this is the triac control. To control the triac we need to have an input of the zero crossing of the 120VAC. Along with this we need to have a dedicated timer that understands how to convert a percentage to a time value of when to pulse the triac. So once that interrupt for the zero crossing is received it will wait the appropriate amount of time to pulse the triac to let power through.

5 DESIGN SUMMARY OF HARDWARE AND SOFTWARE

5.1 HARDWARE DESIGN SUMMARY

GARVIS is composed of four main systems. The main controller, the smart switch, the load controller, and the wireless glove. The smart switch and the load controller are both built together but for the sake of understanding how the system works and interacts with each other we have labeled it as a separate system.

The main controller is the brains of the whole system. It's in charge of gathering data, storing it, and making smart decisions. The system is mainly composed of a Beaglebone Black, which has a Cortex A8 processor, with a cape, which is custom made by us, that allows it access to all the peripherals like the Powerline Communication chip, the Bluetooth channel, the HVAC control, the LCD screen, and power.

The main controller will be powered straight through the 120VAC line. It will go through an AC to DC converter designed by us. This will output the 15V needed by the PLC chip, 5V for the Beaglebone, and 3.3V for the logic on the PLC chip. This will also power the LCD screen which requires separate power from the Beaglebone.

The main controller will communicate to all other devices, except the wireless glove, through power line communication. The power line communication is done through a Texas Instrument chip that takes in inputs through SPI to control a digital to analog converter to inject communication into the power lines. Power line communication was used since there are pre-existing wires in every house that connect to all the outlets. This is extremely convenient for such a system where your two options are wireless communication or power line communication.

The main controller will receive data from the wireless glove through Bluetooth. The Bluetooth protocol will be implemented by the use of a Bluetooth Bee on a UartSBee shield that allows to talk to the BT Bee through UART. The main controller will only listen to the glove though. It has no information to give back to the glove.

The main controller will take care of HVAC control. On the cape there will be a few MOSFETs that will allow us to control the connections of the wires for HVAC. This system is very easy to control since the goal is to either short one wire to the other or leave them open.

The main controller will be in charge of controlling the LCD touchscreen. The Beaglebone will connect straight to the LCD through the HDMI connector. It will get the touchscreen data through USB.

The main controller will communicate with all the devices on powerline communication lines as a master. It will request or send data from all the devices

first to initiate communication. It will constantly query all the smart switches to get their sensor data and keep it up to date. From the data of the switches and the LCD touchscreen it will send messages to control the Load Control.

The smart switch will be composed of an MSP430FR5739, a TMP36 analog temperature sensor, a HIH-4030 analog humidity sensor, a passive infrared module, a microphone, ambient light sensor, LEDs, and a capacitive touch PCB designed by us. The smart switch is mainly responsible for just collecting this data and sending it to the main controller as it requests it.

The load control is responsible for controlling the switches and triacs used to control the lights and fans in a room. The triacs are used in order to provide variable control to dim lights, or slow down fans. This both helps provide this effect along with reducing power consumption of the house.

The wireless glove is composed of an ATMEGA328 microcontroller, an accelerometer, a gyroscope, a magnetometer, and flex sensors. These sensors are used to get the kinesthetic data of the hand and fingers and then send it through Bluetooth to the main controller. This is here in order to add gesture control to the home automation system.

5.2 Software Design Summary

GARVIS requires the use of multiple software entities to implement all parts of the gesture detection and recognition along with the home automation commanding and controlling. The following table outlines all of the software sub-systems.

Sub-System	Software	Means of Implementation
Home Automation System	Central Manager Backend	Embedded C++ Code
	Central Manager Server	Lighttpd Server
	Central Manager GUI	Qt Application
	Database Management System	MySQL
	Smart Switch	Embedded C++ Code
Gesture Control and Recognition	Glove Control Software	Embedded C++ Code
	Glove Manager API	C++ Library

Table 11: Software Sub-Systems

5.2.1 Home Automation System

The home automation part of the system will handle all commanding and statusing of the household devices. It has a backend in charge of conducting adaptive processing via pattern recognition and regression analysis as well as dealing with database management. The backend will query the relational database stored on the Lighttpd server to extract the necessary data for the adaptive commanding. The backend will also insert information at the proper time intervals for future adaptive commanding.

The Central Manager will receive input from the glove, the Central Manager GUI, and from Smart Switches around the home. The Smart Switches are tasked with monitoring sensors and commanding devices when triggered by the Central Manager. Both the glove and the Central Manager GUI are direct user input interfaces that will allow for the user to manually mode and configure the home automation system.

5.2.2 Gesture Control and Recognition

The glove will assist with the gesture recognition in the system where it will take in raw sensor data and send it via Bluetooth to the Glove Manager API for processing and gesture recognition. The Glove Manager will translate the raw data from the sensors into 3-axis acceleration, 3-axis radial acceleration, 3-axis relative magnetic field, and flex sensor data from 3 fingers. The Flex sensor data will be used to know the position of the fingers. The accelerometer and gyroscope data will be used to tell in what direction you are moving or rotating. The magnetometer data will be used to tell where you are facing relative to the North.

The hardest part is correctly recognizing a gesture from this data. For example let's say you have your hand palm side up with straight fingers, and then you move your hand in an upward position like if you were lifting something. So all flex sensor data should be close to zero while accelerometer data should be increasing in the -Z direction (since the accelerometer is upside down). You also have to be doing this for a specific amount of time for it to register so you need to have a timer to time for how long the data of the flex sensors and accelerometer is in this state. Along with this you need to make sure that the gyroscope data is reading zero since you are not rotating.

6 PROJECT PROTOTYPE CONSTRUCTION AND CODING

6.1 PART SELECTION AND ACQUISITION

When picking parts we kept used a simple process; we went to Digikey, created a search with our desired part and features and found a cheap one that met our specs. We also tried our best to go with larger manufactures because they tend to be more willing to send samples of their parts to help save us money. Texas Instruments is a preferred manufacturer because they send samples, have a lot of reference designs and usually have very good documentation. With parts acquisitions, we primarily used Digikey, simply because they have a large selection, an easy to use website, fast shipment, and great customer service.

6.2 PCB VENDOR AND ASSEMBLY

We plan on using two different PCB fabricators. The first being OSH Park (stands for open source hardware) which is infamous for their cheap 2 layer purple PCBs. We have had a good experience with them before and found most of their manufacturing satisfactory and quick. We have found from experience that the typical turnaround time is 2-3 weeks. At $5.00/sq in, the price is hard to beat and we will use them for most of our test boards and non-critical PCBs. Even though the boards we have received before are satisfactory, you get what you pay for, and their constraints are limited.

The other manufacture is Advanced Circuits, known for their 4PCB service. We have not had experience with them, but they have a good reputation among the engineering community. 4PCB has a student deal where students can get a 60 sq. in, 2 layer board for $33 or a 4 layer board for $66. They claim their boards ship in 5 days. We will use 4PCB for our more advanced, and larger boards.

6.3 FINAL CODING PLAN

Due to the nature of this project, the code sub-systems will need to be developed in an order that correlates with the completion of the hardware entities that the code will be ran on. To test full code functionality, there is a need for fully functional hardware so a prototype and hardware modeling are necessary to develop code before hardware is developed.

Since the Central Manager will be housed on an off the shelf part, the Beaglebone Black, development of code for the Central Manager Backend, server, and GUI has already began. To develop the I/O Manager fake data can be fed through the GPIO pins to simulate the PLC converter sending messages. The development of adaptive algorithms, database queries, and the GUI is also independent of any additional hardware (other than the Beaglebone) so these are highest priorities in the software development process.

For the glove, the Glove Manager API can also be developed independently of hardware development on this project since it is ran on either a PC or the Beaglebone. The receiving of data will also have to be simulated by setting up a

microcontroller with a Bluetooth shield to mimic the glove until the actual glove hardware is developed. The glove's gesture sensing firmware is not very complex so it won't require a large time to develop and it depends on the hardware being fully developed so it is a lower priority.

The Smart Switch software is also relatively simple and depends on hardware development to fully integrate so it is a lower priority as well until the hardware for it is completed.

For the software development life cycle, the V Software Process Model will be utilized because it puts the main focus on allowing the developers to be able to test and verify code for correctness. The V Model is more flexible and will allow for the reaching of full capabilities instead of providing restrictions once the team is too far into the testing phase. Even while the system is in the testing phase the V Model allows the designers to go back and re-examine requirements and the design techniques if necessary. The following figure provides a graphical representation of this model.

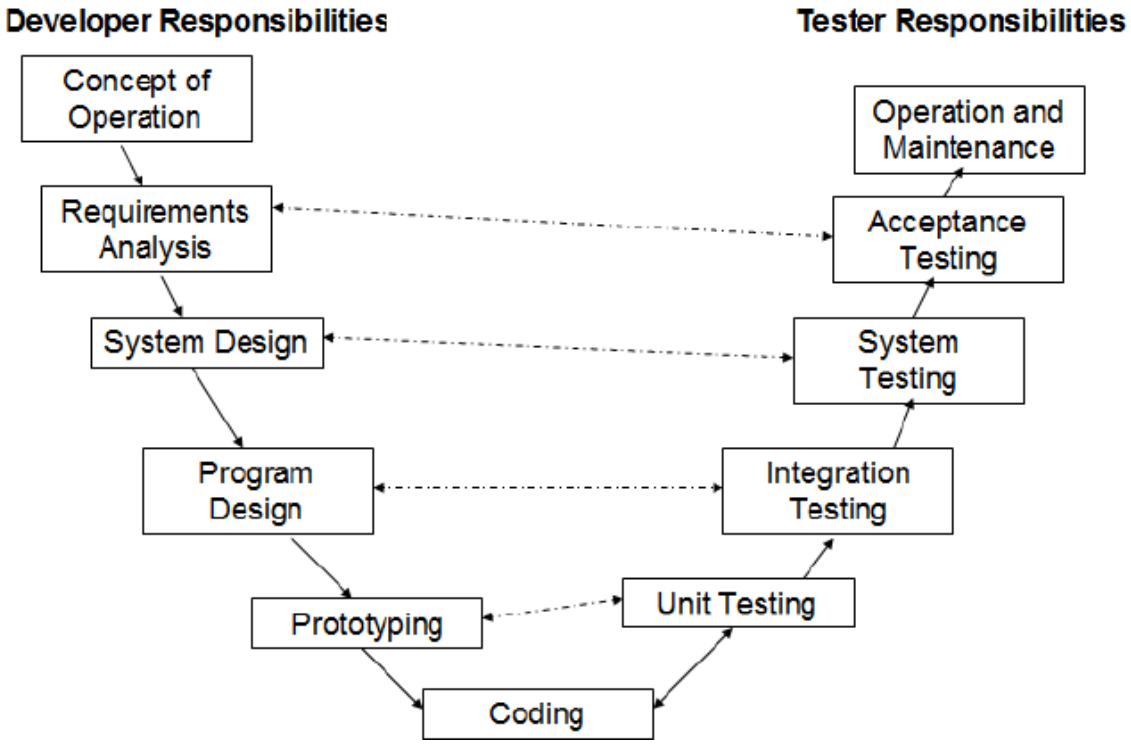


Figure 41: V Software Lifecycle Model

7 PROJECT PROTOTYPE TESTING

7.1 HARDWARE TESTING

There are seven primary types of testing that need to be done to be reasonably sure that a large scale implementation of the household control system is possible. Each one is detailed below, including the specific test that need to be completed for each piece of hardware that is prototyped. For each category and piece of hardware, a list of requirements is given.

7.1.1 Functionality Testing

First and foremost, the primary function of each piece of home automation hardware must be able to perform its task on its own.

7.1.1.1 Main Controller

The main controller must power on, present the user with an application based graphic user touch-based interface, and be able to control the household HVAC system.

7.1.1.2 Smart Switch

The smart switch has the functions which need to be verified of proper operation, those functions are as follows:

- Proximity Detection
- Capacitive touch
- LED Lighting- both edge lit RGB and backfire LEDs
- Ambient light sense
- Temperature sense
- Microphone noise detection
- Humidity sensing
- Occupancy

There are a battery of test to ensure proper calibration and accurate data, however for the scope of this project we are looking for functionality.

7.1.1.2.1 Proximity Detection

To test the proximity detection, we will have a firmware build that will light the backlight of the switch whenever the microcontroller sees the interrupt from the touch controller and the proximity sensor channel has changed. To test range, we will do a couple of test at inch increments to see where the detection falls off.

7.1.1.2.2 Capacitive Touch

To test the capacitive touch sensitivity and levels, we will have a custom firmware load that will read from the touch controller whenever it receives an interrupt indicating that the status of the sensors has changed and will print to a terminal which sensor has changed. We can correspond this with the ones we are touching to ensure proper operation.

7.1.1.2.3 LED Lighting

We will have a firmware load for the microcontroller that will be able to light each LED individually, and all the LEDs together. We will do a couple of sequences of patterns to ensure proper operation. For the RGB LEDs, we send each LED a signal to turn on red, green, and blue separately, then together, then mix all the colors.

7.1.1.2.4 Ambient Light Detection

To test the ambient light sensor, once it has power, we can read the output with a multimeter or oscilloscope while changing the light level with a flash light. Once we verify proper operation of the sensor itself, we will verify our reading. Since the sensor outputs an analog value, we will have to read it with an ADC on the microcontroller. To verify proper reading, we should see a change in value in the register when our hand is covering the sensor and when a flashlight is on it.

7.1.1.2.5 Temperature Sense

Testing the temperature sensor will consist of heating up the entire sensor with a common blow dryer and watch the temperature increase accordingly. In addition, we could hold a compressed air can (common for keyboard cleaning) upside down and spray the liquid on the temperature sensor and observe the decrease in temperature via a serial port into the microcontroller.

7.1.1.2.6 Microphone Noise Detection

Since we are not recording audio, just looking for audio noise patterns such as claps, our testing is easier. We will be sampling the audio with an ADC and will determine if there is an elevation in the standard level. To test this will read the output of the microphone amplifier with an ADC and note whether or not the levels increase with increasing noise. We can increase audio noise with clapping, yelling, screaming, playing music etc. We will test our algorithm for determining which frequencies we want with an audio frequency sweep. Our algorithm should only trip on our specified frequencies.

7.1.1.2.7 Humidity Detection

We will change the humidity of the air around the sensor by injecting steam near the sensor with a hot iron. We should observe a change in the value of the sensor.

7.1.1.2.8 Occupancy Detection

Occupancy detection is done with the passive infrared sensor (PIR). The PIR sensor we are using can detect up to 12m (≈40ft). Since the output will be digital, we can perform movement at the edge of detection at foot increments and verify there is a digital change, which verifies detection.

7.1.1.3 Load Controller

The load controller must be able to control an incandescent light bulb in a gradient. The load controller must be able to control a fluorescent light bulb. The load controller must be able to take a current measurement and calculate average power of a light fixture.

7.1.1.4 Airflow Controller
The airflow controller must be able to open and close an airflow damper. The airflow controller must be able to take a temperature measurement of the airflow.

7.1.1.5 Glove
The glove must be able to gather IMU data and send this data to an external processing device.

7.1.2 Power Supply Testing
Many of the home automation hardware designs include power requirements, some from household main electricity. The proper function of the power supply circuitry is fundamental to the functionality of the system. All of the specified voltages and currents are to be measured with a multimeter during regular hardware operation.

7.1.2.1 Main Controller
The PLC card amplifiers must receive 15V and can be supplied up to 2A without significant voltage drop. The PLC card logic must be supplied 5V. The Beaglebone main power supply must be supplied 5V and be capable of providing up to 500mA. The LCD screen must be supplied 9V and be capable of providing 1A. The ground plane must be checked for signal interference.

7.1.2.2 Smart Switch
The PLC card amplifiers must receive 15V and can be supplied up to 2A without significant voltage drop. The MSP430 must be supplied 5V. The LCD driver must be supplied 5V with enough current to light all LEDs simultaneously.

7.1.2.3 Load Controller
The PLC card amplifiers must receive 15V and can be supplied up to 2A without significant voltage drop. The MSP430 must be provided 5V.

7.1.2.4 Airflow Controller
The PLC card amplifiers must receive 15V and can be supplied up to 2A without significant voltage drop. The stepper motor supply must be supplied 9V and be capable of providing up to 1A. The MSP430 must be provided 5V.

7.1.2.5 Glove
The glove battery must supply required voltage to the microcontroller and IMUs.

7.1.3 Communication Testing
After the individual functionality and power requirements have been verified, the system functionality can be initially checked by sending a simple serial data between all of the devices to ensure that the communication methods are working. The data sent is to be a single character, and the repeatability needs to be very high.

7.1.3.1 Main Controller
The main controller must be able to communicate with the smart switch, load controller and airflow controller using power line communication. The main

controller must be able to connect to the household local network. The main controller must be able to connect to the glove using Bluetooth. Determine the conditions that would cause a power line communication signal to be lost or corrupted badly, including distance traveled.

7.1.3.2 Smart Switch
The smart switch must be able to communicate with the main controller, load controller and airflow controller using power line communication.

7.1.3.3 Load Controller
The load controller must be able to communicate with the main controller, smart switch and load controller using power line communication.

7.1.3.4 Airflow Controller
The airflow controller must be able to communicate with the main controller, smart switch and load controller using power line communication.

7.1.3.5 Glove
The glove must be able to communicate with a desktop pc and with the main controller using Bluetooth.

7.1.4 User interface Testing
Not all of the home automation hardware has a user interface, but the ones that do directly interact with the home owner must verify that the interaction happens as intended. Many of these requirements are more subjective than the other section because the ease of use cannot be measured numerically.

7.1.4.1 Main Controller
The touch based user interface should allow changes to household temperature in under 10 seconds. The user should be able to view current household environmental data. The user can view energy use trends graphically.

7.1.4.2 Smart Switch
The user can turn the lights on and off quickly and easily, and dim lights intuitively. The swipe interface has visual feedback using LED lighting. The user is able to set a desired room temperature. The user can turn lights on and off by clapping.

7.1.4.3 Glove
The glove is capable of recognizing user gestures.

7.1.5 Error management Testing
Once the basic functionality of the system is determined, the robustness of the system is to be evaluated. The ability to handle error situations is critical for a successful system. Note that the components are only responsible to be able to reliable detect errors, and are not responsible to address or fix errors. Communication errors refer to situations in which section 7.1.4 is not met.

7.1.5.1 Main Controller
The main controller should be able to detect a communication error in which a message is sent and no reply is received. The main controller should be able to detect unexpected brownouts or power loss. The main controller should be able to detect network connectivity loss. The main controller should be able to interpret error messages from the smart switch, load controller and airflow controller.

7.1.5.2 Smart Switch
The smart switch must be able to detect communication errors. The smart switch needs to be able to detect temperature or humidity errors in which the value is too high. The smart switch must be able to detect over temperature or under temperature errors in which the HVAC desired room or household temperature is not being met.

7.1.5.3 Load Controller
The load controller must be able to detect communication errors.

7.1.5.4 Airflow Controller
The airflow controller must be able to detect air damper movement errors, in which a movement is commanded but not executed. The airflow controller must be able to detect communication errors. The airflow controller must be able to detect over temperature or under temperature errors in which the HVAC desired temperature is not being met.

7.1.6 Sensor Accuracy Testing
Various components have analog sensors to interpret environmental data. The returned data is to be compared to measured environmental data to determine the accuracy of the sensors.

7.1.6.1 Smart Switch
Compare the accuracy of measured temperature and humidity to actual temperature and humidity.

7.1.6.2 Load Controller
Compare the measured current to the actual current.

7.1.6.3 Airflow Controller
Compare the actual temperature to the measured temperature.

7.1.7 Integration Testing
Testing the system as a whole is not plausible due to the cost of purchasing more than one or two of each piece of home automation hardware. However, there are several autonomous actions that can be tested. Anytime that the room temperature is to be adjusted, it’s possible that the movement of the vent damper will be a signal for the tester to manually close the room vent, due to the cost of installing the duct based air damper. Since it’s unclear at this point if and where the system could be installed onto a HVAC system, the HVAC control may have to be done manually during integration testing.

Determine if an increase in temperature of a room compared to the user set room temperature will prompt the airflow controller to open and allow cool air to flow. Also determine if the opposite situation works.

Determine if the main controller is capable of turning off lights after a user leaves the room. Also determine if the household controller is capable of adjusting household temperature when there is no sensory input to the smart switch for a given period of time.

Determine if the user is able to adjust the room temperature using the smart switch. Determine if the

7.2 SOFTWARE TESTING

7.2.1 Environment

Software testing can occur in many different phases but the prototype testing phase will require the availability of certain hardware in order to completely test all entities for functionality. Most of the software tests require at a minimum hardware including the Beaglebone processor board with the LCD touchscreen. Some of the more complex software tests will additionally need Smart Switches to be available to add devices to the system or the use of imitation hardware to mimic how devices will interact with the software. Another form of imitation will be test programs that can feed realistic data into the software system for a complete functionality analysis. Each of the following test procedures will outline the components to create the software test environment for the procedure.

7.2.2 Software Specific Testing

7.2.2.1 Database Testing

In order to test that the database DDL was implemented correctly and that the DML mechanism being used will work for adding, modifying, or deleting data from the database the following procedure must be utilized. A program to run an exhaustive query test must be created that will call the query methods in Central Manager. This program will populate the database with imitation data, query the database for certain parts of that data, and delete sections of the data.

Procedure:

1. Power on the Central Manager and make sure it is attached to an output display
2. Get a Linux terminal running
3. Save the test program to the Beaglebone
4. Run the test program from the terminal
5. Examine the output file from the test program to ensure that the expected output correlates with the actual output

7.2.2.2 Central Manager GUI Testing

The Central Manager GUI must go through testing to confirm its durability, ease of understanding, and functionality.

7.2.2.2.1 Durability

Durability testing will be conducted to make sure that the GUI can handle expected user input, unexpected user input, and a high workload. The following procedure will focus on testing the GUI's durability.

Procedure:

1. Power on the Central Manager and make sure it is attached to the touchscreen LCD display
2. Touch the screen and attempt to swipe the Pathview navigation element to ensure that the touch capability is fully functional
3. If the touch capability is fully functional, continue with the procedure. If not, refer to the LCD touch screen test procedure to determine the problem.
4. Navigate to the Room Manager page from the Homepage
5. Select a room from the room listing
6. Vigorously tap on the enable and disable of a device
7. Vigorously enter data into the command field
8. Quickly navigate to the Configuration and Setup page
9. Vigorously tap on any buttons on the Configuration and Setup page
10. Confirm that the GUI never crashed or froze and that warning messages appeared after too many attempts at device control and setup control

7.2.2.2.2 Ease of Understanding

A very important aspect of any user interface is how easily the targeted user can understand and use the application. To aid in understanding of the Central Manager GUI a tutorial along with a tips and tricks page will be available on the GUI. For this system the targeted user is a person above the age of 16 that is capable of using a tablet or personal computer. In order to test the ease of understanding of the GUI a few testers that meet the requirements of being a user will be necessary to conduct the following procedure.

1. Power on the Central Manager and make sure it is attached to the touchscreen LCD display
2. Add a new device to the system by plugging it into a Smart Switch
3. Ask the user to navigate to the help page and read the tutorial and tips and tricks pages. Note if they are able to accomplish this task.
4. Prompt the user to add a room to the system
    a. Note if they navigate to the Configuration and Setup page
    b. Note if they select the Add New Room button, type in a room name, and hit submit
5. Prompt the user to detect for new devices and add it to the newly created room
    a. Note if they navigate to the Configuration and Setup page
    b. Note if they select Detect New Device
    c. Note if they confirm the newly detected device and add it to the room
6. Prompt the user to turn on Smart Decision Mode for their new room

    a. Note if they navigate to the Room Manager page
    b. Note if they select their newly created room
    c. Note if they are able to toggle the button to enable Smart Decision Mode
7. Conduct these tests for at least five different users and note any reactions or questions they have and if they are able to complete the tasks

7.2.2.2.3 Functionality

Functionality is a fundamental requirement that must be tested to confirm that the GUI acts in the way it was designed to act. Through exhaustive testing and the use of the following procedure we can check functionality.

Procedure:

1. Power on the Central Manager and make sure it is attached to the touchscreen LCD display
2. Add a new device to the system by plugging it into a Smart Switch
3. Navigate to the help page and go through every tutorial page and the tips and tricks section. Note if this is completely functional as expected.
4. Add a room to the system
    a. Navigate to the Configuration and Setup page
    b. Select the Add New Room button
    c. Type in a room name
    d. Hit submit and confirm that a new room is created
5. Attempt to detect for new devices and add the new device to the newly created room
    a. Navigate to the Configuration and Setup page
    b. Select Detect New Device
    c. Confirm the newly detected device
    d. Add it to the room
    e. Confirm that the new device exists in the new room to test functionality of this task
6. Turn on Smart Decision Mode for the new room
    a. Navigate to the Room Manager page
    b. Select their newly created room
    c. Toggle the button to enable Smart Decision Mode
7. Check the current status of the new device
    a. Navigate to the Room Manager page
    b. Select their newly created room
    c. Note that the new device status information is being displayed correctly for the room
8. Delete the device
    a. Navigate to the Configuration and Setup page
    b. Select Delete Device
    c. Select the device that was added
    d. Hit the Delete Device button
    e. Hit confirm on the popup that asks if you are sure you want to delete it

f. Check that the device no longer exists for the room to test functionality for the deleting of a device

7.2.2.3 Adaptive Control Algorithm Testing

In order to test that the adaptive control algorithms were implemented correctly the following procedure must be utilized. Two programs must be used to conduct this testing. A program to add imitation data to the database must be used to populate the database and provide a dataset for running the adaptation algorithms on. Another program must be used to mode the Central Manager into Smart Decision Mode and to check what commands the Central Manager comes up with based on the input dataset.

Procedure:

1. Power on the Central Manager and make sure it is attached to an output display
2. Get a Linux terminal running
3. Save the test programs to the Beaglebone
4. Run the first program to populate the database
5. Run the second program to mode the Central Manager and create an output file of the adaptive commands
6. Examine the output file from the test program to ensure that the expected output correlates with the actual output

7.2.2.4 User Commanding Testing

The user has the capability to manually command the system and devices in the system from multiple interfaces. This testing will confirm that the user is able to manually command the system via the GUI. The glove commanding will be tested in the Glove Manager API testing section. In order to ensure that the user is able to manually command the system the next procedure must be followed.

Procedure:

NOTE: Prior to the starting of this procedure, make sure that at least two devices are attached to the system, one on/off device and one threshold device.

1. Power on the Central Manager and make sure it is attached to the touchscreen LCD display
2. Navigate to the Room Manager page
3. Select the room where either of your devices are in
4. Toggle the first device to the on state and confirm that it is physically turned on
5. Toggle the first device to the off state and confirm that it is physically off
6. Navigate to the second device and enter a threshold to command it to
7. Select submit and confirm that the second device is moded to the threshold it was given

7.2.2.5 Status Display Testing

Another functionality that must be tested is the Central Manager's ability to send the correct status data to the GUI for representation that the user can understand. The following procedure will test that the desired status data is displayed correctly for the user.

Procedure:

NOTE: Prior to the starting of this procedure, make sure that one Smart Switch is attached to the system.

1. Power on the Central Manager and make sure it is attached to the touchscreen LCD display
2. Once the Central Manager GUI is on the homepage, check that the current average household temperature is displayed and is updating as necessary. Depending on the conditions being tested in, this value is expected to be around 70-80 degrees Fahrenheit.
3. Using a heat source such as a blow dryer, give indirect heat to the Smart Switch temperature sensor and acknowledge if the Central Manager GUI temperature increases on the homepage.
4. Navigate to the Configuration and Setup page on the Central Manager GUI
5. Select the Add New Room button
6. Type in a room name
7. Hit submit and confirm that a new room is created
8. Select Detect New Device
9. Confirm the newly detected device, the Smart Switch
10. Add it to the room
11. Navigate to the Room Manager page
12. Select the newly created room
13. Confirm that the Smart Switch is providing valid sensor data from the room that it is located in in terms of temperature, humidity, motion detection, and sound data.

7.2.2.6 I/O Manager Testing

The I/O Manager deals with all interaction of the Backend Central Manager software with the GUI, glove, PLC interface, and Ethernet interface. The interactions between the I/O manager and these interfaces have been tested in previous procedures so for the sake of avoiding redundancy an additional procedure will not be included to retest this interface.

7.2.2.7 Glove Manager API and Firmware Testing

The Glove Manager API will be tested by feeding the glove manager API with imitation glove data through Bluetooth. Because the Glove Manager will be housed on a processor it can be tested on any laptop or desktop computer. The procedure for testing the Glove Manager API's functionality is as follows.

Procedure:

1. Open an application that can utilize the Glove Manager API

2. Using the glove or a glove prototype, setup up Bluetooth communication with the Glove Manager's host computer
3. Move the glove up and down and examine that the mouse is acting in the same pattern as the glove is being moved
4. Move the glove left to right and examine that the mouse is acting in the same pattern as the glove being moved
5. Using a shooting like motion with your pointer finger of the glove pointing to the screen and the thumb moving downward toward the pointer finger, examine that a click is registered on the Glove Manager API computer

8 ADMINISTRATIVE CONTENT

8.1 MILESTONE DISCUSSION

The development of milestone for the GARVIS project progressed as the team determined the minimum requirements and the stretch requirements for GARVIS. Due to the nature of the project, there were a lot of new technologies, protocols, and methodologies that required extensive research so they were given longer timelines in our milestone chart. The two largest aspects that were considered in the creation of our milestones dealing with time and team member capability.

8.1.1 Scheduling

GARVIS is a complex system with many subsystems that all require a good amount of time to design and implement. Being under the strict time restriction of two semesters to finish the project all milestones had to be very carefully scheduled and prioritized. For example, components that have heavy software needs were given first priority in the hardware development phase. This is because there needed to be enough time to allow for full software development and integration of software and hardware.

8.1.2 Team Composition

The second aspect to consider to get optimal results for our project is assigning each team member with responsibilities they had the skill set and time for. The following list is a breakdown of the team's main responsibilities.

Joshua Illes
- Power Line Communication
- PCB Design
- Smart Switch Hardware
- Manufacturing

Andres Mujica
- Glove Hardware and Software
- Smart Switch Hardware and Firmware
- Digital Communication

Jackson Schleich
- Vent Control Hardware and Firmware
- Main controller Beaglebone Cape PCB
- Adaptive Algorithms

Sarah Strauss
- Central Manager Embedded Software
- User Interfaces
- Database
- Glove Application Programming Interface

8.1.3 Milestone Chart

By merging the time and team composition considerations together, we were able to obtain a milestone chart. This chart shows the amount of time each

subsystem is project to take, their priority, and who is responsible for ensuring they reach completion.

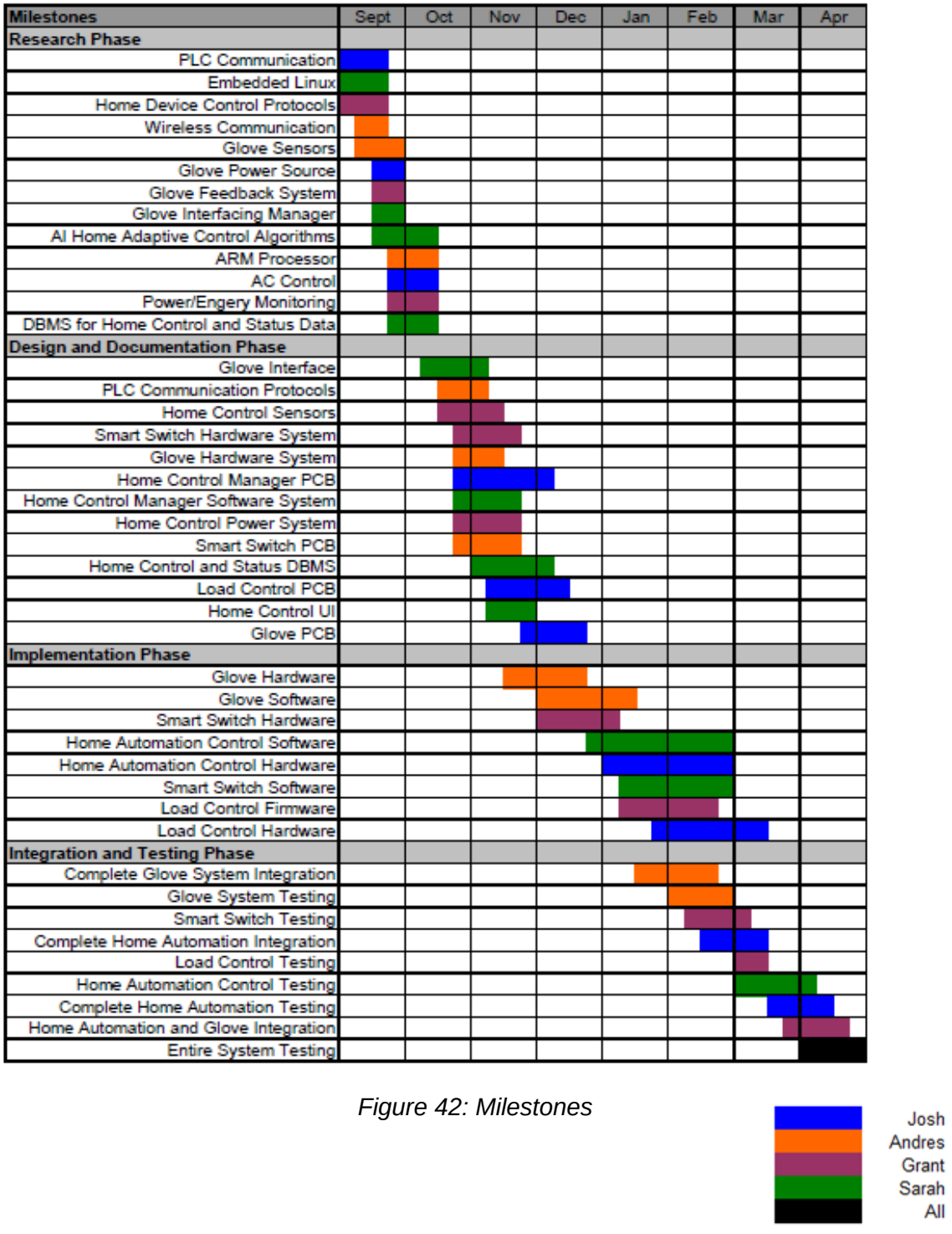


Figure 42: Milestones

8.2 Budget and Finance Discussion

GARVIS is a large system that contains many relatively costly parts. In order to make the project financially feasible the team had to secure funding from various sources. A few groups have been able to help with some of the costs and we very greatly appreciate it. From the help of SoarTech, Leidos, and contributions from relatives we have been able to secure $2200. We were able to gain support from SoarTech due to our project's artificial and adaptive intelligence aspects. They are also providing us with advice from our starting up and through development.

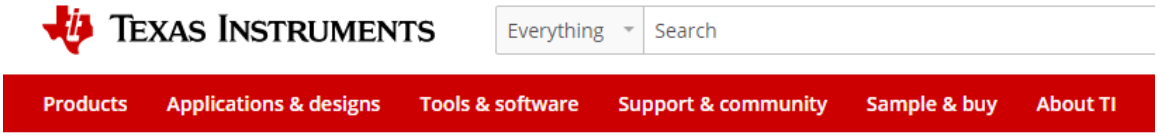
Originally, it was expected that GARVIS would have more funding but due to a sponsor drop out the team has had to be more careful with our finances. Some design decisions have been made based off of this including the move to use an off the shelf processor board as compared to an originally designed PCB that would cost around $500 for the single board. This decision was made because it would save money but not limit any of the features of GARVIS in the process.

The following bill of materials will provide for a clearer outline of GARVIS' total cost.

BOM

BOM

9 APPENDIX A – COPYRIGHT PERMISSION



TI Home > Copyright

Copyrights

Texas Instruments is pleased to provide the information on these pages of the World Wide Web. We encourage you to read and use this information in developing new products.

TI grants permission to download, print copies, store downloaded files on a computer and reference this information in your documents only for your personal and non-commercial use. But remember, TI retains its copyright in all of this information. This means that you may not further display, reproduce, or distribute this information without permission from Texas Instruments. This also means you may not, without our permission, "mirror" this information on your own server, or modify or re-use this information on another system.

TI further grants permission to non-profit, educational institutions (specifically K-12, universities and community colleges) to download, reproduce, display and distribute the information on these pages solely for use in the classroom. This permission is conditioned on not modifying the information, retaining all copyright notices and including on all reproduced information the following credit line: "Courtesy of Texas Instruments". Please send us a note describing your use of this information under the permission granted in this paragraph. Send the note and describe the use according to the request for permission explained below.

Permission to use *Figure 15: LSM303DLHC Standard Setup* from STMicroelectronics (permission requested), *Figure 16: L3GD20 , and Figure 17 L3GD20 electrical connections and external component values* which are figures from STMicroelectronics

From: **Andrès Mujica** (mujicad93@hotmail.com)
Sent: Wed 12/03/14 9:41 AM
To: ame_bo_west_inquiries@list.st.com (ame_bo_west_inquiries@list.st.com)
Cc: Sarah Strauss (s.strauss8@gmail.com)

STMicrolectronics,

I am an Electrical Engineering student at the University of Central Florida. I am requesting permission to include the following figures in my capstone design paper as references.

Figure 4 (LSM303DLHC electrical connection) from the LSM303DLHC datasheet
Figure 2 (pin connection) from the L3GD20 datasheet
Figure 5 (L3GD20 electrical connections and external component values) from the L3GD20 datasheet

I will site them correctly to give STMicroelectronics proper credit. Can I get permission to use them?

Thanks,
Andres Mujica

Permission to use Image

Jackson Schleich <jacksonschleich@gmail.com> Tue, Dec 2, 2014 at 9:09 PM
To: info@suncourt.com

Suncourt,

I'm an engineering student seeking permission to use an image of your 6" normally closed air damper in a research paper on home automation. I've purchased on of the air dampers for prototyping purposes. I've attached the image that I'd like to use, please let me know if you approve of the use of this image.

Thanks,
Jackson Schleich

University of Central Florida
BSEE

Contact Us

To contact us, simply select the reason for your correspondence from the pull-down list below. Please fill in your name and e-mail address, along with the text of your note. Your information will be sent to the appropriate person(s) immediately.

Thanks!

Reason:	Ask for Reprint Permission ▼
Your Name:	Joshua Illes
Your E-mail:	joshua.t.illes@gmail.com
Message:	Hello, I am a UCF Senior Electrical Engineering Student. I was looking to gain permission to reprint a figure from your "How Dimmer Switches Work" article. I will give you full credit.
(400 character limit)	

Request for reprint from "How Stuff Works" of Figure 11

10 APPENDIX B - REFERENCES

[1] Honeywell International Inc., "Honeywell Home Automation," Honeywell, 2012. [Online]. Available: http://homesecurity.honeywell.com/home_automation.html. [Accessed 20 10 2014].

[2] Lutron Electronics Company, "Lutron," Lutron Electronics Company, 2014. [Online]. Available: http://www.lutron.com/en-US/Service-Support/Pages/Service/Overview.aspx. [Accessed 20 10 2014].

[3] "X10 Basics," X10 Home Gadgets, 2014. [Online]. Available: http://www.x10.com/x10-basics.html. [Accessed 20 10 2014].

[4] J. Rowberg, "Keyglove," 2012. [Online]. Available: http://www.keyglove.net/. [Accessed 20 10 2012].

[5] B. Troili, L. Rubio-Perez, A. Mizan and K. Chan, "High Six," 2014. [Online]. Available: http://www.eecs.ucf.edu/seniordesign/fa2013sp2014/g06/documents.html. [Accessed 20 10 2014].

[6] Sparkfun Electronics, "Serial Communication," [Online]. Available: https://learn.sparkfun.com/tutorials/serial-communication#res. [Accessed 10 2014].

[7] B. Baraboi, "Narrowband Powerline Communication--Applications and Challenges—Part I," 3 March 2013. [Online]. Available: http://www.edn.com/design/wireless-networking/4408140/Narrowband-Powerline-Communication-Applications-and-Challenges-Part-I.

[8] K. Dostert, "Telecommunications of the Power Distribution Grid- Possibilities and Limitations," Univeristy of Karlsruhe, Karlsruhe, 1997.

[9] cvolpato, "X-10 Transmission Theory," [Online]. Available: http://digilander.libero.it/cvolpato/elettronica/X-10_Transmission_Theory.htm.

[10] G. Evans, CEBUS Demystified: The ANSI/EIA 600 User's Guide, McGraw Hill, 2001.

[11] Circuit Design Inc, "Digital Modulation: Frequency Shift Keying (FSK, MSK)," [Online]. Available: http://www.cdt21.com/resources/Modulation/modulation_FSK.asp.

[1 A. Bookout, "How do modems work?," 14 May 2000. [Online]. Available:
2] http://www.udel.edu/physics/scen103/XDGS/connection/connection.htm.

[1 Cypress Semiconductor, "What is Power Line Communication?," 11 August
3] 2011. [Online]. Available:
http://www.eetimes.com/document.asp?doc_id=1279014.

[1 L. T. Berger, A. Schwager, P. Pagani and D. M. Schneider, MIMO Power
4] Line Communications, Boca Raton: CRC Press, 2014.

[1 Galco, "How Relays Work," [Online]. Available:
5] http://www.galco.com/comp/prod/relay.htm.

[1 T. Harris, "How Dimmer Switches Work," How Stuff Works, 13 August 2002.
6] [Online]. Available: http://home.howstuffworks.com/dimmer-switch.htm.
[Accessed 3 December 2014].

[1 J. L. H. David A. Patterson, Computer Architecture : A Quantitative
7] Approach, Burlington: Morgan Kaufmann, 2011.

[1 ARM Ltd., "Processor Index," ARM, 2014. [Online]. Available:
8] http://www.arm.com/products/processors/index.php. [Accessed October
2014].

[1 ADTT, "Micro-SD specifications," 2006. [Online]. Available:
9] http://www.dtt8.com/images/micro-sd%20specification.pdf. [Accessed 22
October 2014].

[2 Texas Instruments, "Power Management Guide," 2014. [Online]. Available:
0] http://www.ti.com/lit/sg/slvt145n/slvt145n.pdf. [Accessed 10 October 2014].

[2 Linear Technologies, "Circuit Techniques for Clock source," October 1985.
1] [Online]. Available: http://cds.linear.com/docs/en/application-note/an12fa.pdf.
[Accessed 05 November 2014].

[2 Canonical Ltd. Ubuntu, "Ubuntu ARM," 2014. [Online]. Available:
2] http://www.ubuntu.com/download/server/arm. [Accessed 1 11 2014].

[2 Arch Linux ARM, "Arch Linux ARM," 2014. [Online]. Available:
3] http://archlinuxarm.org/. [Accessed 1 11 2014].

[2 Debian, "Debian ARM," 30 4 2014. [Online]. Available:
4] https://www.debian.org/ports/arm/. [Accessed 1 11 2014].

[2 Texas Instruments, "PCB-Based Capacitive Touch Sensing with MSP430,"
5] October 2007. [Online]. Available:

http://www.ti.com/lit/an/slaa363a/slaa363a.pdf. [Accessed 20 November 2014].

[26] Texas Instruments, "AM335x Hardware Design Guide," 2014. [Online]. Available: http://processors.wiki.ti.com/index.php/AM335x_Hardware_Design_Guide. [Accessed 27 October 2014].

[27] Texas Instruments, "AM335x Thermal Considerations," 8 July 2013. [Online]. Available: http://processors.wiki.ti.com/index.php/AM335x_Thermal_Considerations. [Accessed 28 October 2014].

[28] C. Therobot, "Motors and Selecting the Right One," SparkFun, n.d.. [Online]. Available: https://learn.sparkfun.com/tutorials/motors-and-selecting-the-right-one?_ga=1.39803363.1578921122.1417395190. [Accessed 25 November 2014].

[29] The PHP Group, "PHP Hypertext Preprocessor," 2014. [Online]. Available: http://php.net/. [Accessed 25 10 2014].

[30] The Apache Software Foundation, "Apache HTTP Server Project," 2014. [Online]. Available: http://httpd.apache.org/. [Accessed 3 11 2014].

[31] M. Ester, H.-P. Kriegel, J. Sander and X. Xu, "A Density-Based Algorithm for Discovering Clusters," in *Kdd*, 1996.

[32] Bluetooth SIG, Inc., "Human Interface Device Profile," 2014. [Online]. Available: https://developer.bluetooth.org/TechnologyOverview/Pages/HID.aspx. [Accessed 2014 3 11].

[33] Atmel, "Atmel 8-Bit Microcontroller With 4/8/16/32KBytes," Atmel, San Jose, 2014.

[34] STMicroelectronics, *LSM303DLHC,* STMicroelectronics, 2011.

[35] STMicroelectronics, *L3GD20,* STMicroelectronics, 2013.

[36] SpectraSymbol, *Flex Sensor,* SpectraSymbol.

[37] Seeedstudio, "Seeed Wiki," 14 3 2014. [Online]. Available: http://www.seeedstudio.com/wiki/index.php?title=Bluetooth_Bee. [Accessed 10 2014].

[38] Wang, "063048 LI-POLYMER BATTERY Specification," UNIONFORTUNE, 2006.

[39] Texas Instruments, "AM335x Sitara™ Processors," June 2014. [Online]. Available: http://www.ti.com/lit/ds/symlink/am3358.pdf. [Accessed 25 Octover 2014].

[40] PCB-Pool, "PCB-Pool," 2014. [Online]. Available: http://www.pcb-pool.com/ppuk/index.html. [Accessed 25 November 2014].

[41] Beaglebone, "Beaglebone Black," Beaglebone, 23 November 2014. [Online]. Available: http://beagleboard.org/black#http://elinux.org/Beagleboard:BeagleBoneBlack. [Accessed 29 November 2014].

[42] Texas Instruments, "Arm Processors Overview," Texas Instruments, 1995-2014. [Online]. Available: http://www.ti.com/lsds/ti/arm/overview.page. [Accessed 18 October 2014].

[43] Texas Instruments, "Powering the AM335x with the TPS65217x," September 2014. [Online]. Available: http://www.ti.com/lit/ug/slvu551i/slvu551i.pdf. [Accessed 28 October 2014].

[44] Texas Instruments, "SINGLE-CHIP PMIC FOR BATTERY-POWERED SYSTEMS," April 2013. [Online]. Available: http://www.ti.com/lit/ds/symlink/tps65217.pdf. [Accessed 28 October 2014].

[45] "Beagleboard - BeagleBone Black," 3 December 2014. [Online]. Available: http://elinux.org/Beagleboard:BeagleBoneBlack. [Accessed 25 November 2014].

[46] "MTK41K256M16-32," Micron, 2011. [Online]. [Accessed 27 October 2014].

[47] Texas Instruments, "TPD4S012 4-Channel ESD Solution for USB-HS/USB OTG/USB Charger Interface," August 2014. [Online]. Available: http://www.ti.com/lit/ds/symlink/tpd4s012.pdf. [Accessed 3 November 2014].

[48] Texas Instruments, "USB 2.0 Board Design and Layout Guidelines," August 2014. [Online]. Available: http://www.ti.com/lit/an/spraar7c/spraar7c.pdf. [Accessed 20 October 2014].

[49] Texas Instruments, "Powerline Communications Analog Front-End," August 2010. [Online]. Available: http://www.ti.com/lit/ds/symlink/afe031.pdf. [Accessed 23 September 2014].

[50] Triad Magnetics, "Control Tranformer," 4 Dec 2013. [Online]. Available: https://system.netsuite.com/core/media/media.nl?id=9059&c=ACCT126831&h=88326ea7a1eab1e1de90&_xt=.pdf. [Accessed 12 November 2014].

[51] Texas Instruments, *MSP430FR573x Mixed-Signal Microcontrollers,* Dallas: Texas Instrument, 2014.

[52] Honeywell, *HIH-4030/31 Series, Humidity Sensors,* Minneapolis: Honeywell, 2008.

[53] Knowles, "SPU0410LR5H-QB Datasheet," March 2013. [Online]. [Accessed 28 November 2014].

[54] Texas Instruments, "TL972 Datasheet," May 2012. [Online]. Available: http://www.ti.com/lit/ds/symlink/tl974.pdf. [Accessed 28 November 2014].

[55] Atmel, "Atmel AT42QT1085," May 2013. [Online]. Available: http://www.atmel.com/Images/Atmel-9625-AT42-QTouch-BSW-AT42QT1085_Datasheet.pdf. [Accessed 20 November 2014].

[56] Atmel, "Buttons, Sliders, Wheels Design Guide- QTAN0079," August 2011. [Online]. Available: http://www.atmel.com/images/doc10752.pdf. [Accessed 25 November 2014].

[57] Atmel, "Proximity Design Guide- QTAN0087," Febuary 2012. [Online]. Available: http://www.atmel.com/Images/doc10760.pdf. [Accessed 25 November 2014].

[58] Texas Instruments, "TLC5951 Datasheet," March 2009. [Online]. Available: http://www.ti.com/lit/ds/symlink/tlc5951.pdf. [Accessed 20 August 2014].

[59] Analog Devices, *Low Voltage Temperature Sensors,* Norwood: Analog Devices, 2010.

[60] Coilcraft, "JA4429-AL Flyback Transformer Datasheet," 11 November 2010. [Online]. Available: http://www.coilcraft.com/pdfs/ja4429.pdf. [Accessed 28 October 2014].

[61] Linear Technology, "LT8610 Datasheet," 2012. [Online]. Available: http://cds.linear.com/docs/en/datasheet/8610fa.pdf. [Accessed 30 September 2014].

[62] Texas Instruments, "Mixed Signal Microcontroller," March 2013. [Online]. Available: http://www.ti.com/lit/ds/symlink/msp430g2955.pdf. [Accessed 14 November 2014].

[63] "MSP430x2xx Family User Guide," July 2013. [Online]. Available: http://www.ti.com/lit/ug/slau144j/slau144j.pdf. [Accessed 22 November 2014].

[64] Mercury Motors, "SM-42BYG011-25," 27 March 2009. [Online]. Available: https://www.sparkfun.com/datasheets/Robotics/SM-42BYG011-25.pdf. [Accessed 25 November 2014].

[65] Texas Instruments, "MSP430 32-kHz Crystal Oscillators," April 2009. [Online]. Available: http://www.ti.com/lit/an/slaa322b/slaa322b.pdf. [Accessed 12 November 2014].

[66] Texas Instruments, "STEPPER MOTOR CONTROLLER IC," November 2013. [Online]. Available: http://www.ti.com/lit/ds/symlink/drv8818.pdf. [Accessed 25 November 2014].

[67] T. Hopkins, "Stepper motor driving," 2012 November. [Online]. Available: http://www.st.com/web/en/resource/technical/document/application_note/CD00003774.pdf. [Accessed 25 November 2014].

[68] B. Schmalz, "Big Easy Driver," 11 November 2012. [Online]. Available: http://www.schmalzhaus.com/BigEasyDriver/. [Accessed 25 November 2014].

[69] Analog Devices, "Low Voltage Temperature Sensor," 2013. [Online]. Available: http://www.analog.com/static/imported-files/data_sheets/TMP35_36_37.pdf. [Accessed 15 November 2014].

[70] Suncourt, "Installation Instructions for In-line duct fans," n.d.. [Online]. Available: http://cache-m2.smarthome.com/manuals/301seriesfans.pdf. [Accessed 22 November 2014].

[71] NXP Semiconductors, "BT139-600 Datasheet," 27 September 2013. [Online]. Available: http://www.nxp.com/documents/data_sheet/BT139-600.pdf. [Accessed 1 December 2014].

[72] Allegro , "ACS711 Hall Effect Linear Current Sensor Datasheet," 18 July 2013. [Online]. [Accessed 1 December 2014].

[73] R. Ashworth, "Control Circuits for Air Condition & Heating Systems," 22 2 2013. [Online]. Available: http://highperformancehvac.com/basic-hvac-control-circuits-air-conditioning-heating-systems/. [Accessed 11 2014].

[74] T. Seel, J. Raisch and T. Schauer, "IMU-Based Joint Angle Measurement for Gait Analysis," *National Center for Biotechnology Information,* 2014.

[75] Circuit Design Inc, "RF Technical Resources," [Online]. Available: http://www.cdt21.com/resources/Modulation/modulation_FSK.asp. [Accessed 6 10 2014].

[76] Atmel, "Haptics Design Guide - QTAN0085," Febuary 2013. [Online]. Available: http://www.atmel.com/Images/Atmel-10758-AT42-QTAN0085-Haptics-Design-Guide_Application_Note.pdf. [Accessed 25 November 2014].

[77] Fairchild Semiconductor, "Bridge Rectifiers," October 2013. [Online]. Available: http://www.mouser.com/ds/2/149/GBU6A-188599.pdf. [Accessed 12 November 2014].