

# **SMART Water Heater**

**Senior Design 1 Final Paper**

**12-4-2014**



**Group 36**

**Mauro Cordoba**

**Bryan Mitchell**

**Vipol Sophonwatthanawichit**

# Table of Contents

<b>1 Executive Summary</b>	<b>1</b>
<b>2 Project Description</b>	<b>2</b>
2.1. Project Motivation and Goals	2
2.2. Objectives	2
2.3. Project Requirements and Specifications	3
2.3.1. Hardware Specification	3
2.3.2. Software Specification	5
2.3.3. Project Management	5
<b>3. Research related to Project Definition</b>	<b>5</b>
3.1. Existing Similar Projects and Products	5
3.1.1. Whirlpool Energy Smart Water Heater	6
3.1.2. GE GeoSpring Hybrid Water Heater	8
3.1.3. Other 'Smart' Water Heater Technologies	10
3.1.4. Nest Labs NEST Smart Thermostat	10
3.1.5. Honeywell Lyric Smart Thermostat	12
3.2. Relevant Technologies	13
3.2.1. Smart Thermostats	14
3.2.2. Temperature Control System	15
3.2.3. Embedded Systems	16
3.2.4. Wireless Networks and Communications	17
3.2.5. Mobile Application	18
3.2.5.1 iOS	18
3.2.5.2 Android	19
3.2.5.3 Comparison	20

3.2.6 Touch Screen Technology	22
3.2.6.1 Resistive Touch Screen	22
3.2.6.2 Capacitive Touch Screen	23
3.2.6.3 Surface Acoustic Wave Touch Screen	24
3.2.6.4 Infrared Touch Screen	24
3.3. Strategic Components	25
3.3.1. NEST	25
3.3.2. Controller Board	26
3.3.3. Touch Screen	29
3.4. Possible Architectures and Related Diagrams	29
3.4.1. Main Control Unit	30
3.4.2. Block Diagrams	31
<b>4. Project Hardware and Software Design Details</b>	<b>33</b>
4.1. Process Control Board	33
4.2. Main Control Unit	34
4.2.1. System Control	34
4.2.1.1. Temperature Sensing	35
4.2.1.2. Flow Sensing	38
4.2.1.3. Heater Control	39
4.2.1.4. Heat Pump Control	40
4.2.2. Communications and Intelligence	41
4.2.2.1. Operating System	42
4.2.2.1.1 Linux	42
4.2.2.1.2 File System	42
4.2.2.1.3 Applications	43
4.2.2.1.4 Web Server	43

4.2.2.2. NEST API	44
4.2.2.3. WIFI	47
4.3. NEST Interface	50
4.3.1. User Interface	50
4.3.2. Rooting nest	51
4.4. Touch Screen Interface	53
4.5. Power	54
4.5.1. Voltage Step-down Subsystem	57
4.6. Android Application	60
4.6.1. Application Design	61
4.6.2. User Interface	62
4.6.3. Data Displaying	66
4.7 Artificial Intelligence	67
4.7.1 Learning Algorithm	68
4.7.1.1 Simple Linear Regression (SLR)	68
4.7.1.2 Support Vector Machine (SVM)	69
4.7.1.2.1 Support Vector Classification	69
4.7.1.2.2 Support Vector Regression	70
<b>5. Design Summary of Hardware and Software</b>	<b>71</b>
5.1. Hardware	71
5.2. Software	72
<b>6. Project Prototype Construction and Coding</b>	<b>74</b>
6.1. Parts Acquisition and BOM	74
6.2. PCB Vendor and Assembly	76
<b>7. Project Prototype Testing</b>	<b>76</b>
7.1. Hardware Test Environment	76

7.1.1. Water Reservoir	76
7.1.2. Heating Surface	77
7.1.3. Water Pump	77
<b>7.2. Hardware Specific Testing</b>	<b>78</b>
7.2.1. Unit Test	78
7.2.1.1. Temperature Sensor	78
7.2.1.2. Flow Rate Sensor	79
7.2.1.3. Wi-Fi	79
7.2.1.4. Touch Screen	81
7.2.2. System Test	82
<b>7.3. Software Test Environment</b>	<b>84</b>
7.3.1. Touch screen	84
7.3.2. NEST Interface	84
7.3.3. Android Phone/Tablet	84
7.3.3.1 Genymotion	85
7.3.3.2 Samsung Galaxy S II	85
7.3.3.3 Google Nexus 7	85
<b>7.4. Software Specific Testing</b>	<b>85</b>
7.4.1. Functionalities	85
<b>8. Administrative Content</b>	<b>89</b>
8.1. Milestone Discussion	89
8.2. Budget and Finance Discussion	90
<b>9. Appendices</b>	<b>91</b>
9.1. Appendix A - Copyright Permissions	91
9.2. Appendix B - List of Tables and Figures	95
9.3. Appendix C - References	99

# 1 Executive Summary

Energy Conservation is no doubt one of the great important matters to everyone since we heavily rely on energy usage for many daily activities we do. With the increasing of technologies, more energy usage is required. Thus, many organizations have their researchers working on energy conservation to help us providing the solutions to conserve energy. Because energy supplies are limited, we are suggested to find ways to use energy wisely in order to maintain a good quality of life.

With an ever increasing motivation to conserve energy, much effort has been put forth lately to create smart HVAC systems. According to Duke Energy, water heaters are the second-highest source of energy usage in most homes. This is a largely overlooked area in which a smart solution could increase efficiency, reduce energy costs and carbon footprint, as well as giving the user a more refined control and ultimately comfort. There are several suggested solutions to help us reduce the energy usage for water heater such as buying the new and better water heating system, water conservation, insulate the existing water heater, and so on. Our goal is to offer the alternative solution to this problem.

Our idea for this project is to create a "smart water heater" with the focus of making the usage of water heater more efficient and increasing the awareness of energy usage to the household users. We are also looking to create the affordable platform to allow everyone to reduce their energy consumption and money from water heater. By doing so, we are introducing the water heater thermostat controller that will offer the way to control and monitor your water heater. The user will have the complete control of the water heater. We will also have the system programmed with the algorithm such that the system recognizes the pattern of energy usage and come up with the way to save more energy for each user. The user will be able to control the water heater through several devices that communicate with the main tank.

To make the system "smart", the main features of the system will be its connectivity ability, mainly WIFI, and data control, mainly data manipulation and pattern recognition. Our system will be using NEST as the key component of the design. NEST will record the user information such as water temperature, time stamp, and energy usage for example. With this data, the user will be able to monitor their energy usage. We will have several options that the user can monitor the data including the touch screen, with the appropriate GUI which is similar to the existing ones for the air conditioners, and android application, which can be installed on android devices. Other than monitoring the data, the user will also have a complete control over the water heater from those mentioned devices. The user will be able to set the temperature of the tank at any given time by using the android application or the touch screen. Despite the temperature control from the user and as we mention before, the system will use the data its

collect from the NEST over the certain period of time and come up with the pattern of the user behavior. Then it will use this information and predict when to turn on the temperature to the certain degree or to turn it off. This way, it offers more option to conserve the energy since the water heater will not be turned on when it isn't needed.

## **2 Project Description**

This project is intended to assist a homeowner in energy savings and convenience. The optional interface ability with the NEST programmable learning thermostat adds another layer of learning and data interpretation. The device design itself is intended to be compact and attached to the exterior of an existing water heater tank. Consisting of a touch screen display and an easy to use interface, the learning curve will be kept to a minimum.

### **2.1. Project Motivation and Goals**

After home heating and cooling, water heating is typically the second largest energy expense in the home because it is necessary for so many domestic activities. Despite buying the new water heater, there are several ways that one can reduce the water heating bills and energy usage. The most common solutions are water conservation and buying the new water heater system. Our motivation is to offer the alternate solution to the problem by creating a water heater thermostat controller that is easily installed in an existing water heater. The goal is to offer the user complete control of water temperature at any time of the day from anywhere with internet connectivity. The data of energy usage will be monitored and collected. The controller will be able to recognize patterns from the data that we collect. The controller will be programmed in such a way that it learns autonomously when to heat up water and cool off/turn off without explicit programming. Instead of controlling the temperature on the actual tank, we will have several devices that offer ways to control the water heater. The water temperature will be able to set remotely from a touch screen with the appropriate GUI much like existing air conditioners. The user will also have the access to the data and set the temperature remotely from android smart phones.

### **2.2. Objectives**

Objectives for this device are straightforward. It must be easy to use, from either the Android application or the touch screen interface of the device itself. In addition to being easy to use, it must accurately control the water heater element and learn from the collected usage data so that it can create a user schedule. In order to achieve the goals of making this project, the system requires several main components which are listed below:

- Heater Controller  
To create integrated interface that will control existing water heater
- Data Retrieving  
To access and obtain the data from NEST
- Devices  
To control the heater's functionalities, such as temperature, through NEST interface, Touch Screen, and Android device
- Connectivity  
To integrate the system with WIFI
- Presentation  
To create easy to use interface that gives out useful information on Android Application and Touch Screen

## **2.3. Project Requirements and Specifications**

The project is divided into 2 parts which are hardware section and software section. The hardware consists of the main subsystems for this project including thermostat, touch screen, controller board and the android devices. The software part mainly consists of the android application. Below are the list of minimum requirements of the system for our project.

### **2.3.1. Hardware Specification**

Below is the list of the general specifications for the hardware section.

#### **Heating Element/Thermostat**

This is the subsystem of the device that would directly integrate in to the water heater tank. In most cases, the design of this device will allow direct integration with the heating elements already present in the water heater. The specifications of the thermostat are as follows

- Must be equal to or under the size of most modern water heater thermostats
- Be able to control the temperature from input water temperature up to desired hot water temperature
- Temperature monitoring for controller unit
- Approximately 5 inches high by 3 inches wide by 2 inches deep for upper thermostats
- Approximately 3 inches high by 3 inches wide by 2 inches deep for lower thermostats
- Single element water heaters only have a single thermostat, and those sizes are approximately the same as the upper units in multi-element water heaters
- Standard household water heaters are 220V, thermostat device will need some method of voltage step-down to power the WIFI modules and electronics.



## **Touch Screen**

The touchscreen is integrated into the enclosure that contains the main control board. The enclosure and touchscreen are then attached to the tank of an existing water heater. The specifications of the touch screen are as follows

- 4.3 inch diagonal for an acceptable merger of affordability and viewing area.
- Be able to track at least single touch
- Low energy consumption
- GUI must be responsive to user
- WIFI connectivity

## **Controller Board**

The controller board will contain the processing hardware and memory. All of the components except the temperature sensor, relay connection to the heating element, and flow sensor will be encompassed inside the enclosure. The specifications of the controller board are as follows

- Must have sufficient outputs to interface with touch screen and wireless modules on the thermostats
- Have processing capability to handle calculations for temperature control, scheduling, and learning the hot-water requirements of the household
- The controller board need not be mounted directly on the water heater as it will communicate via wireless to the thermostat modules, so it can be powered via standard household 110V.
- Voltage Regulation: Controller board will need voltage stepdown from 110V to usable 5V/12V.
- For WIFI Interface, Access needs to be secure. Measures must be taken to deter possible attackers of gaining control of water heater
- Serial/UART Interface: Development and debugging
- Must be able to regulate water temperature +/- 1 C of desired temperature

## **Android Device**

The specifications of the android devices are as follows

- Version 2.1+
- At least 40 MB memory
- Have WIFI connection

### 2.3.2. Software Specification

Below are the specifications of the Android Application

- Compatible with Android version 2.1 and up
- Must be able to access the data via WIFI
- Must be able to communicate with the controlling mechanism of the heater
- Must be able to display the data in form of numbers, graphs, and charts
- Have friendly user interface and easy to use

### 2.3.3. Project Management

In order to keep the project manageable, the project management is important. We have divided the project into parts where each member of the group is responsible to each section the person is assigned to. Below is the list of the responsibilities of each member.

<b>Part</b>	<b>Name</b>
Microcontroller	Mauro Cordoba
Power System	Bryan Mitchell
WIFI Module	Vipol Sophonwatthanawichit
Temperature Sensor	Mauro Cordoba
Touch Screen	Bryan Mitchell
Artificial Intelligence	Vipol Sophonwatthanawichit
FTDI Serial	Mauro Cordoba
Heating Element	Bryan Mitchell
Android Application	Vipol Sophonwatthanawichit

**Table 2.1 Project Management**

## 3. Research related to Project Definition

Research related to this project included the current span of water heater technologies, and the programmable learning thermostats from Nest Laboratories and Honeywell Inc (the NEST and Lyric, respectively).

### 3.1. Existing Similar Projects and Products

As of this writing, the available 'Smart' water heaters are available from Whirlpool exclusively. There are multiple models from Whirlpool available, and there are other manufacturers bringing smart, learning water heaters to market.

General Electric's newest entry into energy saving appliances is the GeoSpring, which is a hybrid electric and heat pump water heater. It has user modes similar to Whirlpool's Smart Water, without the comprehensive schedule learning ability.

### **3.1.1. Whirlpool Energy Smart Water Heater**

Whirlpool Corporation's Energy Smart line of water heaters incorporate high efficiency counterparts to traditional electric water heater technology combined with a modern Energy Smart electronic control module.

The Energy Smart module allows users to have more control over the energy usage of their water heater. The Department of Energy estimates that water heater energy usage accounts for 15-25% of a home-owners energy footprint. The Energy Smart module has multiple operation modes to tailor that energy usage, lowering costs and raising energy usage efficiency by not heating water when hot water is not needed. The modes available on the Energy Smart line of appliances are;

- Normal Mode – standard water heater operation
- Energy Smart Mode – energy conservation setting
- Vacation Mode – puts the appliance in a stand-by mode for a set time, minimal energy usage



**Figure 3.1 Whirlpool Energy Smart Water Heater Tank**

In Normal mode, the control module operates the appliance as a standard water heater. It maintains a water temperature set by the user. Even in this mode, the Energy Smart line is more efficient than other standard water heaters by using electronic temperature controls and selectively powering the elements to use a minimum of energy.



**Figure 3.2 Whirlpool Energy Smart Water Heater Control**

The Energy Smart mode is more energy efficient than Normal Mode. In this setting, the control module allows a slightly larger variance in the temperature range. The user sets a desired temperature and the control module uses that as a maximum and automatically chooses a lower temperature (based on the maximum) as a lower limit. The control module monitors the water usage times and makes minor adjustments based on when hot water is required the most. After an amount of ‘learning’ time, the control module forms a basic schedule. For times of high hot water usage (showers and laundry) the control module will make certain that the temperature of the water is at or near the temperature ceiling set by the user. When outside of the normal high usage times, the control module will allow the water to fall to a lower temperature and maintain it at that level until a high-usage time approaches.

Tank and Parts Warranty	Item Number	Family Size	Gallon Capacity	1st Hour Delivery Gallons	Width	Height	Element Wattage (Upper/Lower)	Gallons Per Hour Recovery	Foam Insulation	Energy Factor
12-year	345708	2-3	40	54	22"	48-1/4"	4500/4500	20.7	3"	0.95
12-year	345709	3-4	50	58	24"	47-3/4"	4500/4500	20.7	3"	0.93
12-year	345705	5+	80	89	26"	60-1/4"	4500/4500	20.7	3"	0.92
9-year	345706	2-3	40	54	20"	48-1/4"	4500/4500	20.7	2"	0.92
9-year	345707	3-4	50	58	22"	48"	4500/4500	20.7	2"	0.92

**Table 3.3 Whirlpool Energy Smart Water Heater Specifications**

When in Vacation mode, the control unit maintains a much lower temperature (approximately 60°F). This allows the water heater to utilize very little energy but keeps the water temperature above freezing to prevent damage to the unit in cold climates.

The Energy Smart module also provides the user with diagnostic and safety features. The control module monitors the upper and lower elements to prevent dry firing, since energizing an element that isn't submerged will quickly destroy the element. The control module also monitors the thermistors and will adjust the energy usage to the elements should a thermistor fail. A lockout feature can also be used to stop unsafe or unauthorized temperature changes. In the event of element or thermistor misreporting, the control module can completely cut off current to the elements should the water temperature reach 180°F.

### 3.1.2. GE GeoSpring Hybrid Water Heater

General Electric's GeoSpring is a hybrid style water heater. The GeoSpring utilizes both standard electric elements and heat-pump technology. By utilizing multiple methods of heating the water, the GeoSpring is incredibly efficient. GE states that the GeoSpring will reduce energy costs related to water heating by approximately 60%.



Figure 3.4 GeoSpring Hybrid Water Heater Control

Like the Whirlpool Energy Smart, the GeoSpring has multiple user modes. The Heat Pump and Hybrid modes are exclusive to heat-pump style water heaters.

- Heat Pump Mode – full heat pump operation, no electric element
- Hybrid Mode – combination heat pump and electric element mode
- High Demand Mode – high energy usage electric elements almost exclusively
- Standard Mode – acts as a standard electric water heater, utilizes electric elements
- Vacation Mode – appliance is placed in stand-by, minimal energy usage

In Heat Pump mode, the GeoSpring does not utilize the electric heating elements. It relies solely on the heat-pump to transfer heat from the surroundings to the water, making it much more efficient than standard water heaters. One of the major shortcomings of the heat-pump mode is one found in all heat-pump type appliances; they work poorly in areas of low ambient temperature.



**Figure 3.5 GeoSpring Hybrid Water Heater Tank**

Hybrid Mode uses the heat-pump in conjunction with the standard electric element. If the hot water usage is high enough, the control module will cycle the electric elements for a short time to supplement the heating ability of the heat-pump. This mode is not as efficient as Heat Pump mode, but more efficient than Standard and High Demand.

High Demand mode is similar to hybrid mode, but the control module keeps the electric elements on for a longer amount of time allowing for faster water heating at the cost of efficiency.

In Standard Mode, the GeoSpring uses only the electric heating elements. This mode is used in colder conditions, where heat pumps are not ideal due to low or absent ambient heat energy.

Vacation Mode for the GeoSpring is similar to the Whirlpool Energy Smart. The user activates Vacation Mode and the unit maintains the water at a minimum temperature of 50-60°F, warm enough to prevent freezing at minimal energy usage. If the user chooses to input the length of the vacation, the GeoSpring control module will activate the previously used mode right before the user is slated to return home.

Preliminary designs of the water heater control device will utilize the Geospring as a reference for our various settings for energy savings. With the Geospring being more complex and more capable than a standard water heater, if our device can interface with a Geospring then it would be able to interface with a much simpler standard electric water heater.

### **3.1.3. Other 'Smart' Water Heater Technologies**

While not an actual water heater, one technology being developed will assist in turning water heaters in to 'smart' appliances. Emerson is designing and testing the Electric Water Heater Control device. This device is provided to end-users by the energy utility companies. After installation (which currently does not require an electrician) the device allows the utility company to cycle power to the water heater as needed. Used in short time slots, approximately 10 to 15 minutes, it will allow energy saving without a noticeable impact on water temperatures. While also saving energy, this control will allow utility companies to cycle off the water heaters very quickly in times of sudden energy requirement spikes. With a much faster reaction time than humans, the networked EWHC devices will be able to near-instantly power down the water heater, allowing energy utilities to compensate quicker and making brown-outs and other demand-related outages be resolved much sooner. Future devices are also planned to incorporate a learning algorithm which will monitor usage and allow the user to automate control of the water heater during low usage times, even if the utility company does not cycle if off.

### **3.1.4. Nest Labs NEST Smart Thermostat**

The NEST smart thermostat is not a water heater, and out-of-the box it does not have the capability of controlling a water heater. However, with this project's design team collaborating with Dr. Yier Jin, the team has access to the intricate inner workings of the NEST thermostat. Dr. Jin provides the team with root access to a NEST thermostat, allowing inclusion of the NEST into the wifi network of the water heater control device.

The NEST on its own is a powerful addition to home automation. Some of the capabilities of the NEST include:

- **Learning:** Using dual core Arm processors and a potent learning algorithm, the NEST takes note of when the user adjusts the temperature. With a learning algorithm, the NEST spends the first seven to ten days learning the schedule and temperature preferences of the homeowner. Before the end of the second week, the NEST knows what temperature the homeowner wants, and when. The NEST can also learn how long it takes the user's home to heat or cool, and display an approximate time for the home to reach the set temperature.
- **Motion and Light Sensors:** various sensors allow the NEST to automatically light up when someone approaches if the ambient light around the NEST is low. Also, using the motion sensors, the NEST can detect when people are present in the home. If all occupants are away for the day, the NEST auto adjusts the temperature to the Away setting, to save energy.
- **Humidity Sensors:** The NEST takes in to account the ambient relative humidity of the surroundings. If the humidity is low, the NEST will run the air conditioner a little less, allowing energy savings.
- **Reminders and Notices:** The NEST will also track filter changes, and alert the user to when the filter needs to be changed. In the event of a heating or air conditioner unit failure, the NEST will detect that the temperature is not changing enough in a certain amount of time, and alert the homeowner to a possible malfunction with the heating and air conditioning system.



**Figure 3.6 Nest Thermostat**

- **Weather:** The NEST can pull weather information from the internet based on your area code. This information allows the NEST to fine tune heating or air conditioning based on historical user settings for days of similar temperature and humidity.



- **Remote App Capability:** The NEST's remote application allows the homeowner to monitor and change the settings of the NEST from any smartphone or tablet with wireless or cellular network connections. As of this writing, the NEST remote application is available for Android, Apple, and Windows operating system smartphones and tablets.

With root access to the NEST, our water heater control device will have more robust capabilities as well. The ability to pull scheduling data from the NEST will allow our device to have knowledge of when the homeowner is present in the home. Using the temperature data from the NEST, the water heater control device will also be able to compare the user's desired temperature to the ambient temperature. This can allow some energy savings by lowering the water temperature a degree or two when the ambient temperature of the home is above a certain limit.

### 3.1.5. Honeywell Lyric Smart Thermostat

Another entry into the smart thermostat field is the Lyric, by Honeywell. Designed as a direct competitor to the NEST, it has all of the capabilities of the NEST in addition to Geofencing. Utilizing the Lyric remote application and the smartphone's GPS, the Lyric is able to determine when the user is approaching and within a certain radial proximity to the home. This allows the Lyric to get the ambient temperature to the homeowner's desired temperature by the time he or she arrives. The Lyric has multiple settings for geofencing. One setting has the geofence in a radius of miles, for rural areas. The other allows the user to set the radius in feet, for the more compact living of urban environments.



Figure 3.7 Honeywell Lyric Thermostat

- **Learning:** Like the NEST, the Lyric has capable processors and an intricate learning algorithm. The Lyric relies on its geofencing capabilities slightly more than the learning algorithm, however it is still capable of producing temperature change schedules for home occupants.
- **Motion and Light Sensors:** The Lyric includes various light and motion sensors to automatically set Away status, or to automatically light the thermostat if the ambient light is too low for easy viewing. The Lyric also includes sensors to adjust the temperature settings of the thermostat if it is in direct view of sunlight. Since sunlight can heat the enclosure of the thermostat and alter the thermostat's perception of the actual ambient temperature of the home, the Lyric takes this in to account and adjusts the readings of its own temperature sensors to correct.
- **Humidity Sensors:** The humidity sensors of the Lyric are used in a similar fashion to those of the NEST. Should the relative humidity of the environment be low, the Lyric can adjust the temperature a degree or two, allowing for energy savings without causing discomfort to the homeowner.
- **Reminders and Notices:** The Lyric thermostat and remote app work in conjunction to keep the user alerted to any issues. Should the heater or cooling units fail, the Lyric will be able to detect this failure by monitoring how long the heating or cooling unit is taking to adjust the home's temperature compared to historical readings. The Lyric will also alert the user to when it is time to change the air filters.
- **Remote App Capability:** The Lyric's remote app allows the user to change the schedule or temperature from a smartphone or table. The Lyric remote app also includes the geofencing capabilities mentioned above, out-of-the-box. As of this writing, the Lyric remote application is only available on Apple's iOS and Android OS power smartphones and tablets.

Currently, the NEST does not have first-party geofencing. The NEST requires the usage of a third party smartphone application to interface with the official NEST remote application to give it geofencing capabilities.

The design team does not have root or API access to the Lyric as it does the NEST, so while the Lyric may edge out the NEST as a more capable device, the NEST will be used as our reference to an extra control vector and data center for our design.

### **3.2. Relevant Technologies**

Technologies most relevant to this project would be centered around home automation, learning usage times, and developing a schedule based on that usage. The prime examples at current writing would be the NEST thermostat by Nest laboratories, and the Lyric learning thermostat by Honeywell Inc. The following technologies are representative of the basic building blocks that we will need to use or implement in order to realize our system design.

### 3.2.1. Smart Thermostats

The Nest Learning Thermostat designed by Nest Labs was a revolutionary product introduced to the public in 2011. It managed to rethink something as mundane as a thermostat in a way that made an often forgotten appliance more user-friendly, aesthetically pleasing and efficient. This thermostat is capable of learning usage needs autonomously and lets users actively monitor and effectively control its functions from anywhere in the world thanks to its internet connectivity.

Under the hood, the Nest is a Linux system running on a Sitara ARM Cortex A8 microprocessor. It also has sensors that are used to determine occupancy in the living space to make smart decisions about whether to turn on or off certain electrical components. HVAC components are controlled through industry standard connections offered in every thermostat. A key feature for us is that this particular thermostat is connected to the internet using its WL1270B module. Because this appliance is somewhat dependent on cloud services, it also has a well-documented API that will be incredibly useful to us in order to tie it into our system.

One of the Nest's greatest accomplishments is its user-friendliness. While it can be remotely controlled through a computer, phone or tablet, its main user interface consists of an LCD screen with a rotating ring around it. This has proven to be incredibly intuitive for end users.



Figure 3.8 Smart Thermostats

Honeywell recently introduced their own smart thermostat taking many design cues from the Nest, which has set a standard for these new appliances. While the core functionality is the same, the process is a little different; The Nest uses built-in near field and far field proximity sensors to determine occupancy. The Lyric on the other hand does not use any sensors but instead uses its 'Geofencing' feature from the phone app. By using the mobile device's GPS location, a perimeter can be set around the house.

### 3.2.2. Temperature Control System

Temperature control systems depend on the ability to measure the temperature and the ability to control the rate at which it changes. Temperature can be measured using some of the following devices, which are as a whole referred to as thermometers.

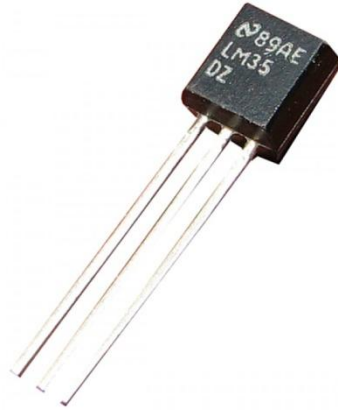
- Infrared thermometers – Heat can be detected by infrared radiation. Infrared thermometers focus and absorb the IR energy from the surface of an object and turn into electricity that can be displayed as a number.



Figure 3.9 Infrared Thermometer

- Thermistor – A thermistor's resistance changes along with its temperature. A temperature to voltage converter can be easily implemented by using a thermistor as part of a Wheatstone bridge.
- Thermocouple – A device made by joining two dissimilar metals on two separate spots. A temperature difference between the contact spots will create a voltage that is proportional to the temperature change.

- IC solutions – There are IC sensors such as the LM35 capable of accuracies of less than one degree Celsius.



**Figure 3.10 LM35 Temperature Sensor**

The control loop part of the system is usually on of the following

- On/off – the system will turn on or off depending on which side of the set point it is. This type of control may have hysteresis, or a margin of inactivity to avoid quickly switching on/off
- Proportional (P) controller – the amount the system is “on” is proportional to the error, or how far away it is from the set point.
- Proportional/Integral/Derivative (PID) controller – this type of controller not only looks at the current error, but also at the sum of the error over time (Integral) as well as the rate of change of the error (Derivative).

### **3.2.3. Embedded Systems**

Embedded systems are all around us and they are driving the IoT movement. It is thanks to these specialized computer systems that we can connect more and more things to the internet and even control them in many cases. As opposed to regular computer systems that are designed to handle a broad spectrum of applications, embedded systems usually are designed to handle a particular task, or in more complex scenarios, many different (but possibly related) tasks towards serving a common goal.

Similar embedded systems of ranging complexities can be found around the average home. A microwave oven consists of an embedded system which acts upon users pressing buttons to run timers and cook for a predetermined amount of time. Smart TVs have embedded systems that allow the TV itself to connect to the internet and access internet services without the aid of general purpose

computer. Thermostats have embedded systems that control contact closures to turn on and off compressors and fans, etc.

Alarm Clock	Microwave
Cell phone	Modem/Router
Thermostat	Digital Picture Frame
IP Camera	Smoke Detectors

**Table 3.11 Examples Of Embedded Systems Usually Found Around Homes**

### **3.2.4. Wireless Networks and Communications**

Wireless communications come in many flavors for just as many different uses. We are specifically concerned with wireless Ethernet networks. These networks are governed by the IEEE 802.11 standards that lay out MAC and Physical layer specifications. There are many problems associated with wireless networks such as collisions and interference that are mitigated by using a standard that takes care of these issues without much involvement on higher layers.

There are many protocols under the 802.11 family currently being used. The most commonly offered protocols by house routers are 802.11b/g/n. The main differences between these protocols are the bandwidth and number of channels available, modulation scheme and data rate. Since our design heavily relies on communicating with the Nest thermostat we must provide our controller board Wi-Fi connectivity that will work on the majority of the homes. For this reason we will use 802.11g or 802.11n depending on which router is available for testing.

Our thermostat will potentially be exposed to the Internet and not just our LAN. This makes the use of encryption imperative for security purposes. Since WEP and WPS can be cracked and thus not safe, we will not consider them a viable solution and as such we will use WPA2 as our encryption protocol.

	WEP	WPA	WPA2
Name	Wired Equivalent Privacy	Wifi Protected Access	Wifi Protected Access 2
Combo	24 bit initialization keys 16.7 million combination	48 bit initialization keys 500 trillion combinations	48 bit initialization keys 500 trillion combinations Advanced Encryption Standard
Encryption	64 bits 128 bits	64 bits 128 bits	64 bits 128 bits
Keys	Static encryption keys	Unique encryption key	Unique encryption key
Speed	Not much processing power	Somewhat processing power	Requires greater processing power
Master Key	Master keys are used directly	Master keys are never directly used	Master keys are never directly used

**Table 3.12 Comparison Between Different Security Protocols**

### 3.2.5. Mobile Application

When it comes down to software development for mobile, Android and IOS have been the two major platforms in the field. Because mobile application is part of our system for this project. It is important to understand the pros and cons from each of these platforms so that we can choose the one that is best for the project.

#### 3.2.5.1 iOS

iOS is developed by Apple. It is no doubt that the platform is known for the quality of the hardware since Apple has high standard for its devices. iOS uses iTunes app store that has strict app policies. The store uses these policies to keep average app quality high. The iOS interface is extremely good and overall user experience does not change drastically from what users are familiar with.

On the cons side, iOS is known for the lack of customization. Because iOS maximizes user friendliness, Apple has created a fixed environment which assumes default setting are good enough for every user. The user will need jailbreak for extensive customization. The iOS devices lacks of device options. There usually be a few device options available at any given time.



**Figure 3.13 iOS Devices**

### **3.2.5.2 Android**

Android is developed by Google. It has its operating system based on the Linux kernel. By starting off with the pros, android offers huge device variety. It comes in all shapes, sizes, and prices. There are several manufacturer that runs their own special blend of Android, providing more features and capabilities. Android devices and development kit are open source which is more customizable. The platform has less restrictive application policies. Google Play store for android is not the only way to distribute android application. User can also distribute the application through any source as long as the source can get the android application apk file to the device. Android is very versatile software. It can be use with many types of devices, not just phone and tablet.

On the other hand, android has several known cons when comparing to iOS. Because of the lighter restrictions, android application are not as dependable as iOS. Side loading from disreputable sources is a surefire wat to get infected with malicious software. The quality of the devices is known to be inferior to those of iOS. Most manufacturers for android devices don't have Apple's reputation when it comes to reliably releasing good products.





**Figure 3.14 Android Devices**

### 3.2.5.3 Comparison

Below is the comparison of the basic features between Android and iOS:

<b>Platform</b>	Android	iOS
<b>Developer</b>	Google	Apple Inc.
<b>OS family</b>	Linux	OS X, UNIX
<b>Customizability</b>	A lot. Can change almost anything.	Limited unless Jailbreak
<b>Programmed in</b>	C, C++, java	C, C++, Objective-C
<b>Easy media transfer</b>	depends on model	with desktop application
<b>Source model</b>	Open source	Closed, with open source components.

**Table 3.15 Comparison between Android and iOS 1**

<b>Platform</b>	Android	iOS
<b>Open source</b>	Kernel, UI, and some standard apps	The iOS kernel is not open source but is based on the open-source Darwin OS.
<b>Available on</b>	Many phones and tablets, including Kindle Fire(modified android), LG, HTC, Samsung, Sony, Motorola, Nexus, and others. Also, Google Glasses	iPod Touch, iPhone, iPad, Apple TV (2nd and 3rd generation)
<b>App store</b>	Google Play – 1,000,000+ Apps. Other app stores like Amazon and Getjar also distribute Android apps. (unconfirmed ".APK's")	Apple app store – 1,000,000+ Apps
<b>Market share</b>	81% of smartphones, 3.7% of tablets in North America (as of Jan'13) and 44.4% of tablets in Japan (as of Jan'13). In the United States in Q1 2013 - 52.3% phones, 47.7% tablets.	12.9% of smartphones, 87% of tablets in North America (as of Jan'13) and 40.1% of tablets in Japan (as of Jan'13)
<b>Available language(s)</b>	32 Languages	34 Languages
<b>Device manufacturer</b>	Google, LG, Samsung, HTC, Sony, ASUS, Motorola, and many more	Apple Inc

**Table 3.16 Comparison between Android and iOS 2**

As far as ease of implementation, both iOS and Android have plenty of API's and resources available. However, the android is not limited to those devices. Android is powering all kinds of things. In this situation, android is more flexible platform. From the development standpoint, iOS operates in a narrower field while android sprawls out over everything. Also, according to the market share, there are more android user than those of iOS.

**Worldwide Smartphone Sales to End Users by Operating System in 2013 (Thousands of Units)**

<b>Operating System</b>	<b>2013 Units</b>	<b>2013 Market Share (%)</b>	<b>2012 Units</b>	<b>2012 Market Share (%)</b>
Android	758,719.9	78.4	451,621.0	66.4
iOS	150,785.9	15.6	130,133.2	19.1
Microsoft	30,842.9	3.2	16,940.7	2.5
BlackBerry	18,605.9	1.9	34,210.3	5.0
Other OS	8,821.2	0.9	47,203.0	6.9
<b>Total</b>	<b>967,775.8</b>	<b>100.0</b>	<b>680,108.2</b>	<b>100.0</b>

Source: Gartner (February 2014)

**Figure 3.17 2012-2013 Smartphone Market Share**

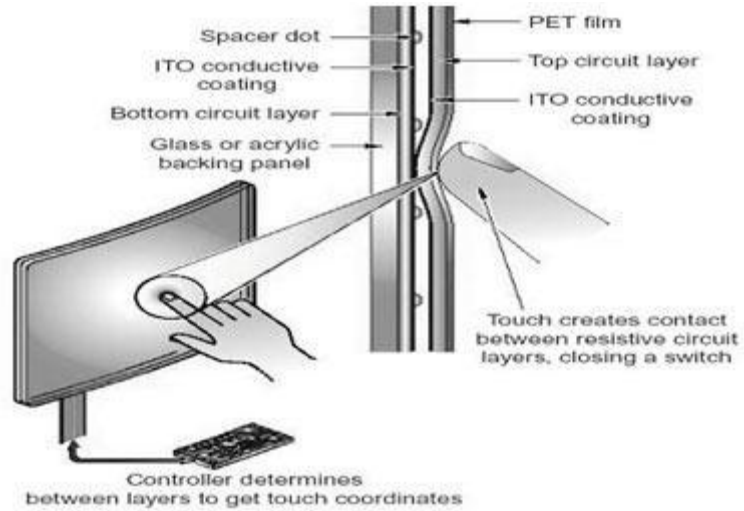
Because of the population of android users and the limitation of what iOS can do in terms of software development, we have concluded that Android is the better platform for our project.

### **3.2.6 Touch Screen Technology**

Touch screen technology is the direct manipulation type gesture based technology which can be categorized into four main touch screen technologies

#### **3.2.6.1 Resistive Touch Screen**

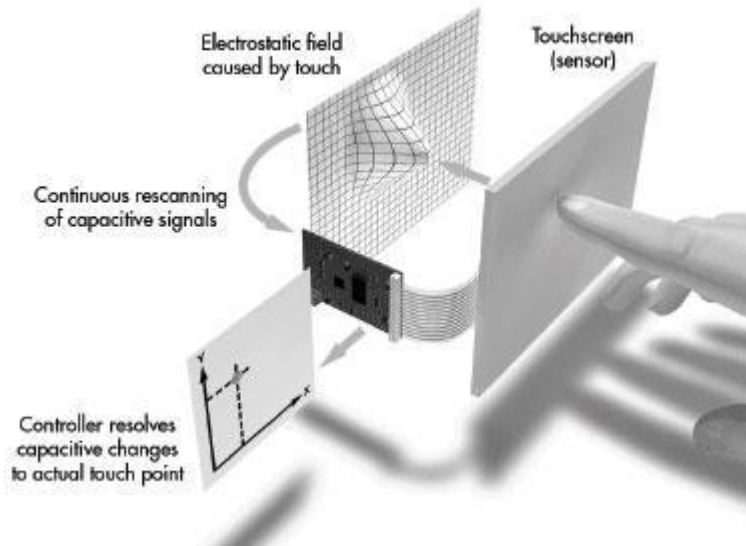
The resistive touch screen consists of a flexible top layer made of Polyethylene and a rigid bottom layer made of glass. Both the layers are coated with a conducting compound called Indium Tin Oxide and then spaced with spacers. While the monitor is operational, an electric current flows between the two layers. When a touch is made, the flexible screen presses down and touches the bottom layer. A change in electrical current is hence detected and the coordinates of the point of touch is calculated by the controller and parsed into readable signals for the operating system to react accordingly.



**Figure 3.18 Resistive Touch Screen Technology**

### 3.2.6.2 Capacitive Touch Screen

The Capacitive Touch screen Technology is the most popular and durable touch screen technology used all over the world at most. It consists of a glass panel coated with a capacitive (conductive) material Indium Tin Oxide (ITO). The capacitive systems transmit almost 90% of light from the monitor. Some of the devices using capacitive touch screen are Motorola Xoom, Samsung Galaxy Tab, Samsung Galaxy SII, Apple's iPad.



**Figure 3.19 Capacitive Touch Screen Technology**

### 3.2.6.3 Surface Acoustic Wave Touch Screen

The Surface Acoustic Wave Touch screen technology contains two transducers (transmitting and receiving) placed along the X-axis and Y-axis of the monitor's glass plate along with some reflectors. The waves propagate across the glass and are reflected back to the sensors. When the screen is touched, the waves are absorbed and a touch is detected at that point. These reflectors reflect all electrical signals sent from one transducer to another. This technology provides excellent throughput and image clarity.

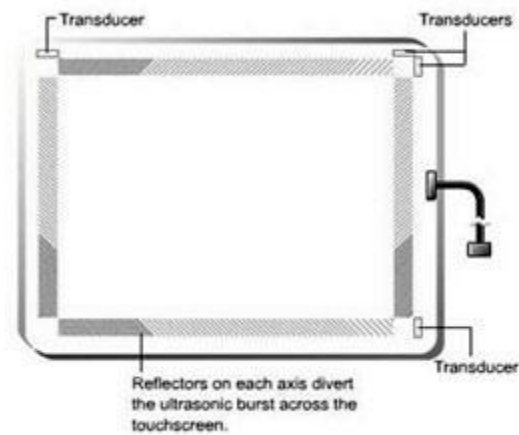
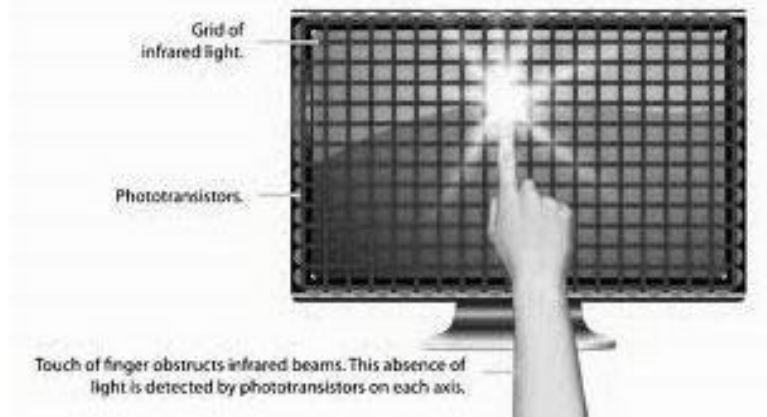


Figure 3.20 Surface Acoustic Wave Touch Screen Technology

### 3.2.6.4 Infrared Touch Screen

In the Infrared Touch screen Technology, an array of X- and Y- axes are fitted with pairs of IR Leds and photo detectors. The photo detectors detect any change in the pattern of light emitted by the Leds whenever the user touches the monitor/screen.



**Figure 3.21 Infrared Touch Screen Technology**

### **3.3. Strategic Components**

These are the necessary components we will need to design or modify and use to control the temperature of the water tank.

#### **3.3.1. NEST**

While we obviously will not design or redesign the Nest, it will be a key component of our design for two reasons. First of all, it will provide the initial “smartness”. We will poll the Nest for its occupancy records and patterns in order to have a starting point for our own algorithms. Having this starting point will make the water heater instantly useable and our own sensors will gradually fine tune its operation for maximum efficiency. The second reason the Nest is so important to us is that its intuitive ui design will make it easier for the user to interact with the water heater. Thermostats are placed with users in mind. Since water heaters are not something that normally need user interaction, their infrastructure is usually out of reach. By integrating the Nest into our systems the users will not have to make any modifications to their homes and will have total control of their appliances.

Nest has very conveniently documenter their API. With this information we hope that it will be enough for to pull all the data we need from the water heater. In the even that their API is not enough, Dr. Jin has extensive experience hacking the nest hardware and software. With this guidance we will modify the firmware to send the data to the thermostat as needed.

We also plan on making a custom interface for the water heater on the Nest itself while retaining all of its original functions. Again, with the guidance of Dr. Jin we will extend its firmware to feature a “water heater mode.” We will have to come

up with a communication protocol or API for the controller to accept commands sent from the Nest. In all likelihood since the Nest API works by sending JSON encoded data, we will make our controller board work in the same way to keep all as compatible and concise as possible.

### 3.3.2. Controller Board

The controller board will be based around the Xilinx Zynq 7000 programmable SoC. Since our functions will not need to react in the order of milliseconds or even seconds, we will base our system on the Linux operating system which can run natively on this board. This will allow us to keep different functions on different scripts and the OS will manage them accordingly.

All of our smartness will come from this SoC. It will be responsible for keeping track of times of interest and status, our algorithms for turning on and off the heating elements as well as the actual actuation, communications, and storing data.

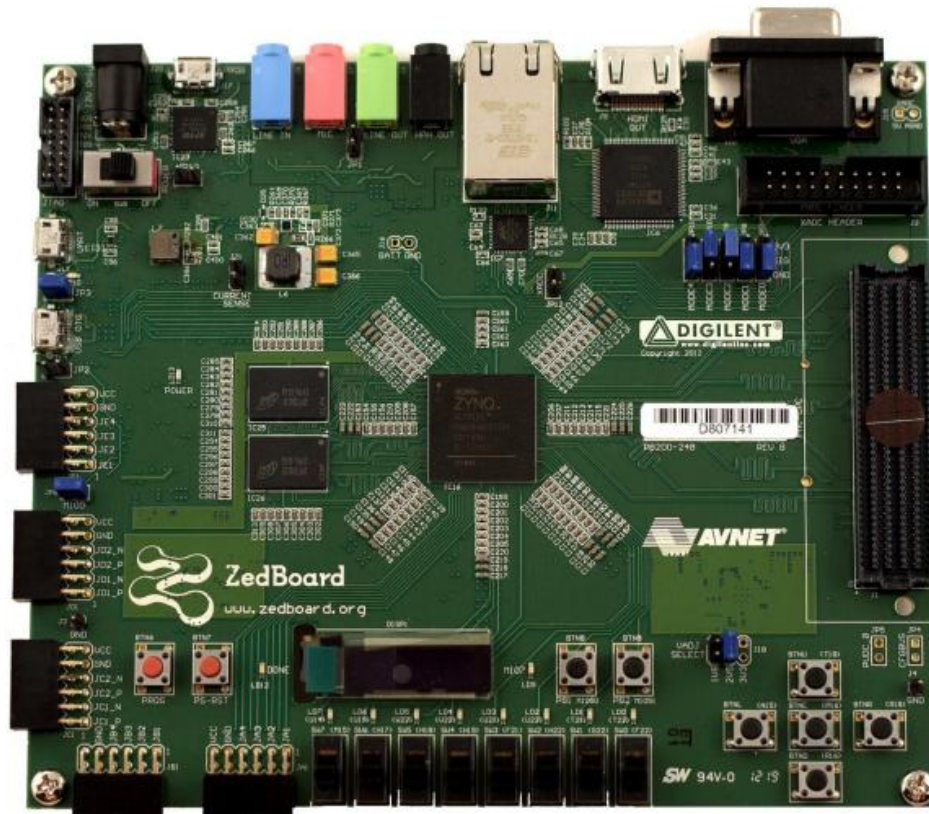


Figure 3.22 ZedBoard Xilinx Zynq-7000

The hardware will be relatively simple. The Zynq 7000 has many built in buses. GPIO could be used to drive relay drivers which will in turn energize the relays when needed. Depending on the sensors we end up getting, we could use I2C or GPIO. Since we also have native USB support, we can shop for a compatible Wi-Fi dongle and that would take care of the communication.

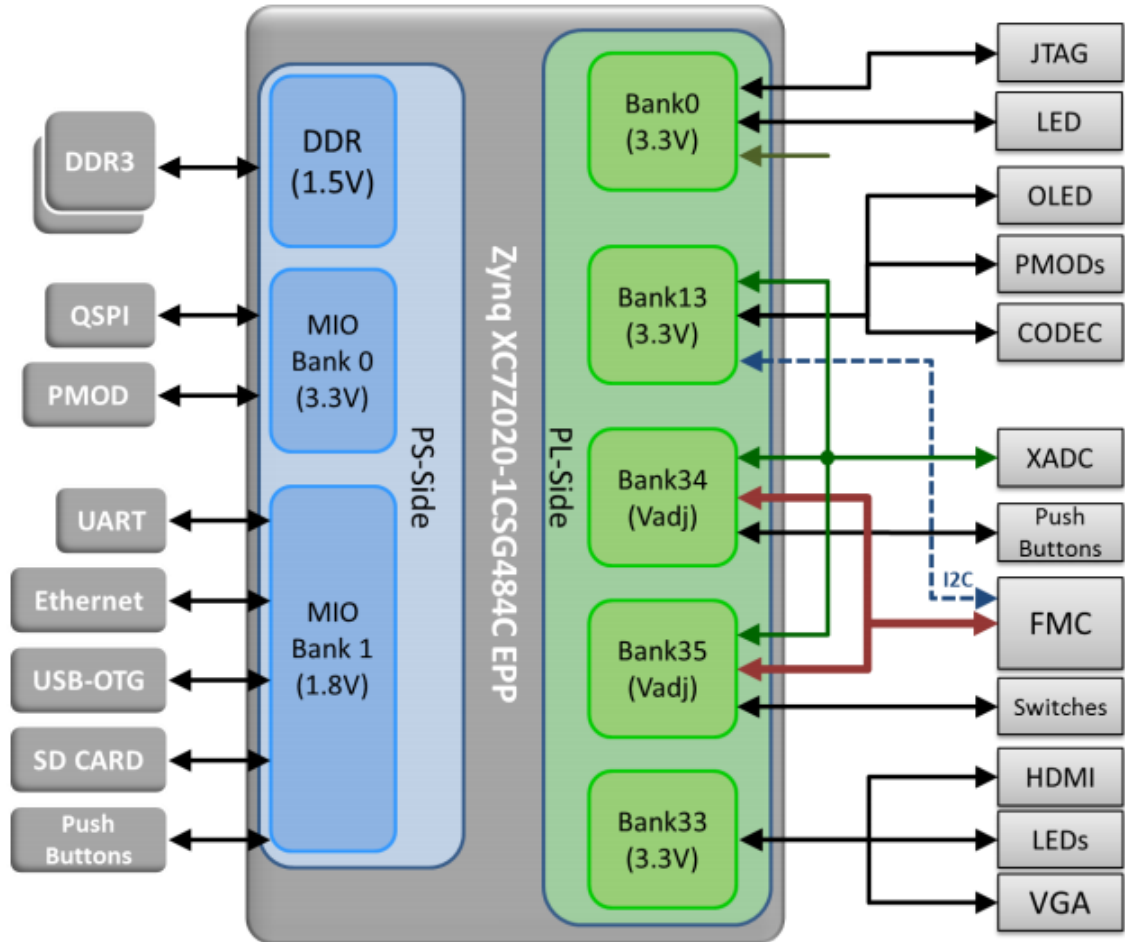


Figure 3.23 Zynq Z7020 CSG484 Bank Assignments



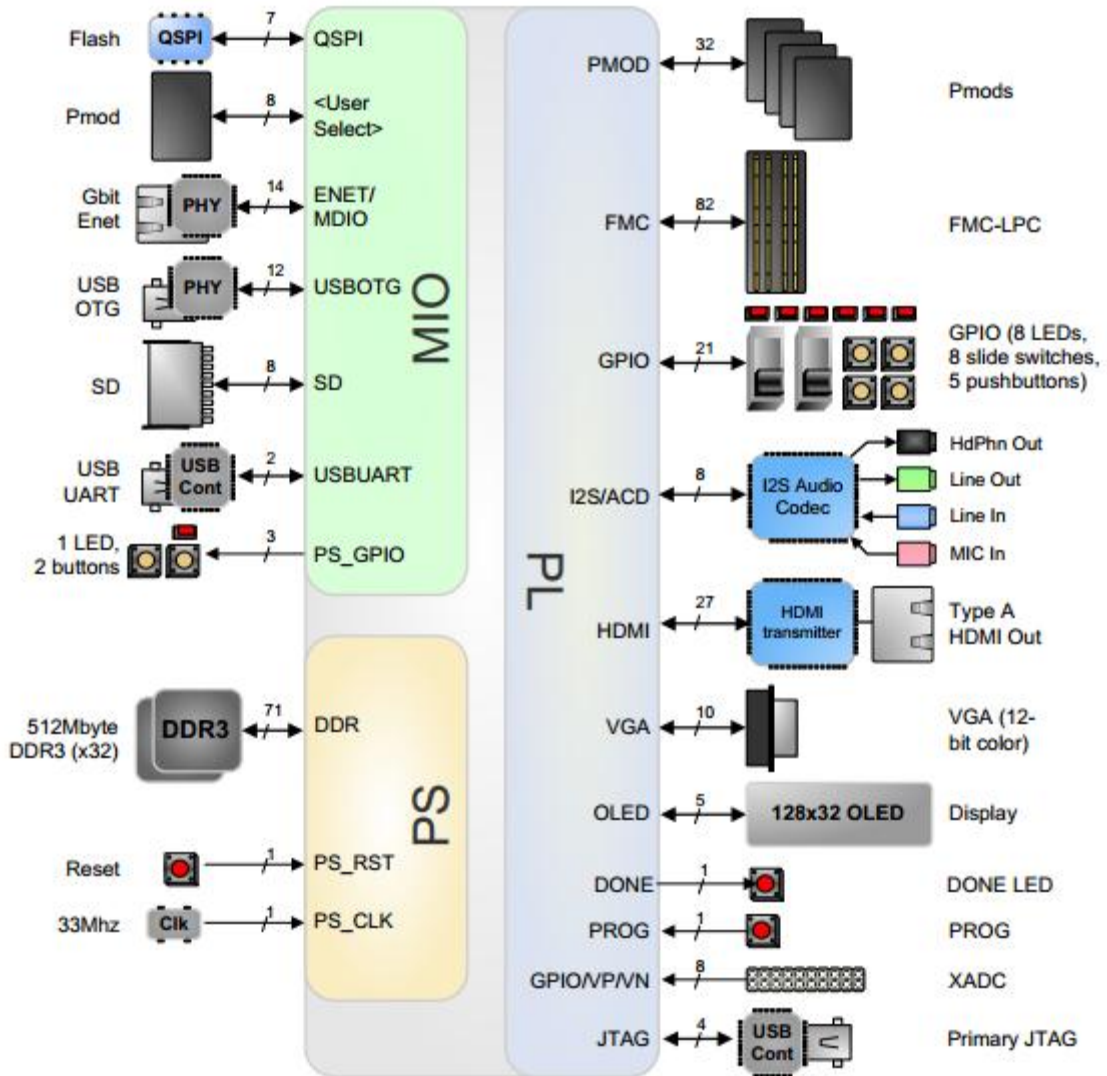


Figure 3.24 ZedBoard Block Diagram

### 3.3.3. Touch Screen

The on board user interface will be used for initial setup (ip address, etc) as well as displaying important messages such as status and faults. The size does not matter too much; it does not need to be huge. However, We would like it to be color. Resistive touch capabilities should be more than enough since no multitouch will be needed.

In order to use the touchscreen with our system we will need to compile our kernel with the necessary drivers and libraries for the screen and the touch panel.



Figure 3.25 CFAF320240F-035T Touch Screen

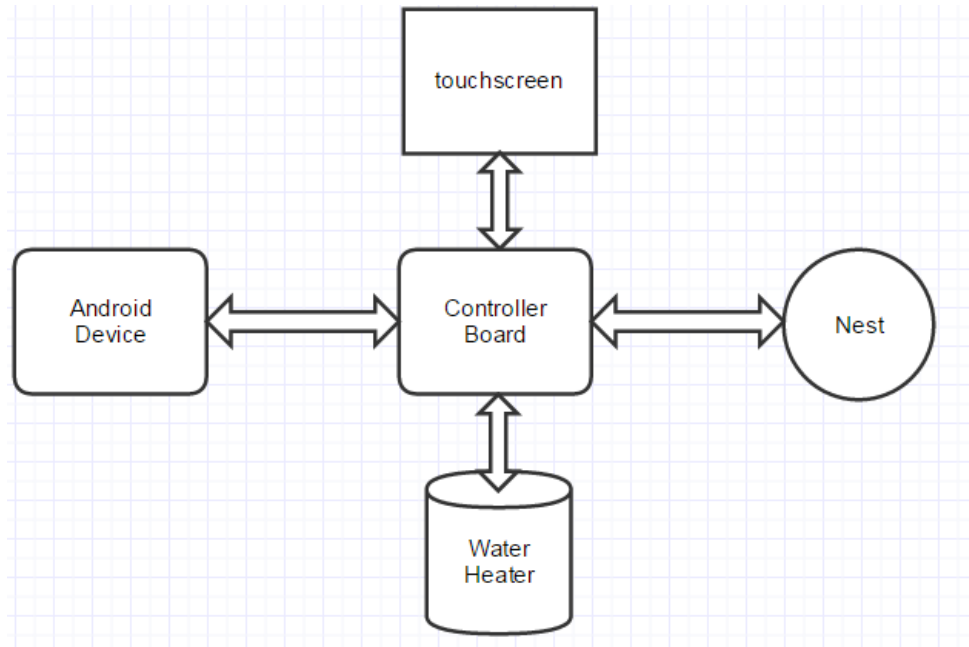
### 3.4. Possible Architectures and Related Diagrams

In order to achieve the best possible tradeoff between hot water availability and power savings we have come up with the following system. Operating under the assumption that there will be a certain minimum infrastructure already in place, mainly having a Nest Smart Thermostat (and thus also having a wifi network), the system will be comprised of the following five components.

- The water heater will be a standard tank. It will have inlet and outlet pipes that will be fitted with flow sensors. It will have water temperature sensors to keep track of the overall temperature and it will have a heating element as well as a heat pump.
- The Nest will act mainly as a remote “people” sensor, but will also be used as an auxiliary display

- The android device will be used as a remote control and display, giving the user control from anywhere with internet connectivity.
- The control board will be central to all the other peripherals, controlling the water heater. It will provide the bridge between a standard water heater and the internet.

The touch screen will serve as an on board display and data entry to provide initial configuration.



**Table 3.26 Basic Architecture of System**

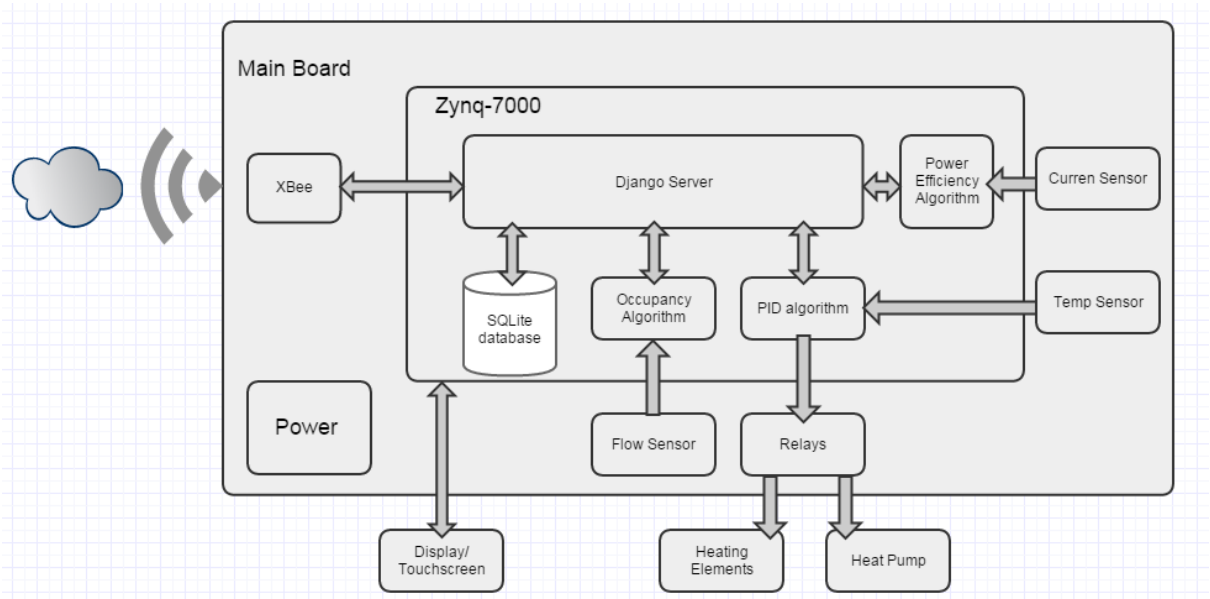
It is somewhat ironic that the user will be presented with three different means of controlling the appliance since normally the only way to interface is a dial, especially when we aim to make the water heater smarter and cut out user involvement as much as possible. The reason for having all these displays is to emphasize the connected nature of the new water heater as well as giving the user a superior control with minimum effort.

### **3.4.1. Main Control Unit**

The controller board will be based around the Zynq-7000 System on Chip. The board design will consist mostly of the power supply subsystem and the sensors and wifi module connected to the microcontroller.

Data collected from the wifi connection will be parsed in the web server and stored in the database to be later examined by the different algorithms. Sensor data will also be saved to this same database to keep data concurrent. The web server will have direct access to these scripts and through them we'll be able to control our heating elements.

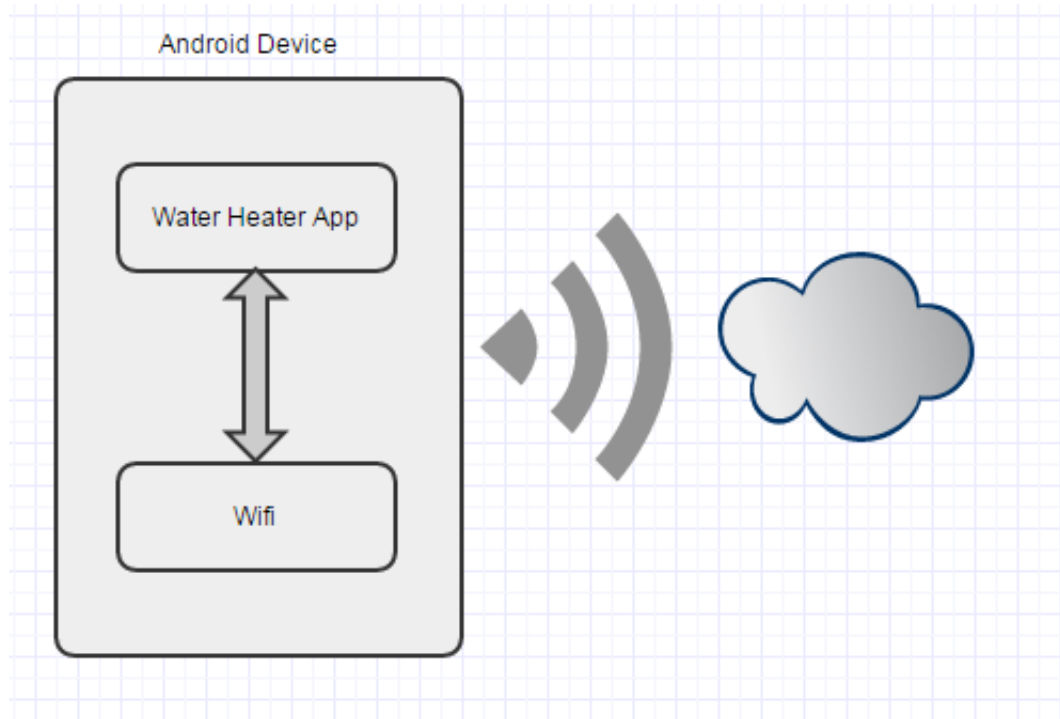
The power subsystem will have different outputs ensuring that each component is receiving a regulated voltage in its operating range.



**Figure 3.27 Block Diagram For Controller Board**

### 3.4.2. Block Diagrams

The following are block diagrams for the other components of the system



**Figure 3.28 Block Diagram Of Android Device**

The application will be running on the Android Operating system. We will specify a minimum SDK, guaranteeing that the application will run on subsequent version of android.

While we don't have direct access to drivers and system configuration, virtually all phones and tables have a Wi-Fi radio built in. Android is sufficiently abstracted that we do not need to worry about low level configurations. Simply opening a TCP port from within the application we write gives us access to the radio to send and receive pertinent data.

The Nest has its own operating system and firmware installed. Since we seek to use these stock functions, any software that we write in this environment will need to be an extension of the existing programs. Our extensions need to work in harmony with the original functions since they are what make this device attractive to us.

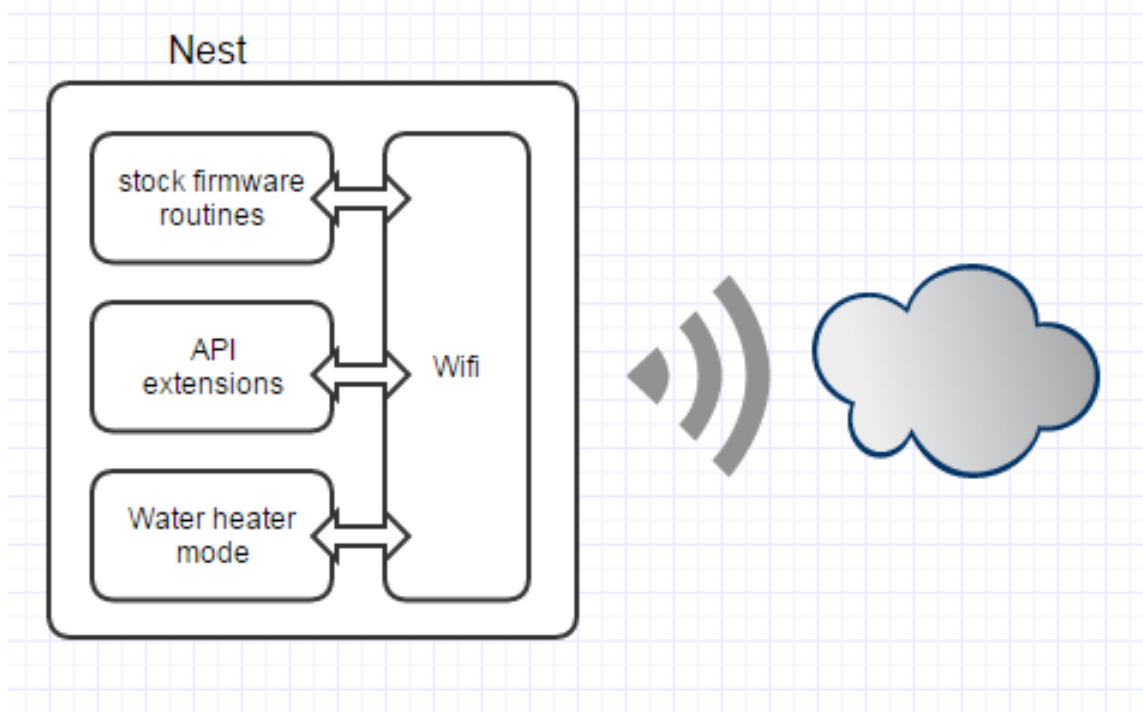


Figure 3.29 NEST Block Diagram

## 4. Project Hardware and Software Design Details

### 4.1. Process Control Board

The Process Control Board will be mounted in an enclosure to the rear of the touchscreen. The Process Control Board (hereafter known as PCB) will contain all of the subsystems of the device, such as the main control unit and various sensors and communication antennae.

The PCB should not be overtly large. Ideally, it will be smaller than the footprint of the 4.3" touchscreen that will be used. The touchscreen and enclosure should be as thin as allowable, allowing it to be attached to the exterior of a water heater tank without being a hazard to passers-by.

A mini-USB port will be required for processor programming. Two programming options exist for the currently selected main processing chip. These are an add-on daughter board containing the chip programming circuitry, or a JTAG programming cable. This device will implement a mini-USB port and the usage of

the programming cable, as the chip will not require in-field programming. This also allows the design to be kept as compact as possible.

Power and filtering requirements will require some non-SMD capacitors and connectors. With few other exceptions, most components will be SMD and therefore require little vertical space allowing for a screen-and-enclosure of less than 1 inch thick.

Due to the complexity of the Zynq-7000 SOC that will be used as the main processor, the preliminary design will utilize development board obtained for this project. After software development and sensor testing has moved beyond the preliminary development stage, a reference design provided by Xilinx will be used to design and develop a custom board for the Zynq-7000. The reference-design based custom board will be more streamlined for this particular project, removing many of the unneeded features provided by the Zynq Development board. The custom design will also be smaller in physical size to allow for the overall project design to remain as small as possible.

## 4.2. Main Control Unit

The Main Control Unit is the processing and learning center of the device. The heart of the MCU is the Zynq 7020 System on Chip. Combined with temperature sensors and a flow rate sensor, the MCU interprets data and makes decisions.

### 4.2.1. System Control

For preliminary development, the group has procured the use of a Zynq Evaluation and Development Board (hereafter known as ZedBoard). The ZedBoard has a Xilinx Zynq-7000 series AP SOC (system on chip) as the primary control chip. The ZedBoard has multiple input/output interfaces as well as onboard audio and video controls. All preliminary software development and sensor testing will be done using the Zedboard provided. As the project extends to the hardware creating phase, a custom control board utilizing the Zynq 7000 chip will be designed and constructed using the Zedboard as a reference point for capabilities.

The Zedboard specifications are tabulated below.

Zynq 7020 SOC	Dual ARM Cortex-A9 MPCore 667 MHz Operation
Memory	512 MB DDR3 256 Mb Quad-SPI Flash Full size SD/MMC card cage 4 GB SD Card

**Table 4.1 ZedBoard Specifications 1**

Connectivity	10/100/1000 Ethernet USB UART USB OTG
Video/Display	HDMI output VGA connector 128*32OLED 9 User LEDS
User Input	8 Slide Switches 7 Push Button Switches
Analog	XADC Header 4 Analog Inputs
Debug/Programming	On-Board JTAG port ARM Debug Access Port

**Table 4.2 ZedBoard Specifications 2**

The final hardware revision of the device will require the capability of interfacing with a temperature sensor bank, a flow sensing bank, element control, wireless network, and a video touch screen. Each of those elements will be detailed below.

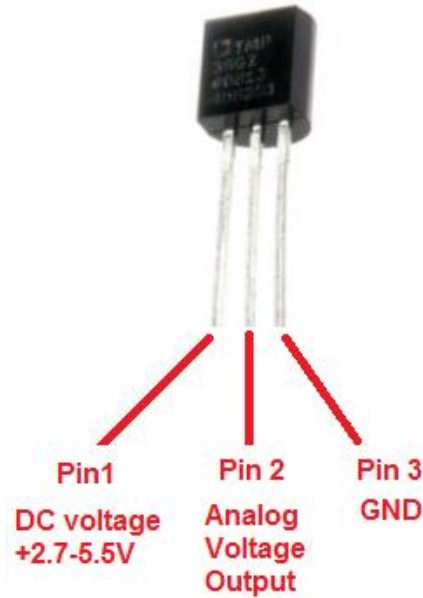
#### **4.2.1.1. Temperature Sensing**

A common upper temperature limit for most household water heaters is between 120 and 130 degrees F. A lower temperature allows for a slower buildup of minerals and corrosion, but will also pose a possible problem of bacterial growth. Higher temperatures eliminate the problem of bacterial growth, but increase the annual cost of energy usage. Temperatures over 140 degrees F increase the chances of scalding injuries.

The temperature system will continually update the control module with the current water temperature. In conjunction with a PID controller, the temperature of the water can be maintained at a preset temperature more efficiently than the standard analog controls on most water heaters.

In the first revision of the temperature sensing subsystem, the main component was a type K thermocouple. The type K thermocouple was chosen primarily for cost and ease of use. The general accuracy of most type K thermocouples is  $\pm 1$  degree Celsius. More expensive thermocouples can bring the accuracy down to a fraction of a degree; however for our usage a single degree disparity is acceptable.





**Table 4.3 TMP36 Temperature Sensor Pinout**

Type K thermocouples have nickel as one of the constituent metals, and as such will suffer from a slight accuracy degradation as the material reaches its Curie point (the temperature at which a materials magnetic properties are permanently altered) of 185 degrees Celsius. However, since this temperature is well beyond the upper range of what humans require from a water heater, this accuracy degradation will not be an issue.

Current hardware revisions utilize a TMP36 from Analog Devices. It is an analog temperature sensor that uses a solid-state method of temperature measurement. More reliable than temperature sensitive resistors or bimetallic strips, and safer than thermometers using mercury, the TMP36 is relatively easy to use. It has no moving parts and it has precision finer than all of the budget-friendly thermocouples researched.

The TMP36 functions by using voltage measurements and transistors. The voltage change across a transistors base and emitters changes as a known rate with temperature changes. Amplifying this voltage change, an analog signal that is proportional to the temperate change can be generated.

The TMP36 is available in both through-hole and surface mount packages. This device utilizes a TO-92 through-hole package due to ease of converting that package in to a waterproof on-lead style probe.

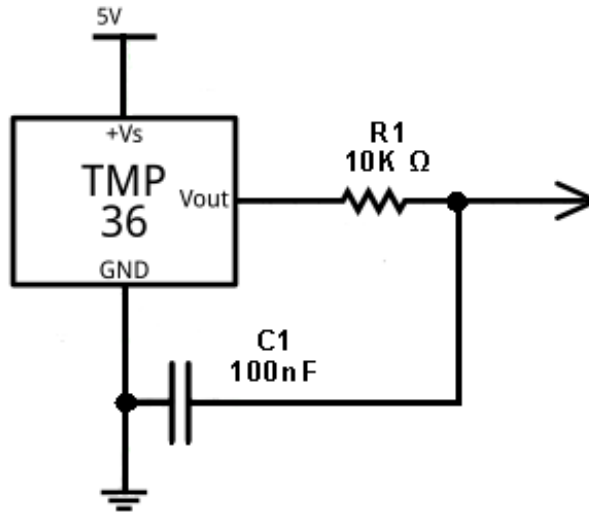


Figure 4.4 Suggested TMP36 Usage

TMP36 Specifications:

<b>Input Voltage</b>	+2.7V to +5.5V
<b>Temperature Range</b>	-40°C to +150°C
<b>Accuracy</b>	±1°C Extremes, 0.5°C Midrange
<b>Stability</b>	150°C for 1000 hours

Table 4.5 TMP36 Temperature Sensor Specs

The linear behavior of the TMP36 simplifies temperature measurements. Measuring the voltage difference of the pins and using a simple formula of;

$$Temp\ in\ ^\circ C = \frac{(Vout\ in\ mV) - 500}{10}$$

Therefore if the voltage on the Vout pin measures 650mV, the corresponding temperature is 15°C. The TMP36 holds this linearity through the majority of its operating range. Only near the extremes does it deviate and alter the accuracy of the measurements. 150° is 302°F, well above our maximum allowed temperature. The TMP36 will remain within its most accurate range for its usage in this device.

#### 4.2.1.2. Flow Sensing

The Flow Sensor subsystem will allow the control module to obtain accurate information on the amount of water used. This data will allow the learning algorithm to determine a general schedule of water usage. After a learning period, this algorithm will be able to predict when the user will need hot water and activate the heat controls.

The two main types of flow sensors are in-line and non-contact. In-line flow sensors are designed to be added to existing plumbing, allowing a sensor to be placed directly in the flow of the measured liquid. These sensors are inexpensive and reliable, but require modification to plumbing pipes if they are installed in an already existing setting.

Non-contact flow sensors use various frequencies of electromagnetic radiation to measure flow rate. Major downsides to non-contact flow meters is that they are expensive and do not work for clear or non-magnetic liquids. The non-contact flow meters require magnetic properties in the liquid, or require the liquid to have particulates that can be measured to determine flow speed of the liquid carrying them. Since this component is for a household water heater and the water must be potable, the non-contact flow meters will not work.

The flow sensor that will be utilized in this hardware revision will be a plastic-housed, in-line pinwheel style flow meter. As liquid flows through the meter, the pinwheel spins. Attached to the pinwheel is a very small magnet. The rotation of the pinwheel and magnet combo is measured by a hall effect magnetic sensor. The use of the magnet and hall effect method allows the separation of liquids from electronics.

Flow Sensor Specifications:

<b>Supply Voltage</b>	+5V to +18V
<b>Current Draw</b>	15mA at +5V
<b>Working Temp Range</b>	-25°C to +80°C
<b>Working Pressure Max</b>	2.0 MPa
<b>Mechanical</b>	½" NPS pipe connection

Table 4.6 Flow Sensor Specs



**Figure 4.7 The YF-S201 Flow Rate Sensor**

Shown above is a representative image of the flow sensor that this design will be implementing. Many Hall Effect flow sensors have the same basic design, and currently the flow sensor of choice is an YF-S201. The three wires are for power (red), ground (black), and Hall Effect pulse output (yellow).

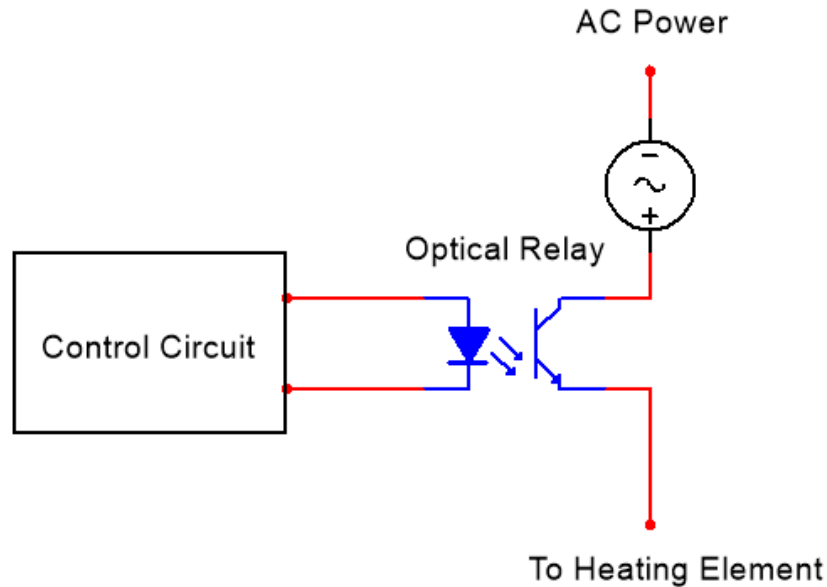
#### **4.2.1.3. Heater Control**

To control the electrical current flowing to a water heater element, most modern water heaters employ a simple thermostat. If the thermostat registers a water temperature below a pre-set threshold, then the thermostat closes a contact and allows a current flow to the element. In this project, the 'dumb' thermostat will be replaced with the designed smart alternative. To allow the device to control the current flow to the heating element, the design will employ a relay circuit.

The relay circuit will allow the device to control the high current without harming the delicate electronics of the control board. In an actual water heater, the voltage supplied is 240VAC. Due to location constraints and testing safety our preliminary design will utilize 120VAC. For this preliminary design, an Omron solid state relay of the G3NA series will be employed. The G3NA series will allow for a 5VDC control signal and can handle a 120VAC/10A load. This type of solid state relay uses an LED to signal a light sensitive MOSFET that switches the load.

Solid state relays offer multiple benefits over physical electromechanical relays. The solid state relays have a much smaller profile and increased lifetime. SSRs also eliminate the voltage spikes when electromechanical relays are switched on

and off. Solid state relays also employ no moving parts, therefore eliminating the possibility of sparking when switched on and off. Due to the closeness of the relay to static sensitive components, this is a very important feature.



**Figure 4.8 Optical Relay Usage**

The relay circuit will be kept very simple. It is only required to cycle the heating element on or off as the control unit needs. Optical relays such as the Omron G3NA series have a very fast operation time, as low as one millisecond, so this series of optical relays will be well beyond the operating speed that will be required.

#### **4.2.1.4. Heat Pump Control**

Heat pump type water heaters work in slightly different method to the same results as a normal element type water heater. Heat pump water heaters operate by using small amounts of energy to extract heat from the air surrounding the water heater and transferring that heat to the water inside the tank. One downside to heat pump style water heaters is that, like their home heating and cooling counterparts, they do not handle extremes well. Therefore, to handle times when there is high demand for hot water, heat pump water heaters will include standard electrical heating elements. This makes most heat pump water heaters a hybrid of sorts, with the less efficient electrical elements only used sparingly.

Heat pump style water heaters are a recent innovation, and many include digital controls on the tank. These controls are used to set the operation mode of the water heater, or put it in a 'vacation' mode so that it powers down and conserves

energy while it is not needed. The proprietary nature of these controls make interfacing with a third party device more difficult, but not impossible. Depending on the software used in the controls of the hybrid water heater, a third party device may coexist or replace existing controls.

Given the scale and time allotment of this design class, including every different hybrid water heater is not feasible. Therefore, this device will be designed with replacement of the existing controls as the goal. This eliminates many variables of the coexisting option, and will expand the hybrid water heater's capabilities.

Hybrid water heaters generally have multiple operating modes.

- High Efficiency Mode – utilizing only the heat pump subsystem for heating the water
- Hybrid Mode – uses both the heat pump subsystem in conjunction with the standard electrical heating elements
- Electric Only Mode – generally used in high demand situations, the least efficient setting
- Vacation Mode – This mode puts the unit in a hibernation setting, no longer maintaining a minimum temperature and suspending energy usage until the occupants return

Controlling these subsystems will only be slightly more involved than a standard electric water heater. At the very basic level, each subsystem is still controlled by a simple on/off setting. Our device will control the power to each of these subsystems, and the interface will allow hybrid water heater owners to set the operating mode just as the default digital controls.

To demonstrate the ability of our device to control a heat-pump style water heater, the testing hardware will include a small fan, similar to those used as case fans in electronics or personal computers. The control unit will switch the fan on to show that it has activated the heat pump system for Hybrid or High Efficiency modes.

#### **4.2.2. Communications and Intelligence**

Our water heating system will consist of several devices physically separated. Smart decisions will have to be made according to data, instructions and conditions coming from any of the components whether it is the thermostat, android device or the water tank. In order to avoid problems associated with distributing processing and decision making we need to relay all the appropriate data to a central node and have this be the brain of the unit. Reliable communication links are imperative for the system to work properly.

#### **4.2.2.1. Operating System**

Our board design will be based around the Zynq-1000 System on Chip which is capable of natively running different flavors of Linux. We felt it would be appropriate to operate under a proper operating system as opposed to dedicated routines on a microcontroller because of the many different tasks we need to perform such as reading sensors, radio, and perhaps more importantly, running a web server at the same time.

It is important to note that the Nest itself also runs its own version of a Linux operating system.

##### **4.2.2.1.1 Linux**

Strictly speaking, Linux is a kernel and not an operating system. A kernel is the backbone of an operating system, interfacing the software (or applications) with the hardware. It manages input/output, memory, processes, etc. but it does not provide much directly to the user. A proper operating system uses a kernel coupled with a shell that adds a user interface.

The Linux kernel was initially created in 1991 by Linus Torvalds partly as an exercise in curiosity but also to fill some gaps in MINIX, the operating system he was using at the time. The kernel on its own was not enough though. It was not until Linux paired up with GNU, an operating system that lacked a solid kernel, that the Linux Operating System (more specifically GNU/Linux) became a more viable solution to more average users.

The operating system we chose is Debian, a linux distribution, because we have the most experience under this particular environment. Debian is very well documented and has a very active community which will help us and surely speed up development time.

##### **4.2.2.1.2 File System**

The Linux file system generally follows a standard directory structure. The main directories we will be concerned about are:

- /bin – this is where system utilities usually reside such as text editors, copy and move functions, etc. What we can do directly on the system while it is running will largely depend on the applications installed in this folder
- /dev – this directory contains files that represent the hardware in our system. Through these files we have logical access to the peripherals
- /etc – this is where we keep and edit all our system's configurations such as network addresses, startup programs, etc.

- /home – we can use this directory for general storage purposes.

#### 4.2.2.1.3 Applications

The software suite to enable all the planned functionality will be comprised of several commonly used applications paired with our own algorithms and implementations to learn occupancy and usage patterns.

#### 4.2.2.1.4 Web Server

Django is a python-based web framework that will allow us to create a web application. With Django we will not only be able to send, receive and parse HTTP requests, but we will also be able to directly make changes to the database. Using Django as an extension of python, we can treat tables in the database as classes and rows as objects, giving us a great deal of flexibility with the data.

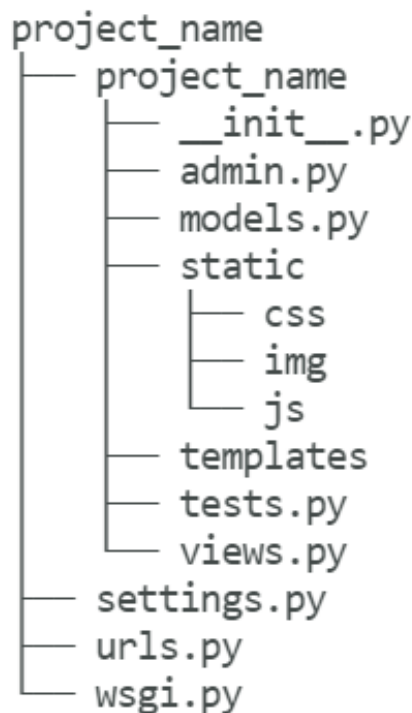


Figure 4.9 Project Layout

A usual, albeit very basic Django project is laid out as above. Django aims to speed up web development by taking care of as much as possible on the backend. Django is especially useful on database driven websites or applications.

Some of the most interesting files in this base project are:



- models.py – In this script we can set up a database by creating classes for each table. The classes are not database specific, and will translate to any database we set up on the backend (MySQL, PostgreSQL, SQLite, Oracle)
- views.py – this script is responsible for taking in requests and sending out the appropriate response, like serving a website. It is from here that we can interface to our own scripts and algorithms to control the hardware from data we receive over wifi.
- manage.py – we cannot modify this file, but it is a sort of Django Swiss Army knife. With this script we are able to run the server, change server settings such as port and we can even make changes to the database and synchronize old content to the new schemas.

Our heater will have to communicate to other devices using Wi-Fi mainly through HTTP (Hypertext Transfer Protocol) requests. For this reason we will need a webserver capable of handling the backend of data transfer. Although there are more lightweight choices, we will be using Apache as our webserver mainly because it is recommended by Django.

Our heater will have to communicate to other devices using Wi-Fi mainly through HTTP (Hypertext Transfer Protocol) requests. For this reason we will need a webserver capable of handling the backend of data transfer. Although there are more lightweight choices, we will be using Apache as our webserver mainly because it is recommended by Django.

#### **4.2.2.2. NEST API**

The data structure for the nest is fairly simple. All the information is encoded into a single data object using JSON. This data object is split into three sections: Metadata, thermostats and structures. 'Metadata' is authentication information which we don't care about. 'Thermostats' has typical thermostat information such as target temperature, hysteresis, current temperature, mode, etc. and 'Structures' contains information about the house itself.

Thermostats and Structures could be arrays of multiple nests in the house. Each member in the array could potentially have different data depending on how they are set up. The API does not directly say whether the AC (or heater) is on, but that can be easily extrapolated from the current temperature and target temperatures over a period of time. If we poll enough times over the day, our heater should be able to learn when the AC is expected to come on or not, and from there extrapolate when the water heater should be on or not.

Under structures we can find the "away" field which can be manually set by the user, or can be set to auto. No matter how the user sets it, the end functionality is the same. We could check the away field against the current AC mode (on- off) to better learn occupancy and make more robust choices of water heater usage.

Below is the data structure for the NEST:

```
{
  "metadata": {
    "access_token": "c.FmDPkzyzaQe..." ,
    "client_version": 1
  } ,
  "devices": {
    "thermostats": {
      "peyiJNo0IldT2YIIVtYaGQ": {
        "device_id": "peyiJNo0IldT2YIIVtYaGQ" ,
        "locale": "en-US" ,
        "software_version": "4.0" ,
        "structure_id": "VqFabWH21nwVyd4RWgJgNb292wa7hG_dUwo2i2SG7j3-
        BOLDY0BA4sw" ,
        "name": "Hallway (upstairs)" ,
        "name_long": "Hallway Thermostat (upstairs)" ,
        "last_connection": "2014-10-31T23:59:59.000Z" ,
        "is_online": true ,
        "can_cool": true ,
        "can_heat": true ,
        "is_using_emergency_heat": true ,
        "has_fan": true ,
        "fan_timer_active": true ,
        "fan_timer_timeout": "2014-10-31T23:59:59.000Z" ,
        "has_leaf": true ,
        "temperature_scale": "C" ,
        "target_temperature_f": 72 ,
        "target_temperature_c": 21.5 ,
        "target_temperature_high_f": 72 ,
        "target_temperature_high_c": 21.5 ,
        "target_temperature_low_f": 64 ,
        "target_temperature_low_c": 17.5 ,
        "away_temperature_high_f": 72 ,
        "away_temperature_high_c": 21.5 ,
        "away_temperature_low_f": 64 ,
        "away_temperature_low_c": 17.5 ,

```

Figure 4.10 NEST Data Structure 1

```

" hvac_mode": "heat" ,
" ambient_temperature_f": 72 ,
" ambient_temperature_c": 21.5 ,
" humidity": 40
}
} ,
" smoke_co_alarms": {
" RTMTKxsQTCxzVcsySOHPxKoF4OyCifrs": {
" device_id": "RTMTKxsQTCxzVcsySOHPxKoF4OyCifrs" ,
" locale": "en-US" ,
" software_version": "1.01" ,
" structure_id": "VqFabWH21nwVyd4RWgJgNb292wa7hG_dUwo2i2SG7j3-
BOLY0BA4sw" ,
" name": "Hallway (upstairs)" ,
" name_long": "Hallway Protect (upstairs)" ,
" last_connection": "2014-10-31T23:59:59.000Z" ,
" is_online": true ,
" battery_health": "ok" ,
" co_alarm_state": "ok" ,
" smoke_alarm_state": "ok" ,
" is_manual_test_active": true ,
" last_manual_test_time": "2014-10-31T23:59:59.000Z" ,
" ui_color_state": "gray"
}
}
} ,
" structures": {
" VqFabWH21nwVyd4RWgJgNb292wa7hG_dUwo2i2SG7j3-BOLY0BA4sw": {
" structure_id": "VqFabWH21nwVyd4RWgJgNb292wa7hG_dUwo2i2SG7j3-
BOLY0BA4sw" ,
" thermostats": [ "peyiJNo0IldT2YIIVtYaGQ" , ... ] ,
" smoke_co_alarms": [ "RTMTKxsQTCxzVcsySOHPxKoF4OyCifrs" , ... ] ,
" away": "home" ,
" name": "Home" ,
" country_code": "US" ,
" postal_code": "94304" ,
" peak_period_start_time": "2014-10-31T23:59:59.000Z" ,

```

Figure 4.11 NEST Data Structure 2

```
"peak_period_end_time": "2014-10-31T23:59:59.000Z" ,
"time_zone": "America/Los_Angeles" ,
"eta": {
"trip_id": "myTripHome1024" ,
"estimated_arrival_window_begin": "2014-10-31T22:42:59.000Z" ,
"estimated_arrival_window_end": "2014-10-31T23:59:59.000Z"
}
}
}
}
```

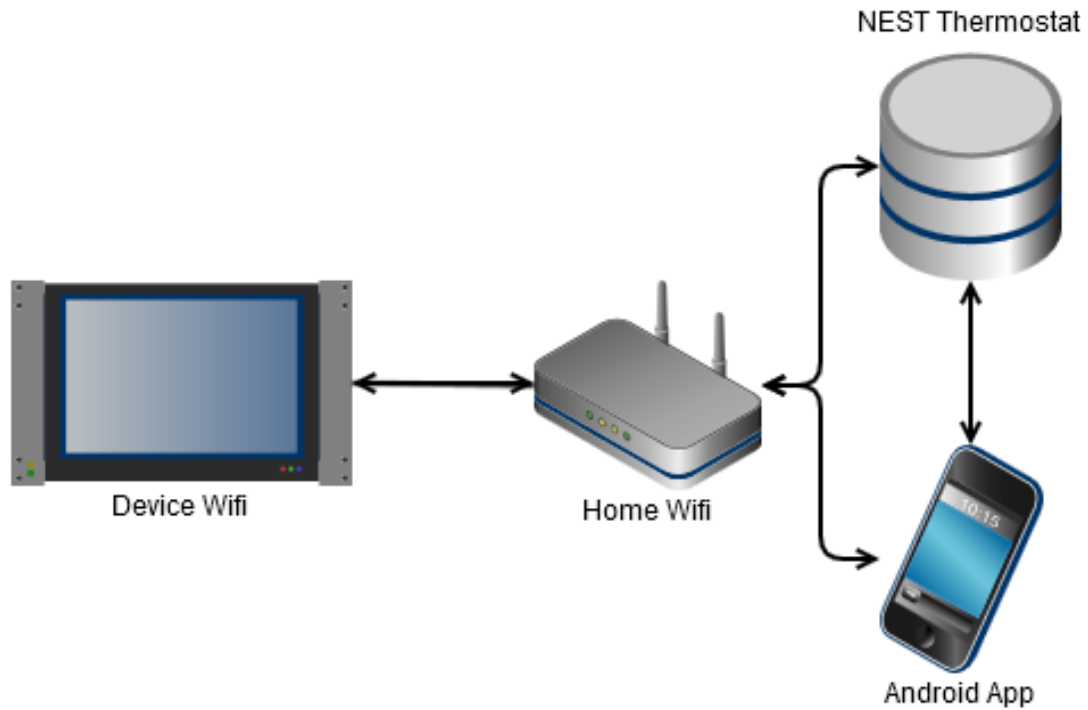
**Figure 4.12 NEST Data Structure 3**

All the information available from a Nest is in the above data object. This information can theoretically only be accessed from their cloud services. As a result we are left with two options. We can store nest credentials on our controller board and go through the cloud to poll for information, or extend the firmware to echo this same data to an internal network address and talk to the water heater directly.

In the even that we end up skipping their cloud services all together, the first plan of action will be to simply echo the POST request to the specified address of the water heater. We do not want to interfere with the expected functionality of the thermostat, and a key function is its communication with the Nest servers to orchestrate control from the mobile devices. By sending the packet twice, once to the Nest servers as usual, and then to our water heater all functionality should be unaffected. After verifying this works, it may be worthwhile to create a specialized JSON object to be used specifically with our system. Out of the 50+ fields that form part of the JSON object being sent out, we only care about a few of them, and some other properties such as current usage are not advertised at all. If we can be more explicit while reducing overhead it will be beneficiary to us in the long run.

#### **4.2.2.3. WIFI**

The wireless subsystem of this device will consist of an XBee-Pro Series 1 wireless module. This particular module has a trace antenna, keeping the overall size of the module extremely small.



**Figure 4.13 Wireless Communication Routes**

The wireless subsystem of this device will consist of an XBee-Pro Series 1 wireless module. This particular module has a trace antenna, keeping the overall size of the module extremely small.

Multiple protocol support and a 2.4GHz operating frequency insure compatibility with home networking systems. The encryption support maintains home network security integrity.

XBee Module Specifications:

<b>Supply Voltage</b>	+2.8V to +3.4V
<b>Transmit Current</b>	250mA at +3.3V
<b>Idle/Receive Current</b>	55mA at 3.3V
<b>Power-Down Current</b>	<10 $\mu$ A
<b>Operating Frequency</b>	ISM 2.4GHz
<b>Indoor Range</b>	Up to 300 feet
<b>Data Rate</b>	250kbps

**Table 4.14 XBee Specifications**

The pinouts for both the XBee and XBee Pro modules are the same. Since the XBee module itself will be connected to the control unit via headers, in the event of a malfunction or module exchange, there will not be any need to have any of the PCB reworked. Replacement of the module would be a simple plug and play operation.

Pinout data for the XBee modules tabulated below.

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Out	UART Data Out
3	DIN	In	UART Data In
4	D08	Out	Digital Output 8
5	Reset	In	Module Reset
6	PWM0	Out	PWM Output 0
7	PWM1	Out	PWM Output 1
8	[reserved]	-	Do not connect
9	SLEEP_RQ	In	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground

**Table 4.15 Pinout data for XBee 1**

Pin #	Name	Direction	Description
11	AD4/DIO4	Either	Analog Input 4 or Digital I/O 4
12	CTS/DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	On/Sleep	Out	Module Status Indicator
14	VREF	In	Voltage Reference for A/D Inputs
15	AD5/DIO5	Either	Analog Input 5 or Digital I/O 5
16	AD6/DIO3	Either	Analog Input 6 or Digital I/O 6
17	AD3/DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2/DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1/DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0/DIO0	Either	Analog Input 0 or Digital I/O 0

**Table 4.16 Pinout data for XBee 2**

Some XBee module notes from Digi international;

- Minimum connections are VCC, GND, DOUT, and DIN
- Signal direction is specified using the module as reference
- There is a 50KΩ pull-up resistor attached to RESET

Pins unused by the design should be left disconnected, not attached to ground.



Figure 4.17 XBee Pro S1

### 4.3. NEST Interface

The Nest User Interface will play a secondary role in the system, but nonetheless we hope to be able to implement it cleanly. The only two ways a user can interface and interact with the thermostat are the 320 x 320-pixel color LCD and a rotating outer ring that can also be clicked. Under normal operation, the temperature can be adjusted by rotating the wheel. If the system is actively cooling, the background turns blue. If the system is actively heating, the background will turn orange. By pressing the ring, one can enter the menu in which different settings and metrics can be accessed.

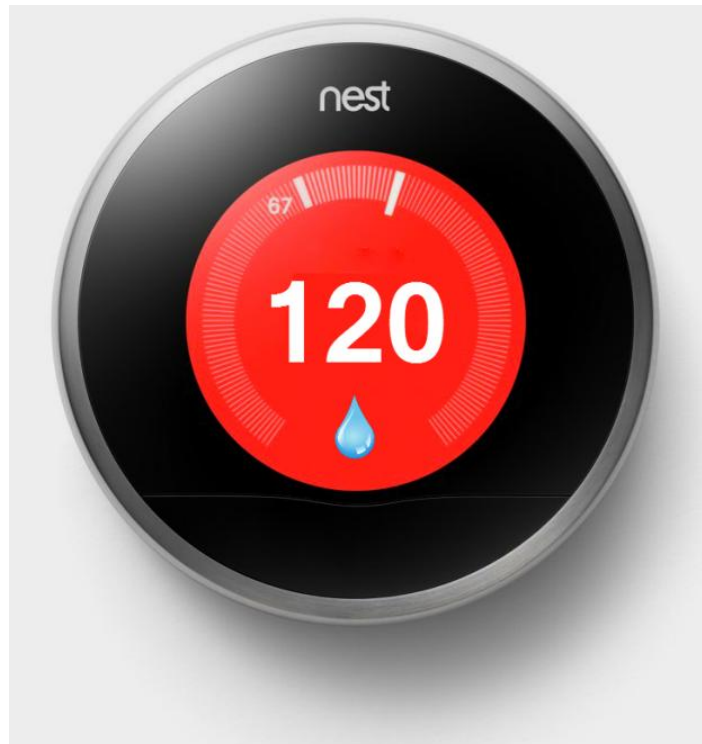
#### 4.3.1. User Interface

Using the same basic concepts, we are going to extend the functionality of the thermostat to allow for some basic control of the water heating system.

We will offer an additional mode to the already existing Heating and Cooling by adding a 'Water' mode. Once we have access to the source code we will make a more informed decision on the specifics of accessing this new mode, but it will either be through a double-click, or a long click. Trying to keep the interface consistent, we will show target temperature as well as activity. The user will need

a way to quickly know these factors, but also know that the thermostat is in the new water mode. Without an obvious visual clue the user could get confused about the unusually high temperature setting.

The thermostat sometimes shows a 'Nest Leaf' logo under the target temperature to symbolize Energy Savings. Similarly, we could employ a water drop (or anything water related, for that matter) to show that we are in the Water Heater mode. We could use a red background to denote the system is actively heating, in which case the blue drop would sufficiently contrast.



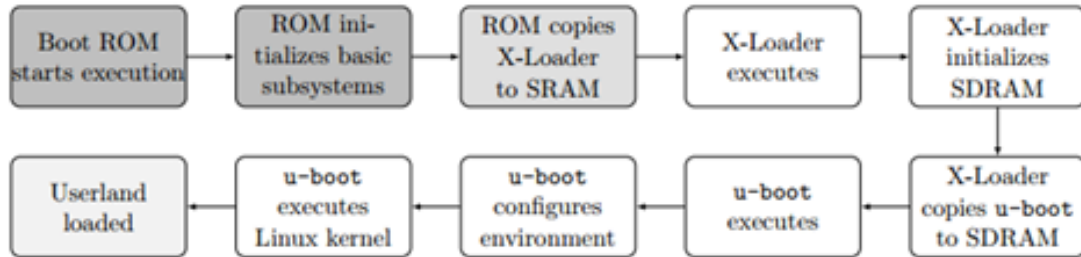
**Figure 4.18 Mock-up Of Possible Water Heater Mode**

#### **4.3.2. Rooting nest**

As previously mentioned, some of the functions on our system will depend on having access to the nest not just physically, but also the firmware running inside of it. In order to modify or extend the existing firmware we will need to have root access to the device so we can run our own code. This is generally not an easy feat; End users are generally not expected, or even allowed in many cases, to extend the device's functionality because they run the risk of breaking the device in question. However, many times there will be back doors that facilitate control to the device. These back doors are normally used by the



manufacturer to set up the device before it is shipped or to diagnose and repair the device if it should be returned.



**Figure 4.19 Standard Nest Thermostat Boot Process**

The normal booting process can be interrupted, and due to a bug (or overlook) a new firmware can be installed onto the Nest through the USB port in the back. This new firmware will allow us to make changes.

By holding down the ring for ten seconds, the processor will enter the DFU (Device Firmware Update) mode.

“ This mode allows a user to push a set of images and addresses to be loaded through the device’s USB port with a utility called “omap3\_loader”. DFU mode is only intended as a catalyst to load the next stages of code, the first of which in our case is the x-loader binary. X-loader is a stage 1 boot-loader that is used on the Nest as the initial loading point for the system. X-loader handles getting the device ready to execute the second stage boot-loader that is responsible for loading up the Linux kernel. On the Nest, the second stage boot-loader is an open source piece of software widely used on embedded devices known as “U-Boot”. We use our own custom modified version of U-Boot that is based on the GPL released Nest version to boot a Linux kernel.”

[http://wiki.gtvhacker.com/index.php/Exploiting\\_Nest\\_Thermostats](http://wiki.gtvhacker.com/index.php/Exploiting_Nest_Thermostats)

We will also need a toolchain to cross-compile our code before we can install it to the Nest. Fortunately this is also provided. Once our cross compiling environment is set up we will be able to start working directly on the device.

#### 4.4. Touch Screen Interface

Current hardware revisions have the device using a low-cost Sunbond LB04302 4.0 inch thin-film transistor liquid crystal display (TFT LCD) screen with a resistive touch screen overlay.

- 4.3" display at 480\*272 resolution
- -20°C to 70°C operating temperature range
- 25ms response time

The view angles of the LB04302 screen are 110° horizontal and 90° vertical. These viewing angles are very acceptable for a low cost screen. This insures that the screen is viewable even if the device has to be placed in a less than optimal condition, as many end users would be unlikely to reposition an entire water heater for the sake of viewing.

The LB04302 is a raw pixel-dot-clock display and does not include an onboard driver or SPI/parallel controller. If used with a less powerful processor or microcontroller, this LCD would require an additional driver board or circuit. However, due to the high capabilities of the Zynq 7000 series SOC that will be used, any additional hardware for the display will be limited to power supply only.

The LB04032 display has a 40-pin ribbon connector that includes an 8-pin red, an 8-pin blue, and an 8-pin green data bus for 24-bit color. There are also four pins set aside for the resistive touch overlay. Comprehensive pin listing is tabulated below.

Pin #	Name	Description
1	VLED-	Backlight power -
2	VLED+	Backlight power +
3	GND	Ground
4	VDD	Power supply, 3.3V
5-12	R0-R7	Data Bus Red
13-20	G0-G7	Data Bus Green
21-28	B0-B7	Data Bus Blue
29	GND	Ground
30	PCLK	Dot-Clock signal, Oscillator source
31	DISP	Display On/Off
32	HSYNC	Line Sync signal
33	VSynd	Frame sync signal
34	DE	Display Enable
35	NC	Not Connected
36	GND	Ground

Table 4.20 Sunbond Touch screen Pinout 1

Pin #	Name	Description
37	XR	Touch for X axis, Right
38	YD	Touch for Y axis, Down
39	XL	Touch for X axis, Left
40	YU	Touch for Y axis. Up

**Table 4.21 Sunbond Touchscreen Pinout 2**

The programming for the touch screen will be done in Python using the PyQt framework. Originally Qt was created as an extension to C++ but has been ported to Python. We will use the Python port to keep the languages as consistent as possible over all the different programming platforms.

The main features of Qt are:

- Qt is intended to be cross-platform. It can be interchangeably used in all common platforms such as Windows, OSX, Android, iOS and X11. Most importantly for us though, it can also be used on embedded linux devices that may not have a native graphical user environment. In our case we will only be running one graphical application. Qt will make use of a framebuffer that will not require a window system. This also means that we do not need to write platform specific code. All graphical functions are abstracted and as such the same code we write for a windows environment, for example, will work on an embedded Linux environment provided we have the right compiler.
- It uses a signals and slots systems. Objects can emit signals as well as receive signals from other objects through slots. This functionality is not native to C++ and can help making code more concise. Having explicit signals trigger methods instead of pointers also reduces the chances of encountering bugs.

## **4.5. Power**

As our device will be integrated into existing water heaters, it will need a power source. Most water heaters utilize 240VAC, and standard 120VAC outlets are normally not located within a convenient radius. Therefore, the device will utilize the 240VAC of the water heater by using a transformer and subsystem to step down the voltage to an electronics-friendly 12VDC. A filter circuit will also be included to eliminate any line noise and voltage fluctuations, as these would be detrimental to the sensitive chipset used for the main control system.

The power subsystem will be supplying multiple subsystems requiring various voltages and currents. The main component power requirements are below.

Touch screen Display:

The power specs for the touchscreen display are listed in the table below. The touchscreen itself requires power for the backlights and the resistive touch overlay, hence the multiple input voltage requirements.

	<b>V Typical</b>	<b>V Range</b>
<b>Logic VDD</b>	3.3V	-0.5V to +5V
<b>Analog VDDA</b>	5.0V	-0.5V to +7.5V
<b>LED High( VGH)</b>	+15V	+9V to +16V
<b>LED Low (VGL)</b>	-10V	-9V to -11V

**Table 4.22 Voltage Needs: Display**

Flow Sensor:

The power specs for the flow sensor are listed in the table below. The flow sensor has a very wide acceptable input voltage range, due mostly to the relatively simple electronics found inside it. The current draw at 18VDC is approximately 50mA, but for this design it the flow sensor will be utilized at the +5VDC input voltage to allow it to easily use an already existing 5VDC power node.

<b>Working Voltage</b>	+5V to +18V
<b>Max Current Draw</b>	15mA at +5V

**Table 4.23 Power Specs: Flow Sensor**

Xbee Wireless Module:

The XBee wireless module power specs are listed in the table below. Second only to the Zynq 7000 series All Programmable System on Chip, the XBee is highly sensitive to power fluctuations. It is important that the XBee receives the needed current for transmission, otherwise data transfer to the application and optional NEST thermostat will be lost. Lost data could cause issues with the learning algorithm's schedule making ability, and also make historical power usage charts inaccurate. If software development time allows, the Zynq chip program will store data in the event of a failed transfer and continue attempts to transfer until successful.

<b>Supply Voltage</b>	+2.8V to 3.4V
<b>Transmit Current</b>	250mA at +3.3V
<b>Idle Receive Current</b>	55mA at +3.3V
<b>Power-Down Current</b>	<10 $\mu$ A

**Table 4.24 Power Specs: XBee Module**

Temperature Sensor:

The temperature sensor input voltage operating range is +2.7VDC to +5VDC. Anecdotal evidence supports that TMP36 temperature sensor can operate at a voltage lower than +2.7VDC, however this design is not so stringent on power requirements that the TMP36 will be voltage starved. As this design will have multiple components able to use or requiring +3.3VDC, the TMP36 will likely be operated at that voltage input.

Zynq 7000 Series All Programmable System on Chip:

The Zynq 7000 series All Programmable System on Chip has multiple power requirements. It is a rather complex integrated circuit and highly sensitive to power requirements. It is very important that the power system utilize storage and ripple smoothing elements to make certain that the voltage inputs to the Zynq 7000 series have exactly what they need at all times.

	<b>V Typical</b>	<b>V Max Ranges</b>
<b>Processing System (PS)</b>	--	--
<b>PS Vcc Internal Logic</b>	+1.0V	-0.5V to +1.1 V
<b>PS Vcc Auxillary</b>	+1.80V	-0.5V to +2.0V
<b>PS DDR I/O</b>	+1.14V to +1.89V	-0.5V to +2.0V
<b>PS Vpref Reference</b>		-0.5V to +2.0V
<b>Programmable Logic (PL)</b>	--	--
<b>PL Internal Supply</b>	+1.0V	-0.5V to +1.1V
<b>PL Auxillary</b>	+1.80V	-0.5V to +2.0V
<b>PL RAM Block Supply</b>	+1.0V	-0.5V to +1.1V
<b>PL Vref Reference</b>		-0.5V to 2.0V

**Table 4.25 Voltage Needs: Zynq 7000 Series SOC**

### 4.5.1. Voltage Step-down Subsystem

The voltage step-down subsystem will take the high voltage AC of the existing appliance and convert it to a lower-voltage DC that the sensitive electronics can utilize.

*Note:* For the purpose of safety and available resources, the device will be using 120VAC instead of the 240VAC supplied to an actual water heater. A 240VAC outlet will not be available during testing, and the test-bed being used will be a 120VAC appliance, such as a hot-plate. For a device connected to an actual water heater, some components in the voltage step-down subsystem would need to be exchanged for their 240VAC counterparts.

The power system will utilize a step-down transformer and rectification circuit to convert from the 120VAC to usable DC voltages. A transformer-less AC/DC converter was considered, but the lack of electrical isolation posed safety concerns regarding testers and equipment. As such, the AC/DC converter will not be used.

According to data sheet the temperature sensor is at its most accurate at +3.3VDC. The Logic VDD of the display and the Xbee wireless module are both able to utilize +3.3VDC. In addition to +3.3VDC, the display also needs +5VDC, +15VDC, and -10VDC inputs. The flow sensor also operates at +5VDC to +18VDC, and will likely be utilized at +5VDC in order to keep the power subsystem as simple as possible.

A transformer capable of 120V to 15V/24V would suffice. Coupled with the complementary circuits and voltage regulators, the various voltage inputs can be provided to the hardware in need.

The preliminary power circuit schematic is below. This circuit will be tweaked and adjusted in future revisions.

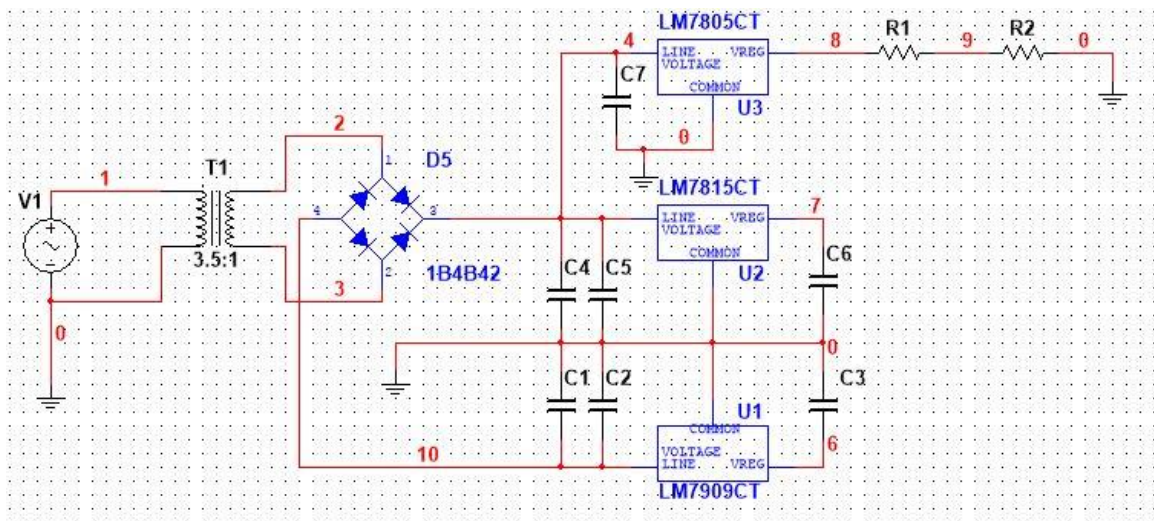


Figure 4.26 Power Circuit 1

Referring to the graphic above, node 8 provides a regulated +5VDC. Node 9 provides +3.3VDC, while nodes 6 and 7 provide -9VDC and +15VDC respectively.

Component Info for schematic

C1 = 2200 $\mu$ F	C4 = 2200 $\mu$ F	C7 = 2200 $\mu$ F
C2 = 100nF	C5 = 100nF	R1 = 500 $\Omega$
C3 = 220nF	C6 = 220nF	R2 = 1000 $\Omega$

**Table 4.27 Component Values, Power Circuit 1**

Initial simulation in Multisim 12.0 shows that the voltages at each node reach expected levels with a relatively clean signal. However, the simulation is using a generic ideal transformer with zero losses or leakages. Likewise, the rectifier bridge is also a generic choice without any negative characteristics modeled for the simulation.

Bill of Materials: (Through-hole specifications listed in table, for testing purposes)

Name	Type	Value	Qty	Price (USD)
LM7805CT	Lin. Regulator	+5VDC out	1	.62
LM7815CT	Lin. Regulator	+15VDC out	1	.67
LM7909CT	Lin. Regulator	-9VDC out	1	.61
HD04-T	Dac Diode Bridge	-	1	.12
T1	Transformer	-		*
C1, C4, C7	Capacitor	2200 $\mu$ F,50V	3	1.21
C2, C5	Capacitor	100nF, 24V	2	.18
C3, C6	Capacitor	220nF,50V	2	1.09
R1	Resistor	500 $\Omega$	1	.10
R2	Resistor	1000 $\Omega$	1	.10
				~8.39 *

**Table 4.28 Bill Of Materials, Power Circuit 1**

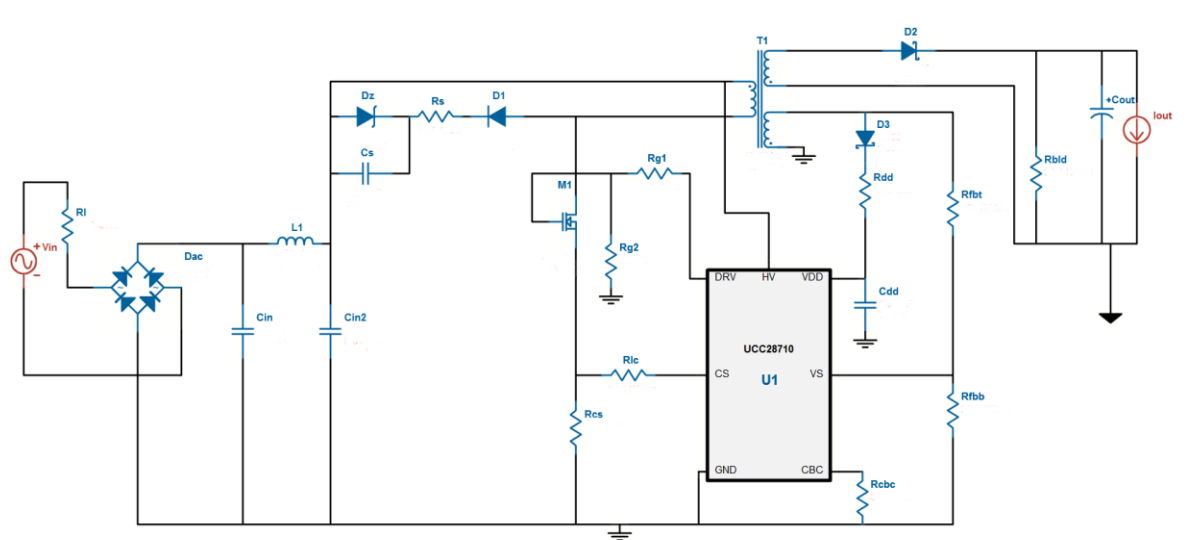
\*exact transformer price not included in BoM, likely to be approximately 4.00 to 6.00 USD. I also have a very large number of various transformers that will be searched for compatible transformers before any purchases are made.

The bill of materials for the circuit containing surface mount components should be relatively close to the same amount. Through hole components were quoted in the above table due to the ease of prototyping of through hole components versus surface mount components.

**Alternate Voltage Step-Down System Design:**

An alternate voltage subsystem that is also under consideration is a design that was created utilizing Texas Instruments Webench design tool. This design steps away from using the older technology of the linear voltage regulators and uses a Texas Instruments UCC28710 flyback power supply controller. While this circuit has more individual components, the overall size of the circuit is slightly smaller; approximately 3200 mm<sup>2</sup> compared approximately 4100 mm<sup>2</sup> needed for the circuit containing the LM series linear regulators. The usage of smaller capacitors (in both size and Farad rating) and the fact that the UCC28710 is much smaller than each of the surface mount versions of the LM series regulators (7mm<sup>2</sup> for the UCC28710 vs 42mm<sup>2</sup> for the LM series allows for a more compact size.

Each circuit has a relatively inexpensive bill of materials, so both will be simulated virtually and also tested physically to see which will perform more satisfactorily.



**Figure 4.29 Power Circuit 2**

Like the first power system circuit, the BoM for this circuit will have the through-hole counterparts to the smd parts used in the Webench designed circuit.



Bill of Materials:

Name	Type	Value	Qty	Price (USD)
Cdd	Capacitor	1.0 $\mu$ F	1	.05
Cin, Cin2	Capacitor	4.7 $\mu$ F	2	1.83
Cout	Capacitor	390 $\mu$ F	1	.63
Cs	Capacitor	1.0 nF	1	.02
D1	Diode	VF=1.3V	1	.09
D2	Diode	VF=500mV	1	.14
D3	Diode	VF=850mV	1	.21
Dac	Diode Bridge	VF=1.0V	1	.12
Dz	Diode, Zener	-	1	.11
L1	Inductor	470 $\mu$ H	1	.26
M1	Mosfet	-	1	.41
Rbld	Resistor	30.1k $\Omega$	1	.01
Rcbc	Resistor	309.0 k $\Omega$	1	.01
Rcs	Resistor	950.0 m $\Omega$	1	.1
Rdd	Resistor	22.0 $\Omega$	1	.01
Rfbb	Resistor	25.5 k $\Omega$	1	.01
Rfbt	Resistor	191.0 k $\Omega$	1	.01
Rg1	Resistor	10.0 $\Omega$	1	.01
Rg2	Resistor	10.0 k $\Omega$	1	.01
RI	Resistor	5.0 $\Omega$	1	.01
Rlc	Resistor	1.96 k $\Omega$	1	.01
Rs	Resistor	97.6 $\Omega$	1	.01
T1	Transformer	.149 Ns to Np	1	*
U1	Switcher	-	1	.41
				~4.48 *

**Table 4.30 Bill Of Materials, Power Circuit 2**

As in the first power circuit, the transformer is not included in the bill of materials price.

#### 4.6. Android Application

For this project, the design for android application is intended to give the user the convenient way to control and be aware of the provided data from the system.

### 4.6.1. Application Design

The android application is part of what makes this project "smart." It gives the user the sense of "connection" to the water heater. The functionalities of the android application are mainly consisting of the control of water heater and data access to the database that collects the information from the NEST.

As for Data Control. the application will have the access to the database and be able to retrieve the appropriate information. The information stored in database will include temperature of the tank and energy usage with time stamp at any giving time. With this information, the user will be able to see the statistical analysis information of the water heater. As for Water Heater Control, the user will be able to turn on and off the heater from the android application. User can also control the temperature and set it to the desire value within the specific limit. Below is the class diagram of the functionalities.

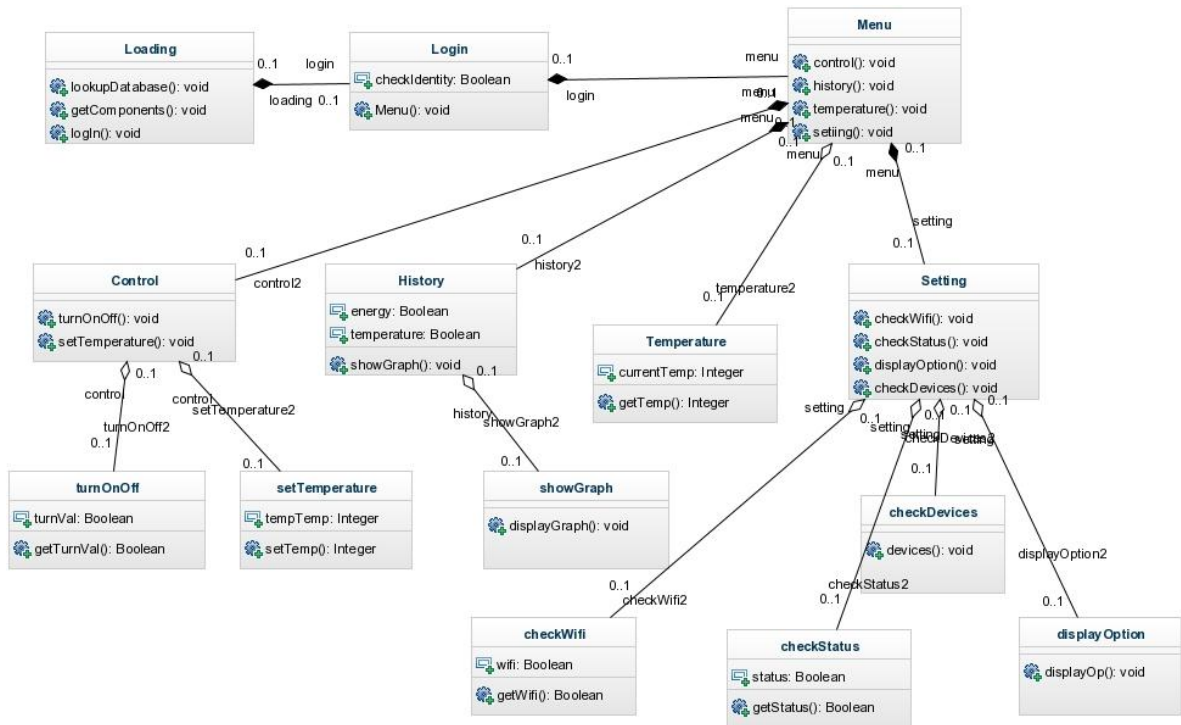


Figure 4.31 Class Diagram of Application Functionalities

As shown in the class diagram above, user will first encounter the loading screen once the application starts. After the application successfully gather the data it needs to start the program, it will bring the user to the login screen. In the log in screen, the user will be prompted to type in their user information. In order to use

the system, each user will be assigned his own log in information for the security purpose. Once the user successfully put in the appropriate log in information, the application will bring the user to the main menu screen which user can find all the available functions of the application.

The android application will consist of four main functions in the main menu screen. The first function is called "control." In the control function, the user will have control over the water heater tank and the temperature of the water heater. The user will be able to turn on or off the heater. When the user choose to turn on or turn off the water heater, the application will send the signal to the main tank. Then the tank will response to the signal and turn on or off respectively. The user will also be able to set the water to the desire temperature. When the temperature is set by the user, the system is expected to set the temperature to the desire value within certain amount of time after the signal has been sent to the heater.

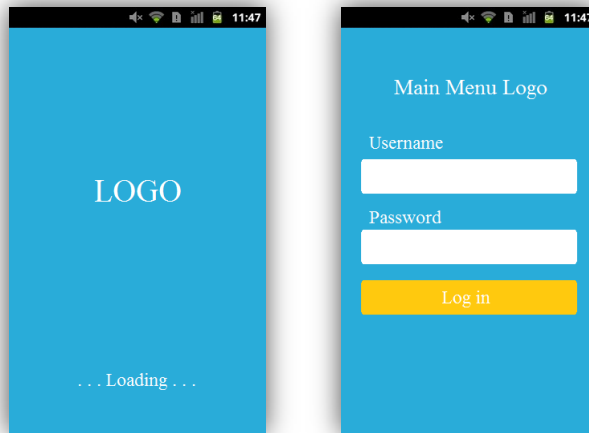
The second function in the main menu is called "History." In this function, the user will be able to select the history of the information that the user wants to see. The initial design of this section includes the history of energy usage and the history of temperature change. We will try to include other useful information as well. The information will be presented in the form of graph and table. Further discussion regarding to the data presentation will be discussed in the section 4.6.3 Data Displaying below.

Follows the history function, the third function will give the user the current information about the water heater. The application will show the current statistical data of the tank such as the current temperature, the average energy usage, the time stamp, and so on. The goal of this section is to give the user the sense of connection to the system.

Lastly, Setting function is the last option in the main menu screen. In this option, the user will be able to control the basic setting of the application. The user will be able to see the current status of the system, the current status of the wifi connection, the devices that are being connected to the system, as well as the configuration of the display setting.

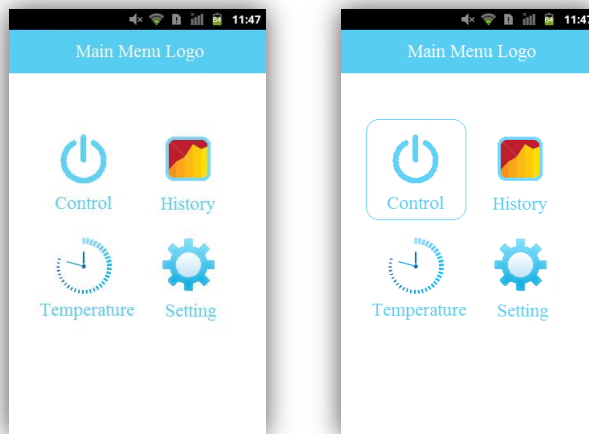
#### **4.6.2. User Interface**

The user interface is meant to be user friendly and to be easy to use to every user. With that in mind, we have design the application in such a way that it is very simple and basic but it gives user useful information. The application is easy to navigate since everything is fairly straightforward. Below are the sample user interface of our android application



**Figure 4.32 Android UI, Loading and Log In**

The first layout of the application will be the loading screen and the main screen. The main screen will consist of the log in form for the user to type in their username and password. Users are required to login in order to use the application. To prevent illegal access from people other than the owner of the water heater, each user has unique identity and will be assigned the unique log in information.



**Figure 4.33 Android UI, Main Menu and Select Control**

The main menu screen will consist of several functionalities that the application provide, including Control, History, Temperature, and Setting.

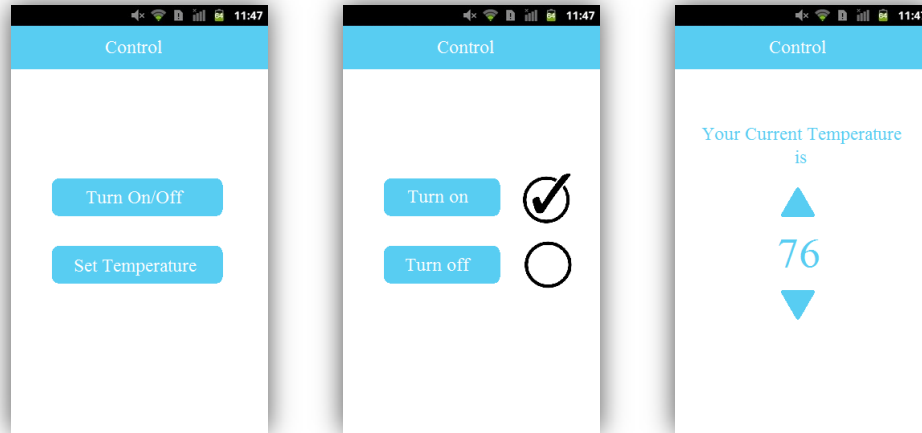


Figure 4.34 Android UI, Control, Turn on, and Temperature Selection

Control function is the option that allows user to control the water heater. The user will be able to turn on or turn off the water heater from their devices as well as to set the temperature of the water heater

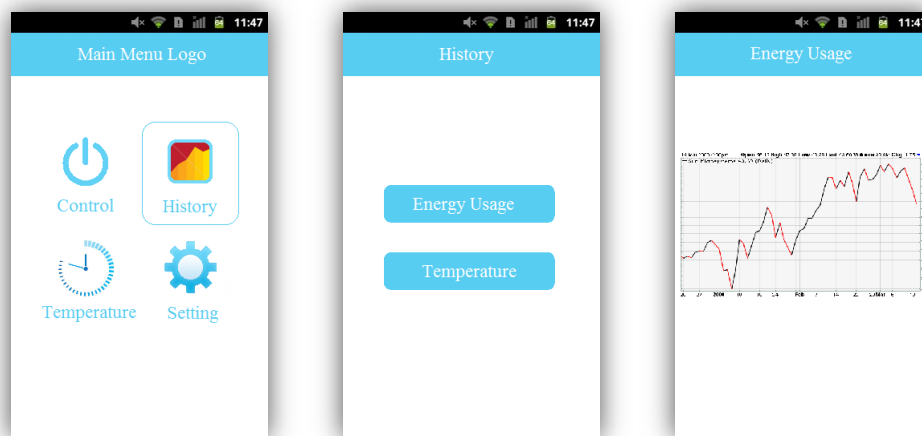
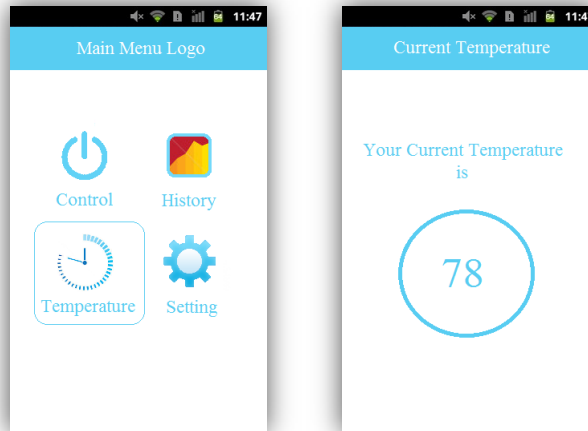


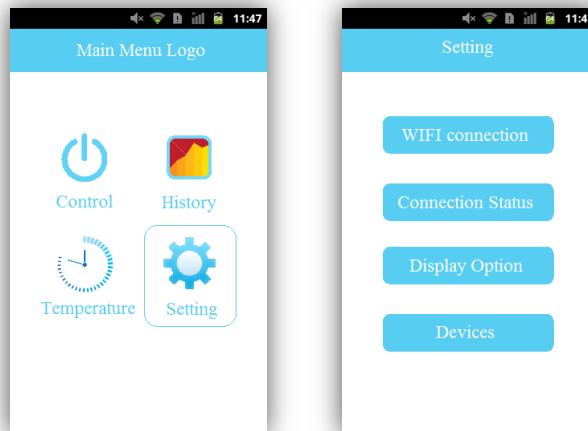
Figure 4.35 Android UI, Select History, History Type, and History Sample

History function will display the information that NEST has stored in the database, including history of temperature and energy usage.



**Figure 4.36 Android UI, Select Temperature and Show Temperature**

Temperature function will display the current temperature of the water heater.



**Figure 4.37 Android UI, Select Setting and Show Setting**

Setting function allows the user to check the WIFI connection, check the connection between the device and the water heater, change display option, and check to see the devices that are connecting to the system.

### 4.6.3. Data Displaying

To reduce the amount of wasted energy, we need to be able to see the history of the energy usage so that we can improve our behavior on the use of energy. One of the functionalities of the application is to display the usage of the energy and the temperature of the water heater. The history of energy usage and the history of temperature changing will be displayed within the 3 months time frame.

For the energy usage history, the data will be displayed as a graph of the energy usage follows by a set of statistical number indicates the usage of the energy of the current month. The graph will consist of the three months time period for the x-axis and the amount of energy in Kwh.

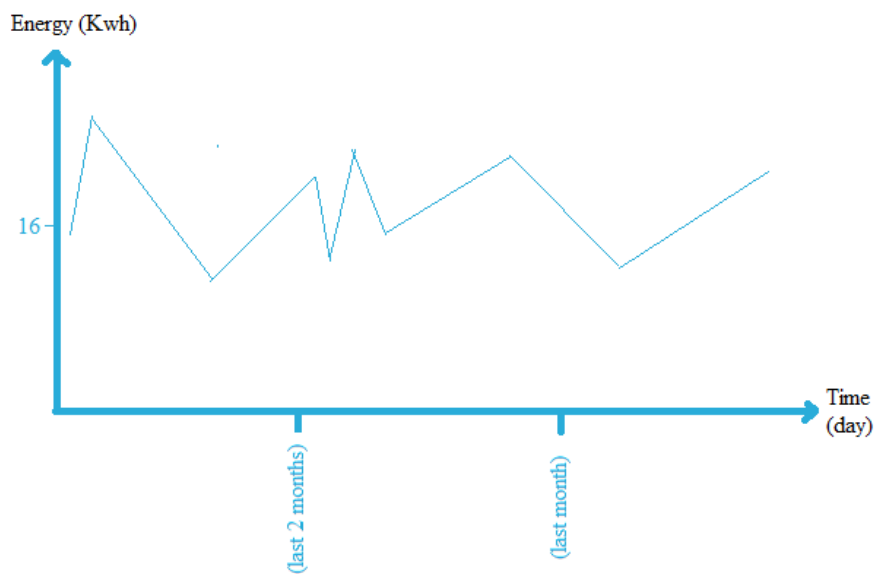


Figure 4.38 History Of Energy Usage Graph

For the temperature history, the data will be displayed as the table where horizontal attributes will consist of 24 hours in a day and vertical attributes will consist of 7 days in a week. Each cell will contain the average temperature of the tank during that specific hour as shown below.

Day\Hour	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Monday																								
Tuesday																								
Wednesday																								
Thursday																								
Friday																								
Saturday																								
Sunday																								

**Table 4.39 History of Temperature Change**

### 4.7 Artificial Intelligence

We have not devised a fully-fledged algorithm yet but it will take the following data into consideration:

- Nest Data – Using occupancy (when users are home) and activity (when the HVAC system is on and running) we will have a pretty good idea about when hot water is needed. Additionally, if the user sets the thermostat to away or vacation mode, we’ll have his information as well.
- Tank Temperature – We will constantly monitor temperature, not only to maintain it when needed, but also to figure out how long it takes to heat up any amount of water.
- Water Flow – By measuring the inflow and outflow of water, along with the occupancy data from the Nest we’ll have a finer grain of accuracy about usage times and needs.
- Current Sense – We’ll use this to calculate efficiency and weigh which heating mechanism is more appropriate at the time.

The goal is to reduce power consumption when compared to a standard water heater. This will be done in two ways: keeping the heating elements off as much as possible, and if there is more than one source of heat (heating element and heat pump), to decide which one to use (or both). It is also of utmost importance to make this process seamless to the user, always providing hot water when needed.

The learning process will be iterative. There is no way the system could know the exact user schedule from the installation short of hard coding it. The initial operation will tend to mirror regular heaters until usage patterns can be trusted to have some sort of accuracy. To speed up the process, we will perform some tests and come up with baseline numbers to have a relatively usable experience out of the box.



### 4.7.1 Learning Algorithm

In order to come up with the pattern recognition algorithm for our project, we have been looking up several supervised learning algorithms so we can understand how machine learning works. Below are the three algorithms that we consider to use or to adapt it to our project.

#### 4.7.1.1 Simple Linear Regression (SLR)

Simple linear regression is the most commonly used technique for determining how the interested dependent variable is affected by the independent variable. The main purposes of simple linear regression are to describe the dependency of the variables and to predict the values of one variable from another.

In this case, our dependent variables will consist of those that we have mentioned before: Nest Data, the temperature, water flow, and current sense. In the other word, these are the information that we will give to the user. The independent variable in this case would be time since we want to predict the user behavior from looking at those information at specific time period.

The derivation of simple linear regression equations are fairly straightforward. By given the data that we collect from the Nest. Let D represents the dependent data that we are considering and T be the independent data, which in this case is time. The set of these data will be in the form of coordinate point (T, D). With these points, we will find the best fit line to predict future data D' from the equation

$$D' = a + bT$$

where the sum of squared error in D,  $\sum(D_i - D_i')^2$  is minimized and a and b are fixed regression parameters to be determined from the data that can be calculated from the following equations

$$a = M_D - b M_T$$

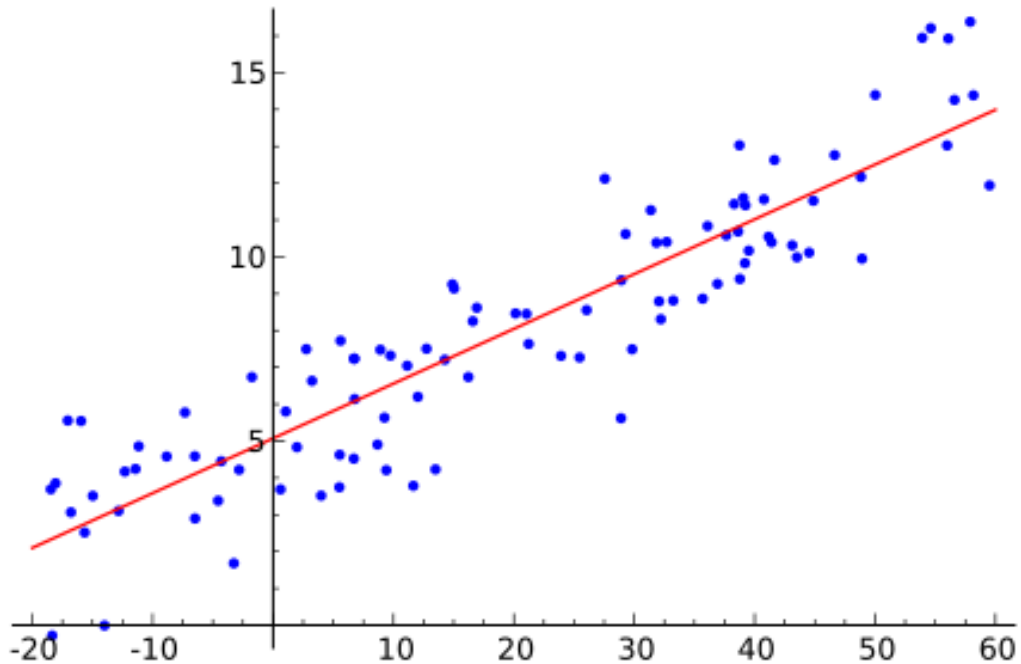
$$b = r \frac{S_D}{S_T}$$

where  $M_D$  is the mean of D

$M_T$  is the mean of T

$S_D$  is the standard deviation of D

$S_T$  is the standard deviation of T



**Figure 4.40 Sample Linear Regression**

There are several pitfalls in simple linear regression such as mistakenly attributing causation where the regression assumes that T causes D but it cannot prove that T causes D or overlooking hidden variables where hidden variables can distort the dependence of D on T. Because we still don't know what kind of data pattern we will get from the nest for each user. There will be limitations from this model that will affect the prediction of our data. With that in mind, we will have to modify this model in order to adapt it to our system if we are to use linear regression.

#### **4.7.1.2 Support Vector Machine (SVM)**

Support Vector Machines are supervised learning models with associated learning algorithms that analyze data and recognize patterns. It is a well-know approach in the machine learning community. It is usually implemented for a classification problem in a supervised learning framework. But SVM can be used also in a regression problem, where we want to predict or explain the values taken by a continuous dependent variable.

##### **4.7.1.2.1 Support Vector Classification**

The concept is mainly focusing on finding optimal separating hyperplane which gives us the classifier of the future data. The classification problem can be

restricted to consideration of the two-class problem. Consider the problem of separating the set of training vectors belonging to two separate classes

$$\mathcal{D} = \{(x^1, y^1), \dots, (x^l, y^l)\}, \quad x \in \mathbb{R}^n, y \in \{-1, 1\},$$

with a hyperplane,

$$\langle w, x \rangle + b = 0.$$

The set of vectors is said to be optimally separated by the hyperplane if it is separated without error and the distance between the closest vector to the hyperplane is maximal.

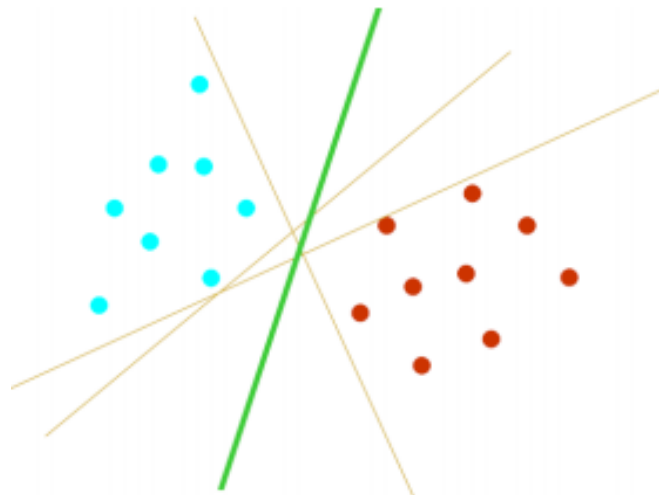


Figure 4.41 Optimal Separating Hyperplane

#### 4.7.1.2.2 Support Vector Regression

Suppose we have the training data  $\{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{R}^n \times \mathbb{R}$  where  $X$  denotes the space of the input patterns. with a linear function

$$f(x) = \langle w, x \rangle + b.$$

The optimal regression function is given by the minimum of the functional,

$$\Phi(w, \xi) = \frac{1}{2} \|w\|^2 + C \sum_i (\xi_i^- + \xi_i^+),$$

where  $C$  is a pre-specified value, and  $\xi^-$ ,  $\xi^+$  are slack variables representing upper and lower constraints on the outputs of the system.

## 5. Design Summary of Hardware and Software

### 5.1. Hardware

The hardware design requires the following major components: Water Heater tank, Controller Board, Touchscreen, Nest Thermostat and some sort of Android device. The system also depends on an underlying, already existing Wireless Network that will be assumed since it is also a requirement for the Nest. The controller board acts a central connection hub, having direct access to the rest of the components. The android device and nest will communicate wirelessly to the control board and the touchscreen and water heater through GPIO pins. Any changes to working state of the system will be triggered by external data from any of the peripheral devices, but will be ultimately made through logic on the board.

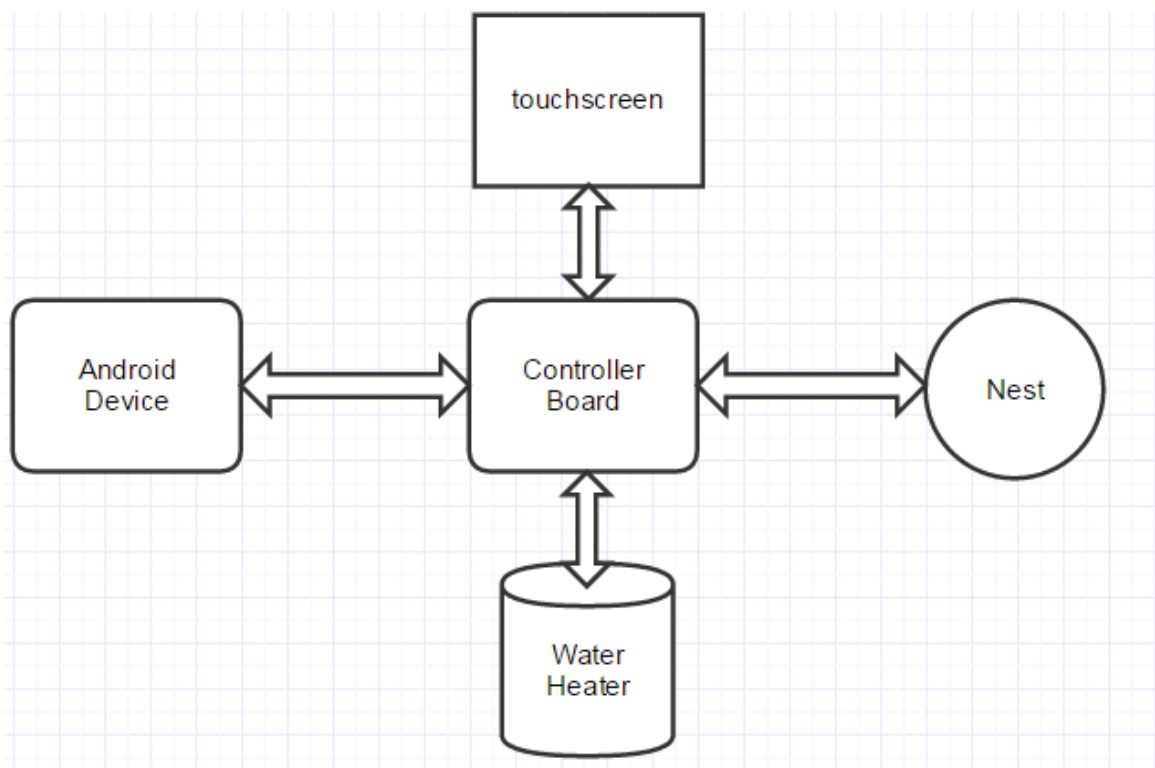


Figure 5.1 Major Components

Currently the Geospring water heater most closely resembles our system, and as such we will be taking design cues. Trying to improve on its design, we will upgrade the communication system to make use of wireless networks instead of wired Ethernet and make use of a touchscreen interface instead of 7 segment displays and buttons. These upgrades will not only make it more appealing and ease installation (Ethernet wiring is not usually found in garages or closets) but it will also modernize and already modern system.

## **5.2. Software**

The software designed will be distributed over three different platforms. The controller software will be responsible for bringing up the system and interfacing with the physical components as well as making decisions from the gathered data.

The android environment will be a portable user interface that will act as a remote control for the system.

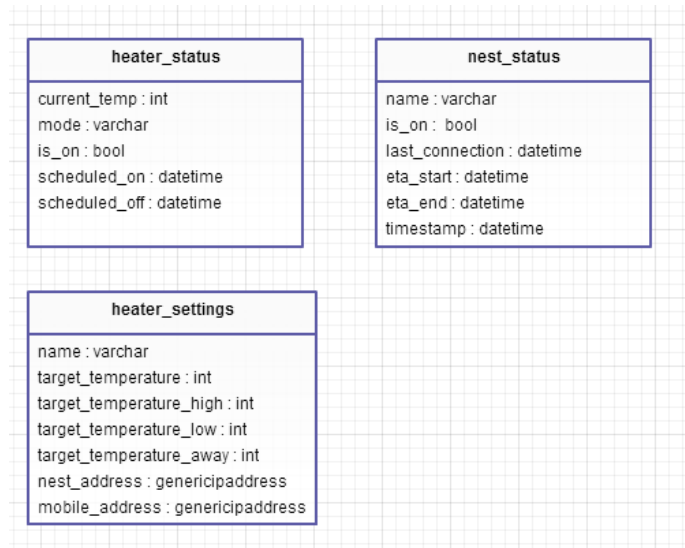
The Nest will be mostly used as a remote, easily accessible and well positioned remote sensor to gather occupancy data.

Software design will have to be carefully planned since the three platforms will have three completely different development environments. We will use HTTP POST requests with JSON data because it is natively used in the Nest, but also because it is easy to work with on any platform.

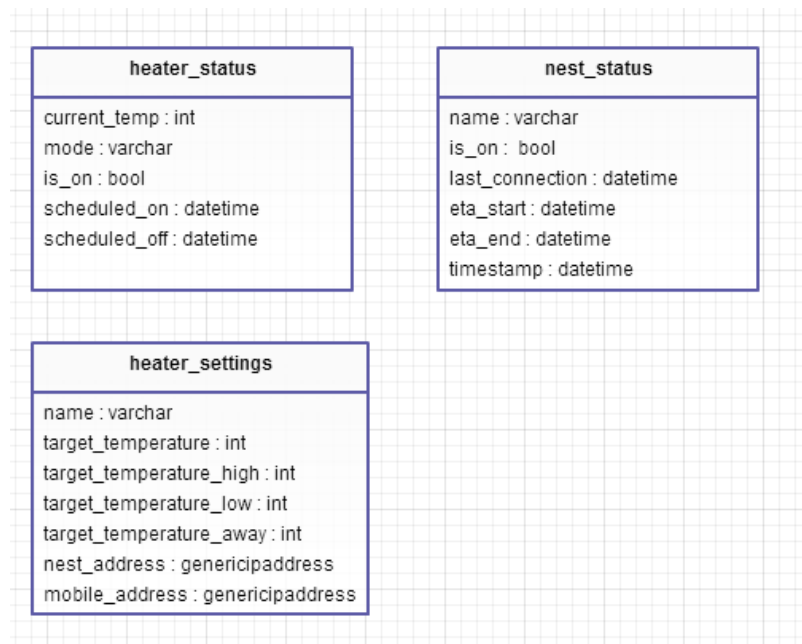
In the most basic sense, we will have to create a schedule from house occupancy data and water flow data. Since the nest already has an occupancy schedule, we will poll for this information and use it in our system. Temperature transfer cuvees for a 40 gallon water tank and a house are expected to be totally different but linear with respect of starting temperature and volume. Temperature sensors will aid us in figuring out how quickly the system reacts and by how much to shift the occupancy data to accommodate the users' schedule. Since the correlation of AC to hot water usage is not the same, we will use the added water flow data to refine the schedule.

Certain variables will be static until they get explicitly changed by the user, like for example target water temperature, ip addresses, etc. All these will have to be initially set on the touch screen since they will define the settings used to communicate with the wireless devices. After the network parameters are set up, these settings will be able to be changed from other devices. Though the water heater is not expected to go through very many power cycles, these settings should be persistent as to avoid having to set it up repeatedly.

Status variables from the heater will be used to communicate with the wireless devices. Through these variables, the android device and nest will know the current state of the water heater.



**Figure 5.2 Device data kept in database**



**Figure 5.3 Sensor data kept in database**

Temperature and flow sensor values are going to be used in real-time to set the state of the water heater. These values will also be used over time to try to fine tune patterns and also to be able to display a history to the user.

## 6. Project Prototype Construction and Coding

### 6.1. Parts Acquisition and BOM

The following tables outline our bill of materials for the various systems. Many of the passive components are likely already in the component collection of the team, and those that are not owned will be procured from online component vendors such as Digikey and Adafruit.

#### Power Circuit 1

Name	Type	Value	Qty	Price (USD)
LM7805CT	Lin. Regulator	+5VDC out	1	.62
LM7815CT	Lin. Regulator	+15VDC out	1	.67
LM7909CT	Lin. Regulator	-9VDC out	1	.61
HD04-T	Dac Diode Bridge	-	1	.12
T1	Transformer	-		*
C1, C4, C7	Capacitor	2200 $\mu$ F, 50V	3	1.21
C2, C5	Capacitor	100nF, 24V	2	.18
C3, C6	Capacitor	220nF, 50V	2	1.09
R1	Resistor	500 $\Omega$	1	.10
R2	Resistor	1000 $\Omega$	1	.10

Table 6.1 Bill Of Materials, Power Circuit 1

#### Power Circuit 2

Name	Type	Value	Qty	Price (USD)
Cdd	Capacitor	1.0 $\mu$ F	1	.05
Cin, Cin2	Capacitor	4.7 $\mu$ F	2	1.83
Cout	Capacitor	390 $\mu$ F	1	.63
Cs	Capacitor	1.0 nF	1	.02
D1	Diode	VF=1.3V	1	.09
D2	Diode	VF=500mV	1	.14

Table 6.2 Bill Of Materials, Power Circuit 2 Part 1

Name	Type	Value	Qty	Price (USD)
D3	Diode	VF=850mV	1	.21
Dac	Diode Bridge	VF=1.0V	1	.12
Dz	Diode, Zener	-	1	.11
L1	Inductor	470 $\mu$ H	1	.26
M1	Mosfet	-	1	.41
Rbld	Resistor	30.1k $\Omega$	1	.01
Rcbc	Resistor	309.0 k $\Omega$	1	.01
Rcs	Resistor	950.0 m $\Omega$	1	.1
Rdd	Resistor	22.0 $\Omega$	1	.01
Rfbb	Resistor	25.5 k $\Omega$	1	.01
Rfbt	Resistor	191.0 k $\Omega$	1	.01
Rg1	Resistor	10.0 $\Omega$	1	.01
Rg2	Resistor	10.0 k $\Omega$	1	.01
RI	Resistor	5.0 $\Omega$	1	.01
Rlc	Resistor	1.96 k $\Omega$	1	.01
Rs	Resistor	97.6 $\Omega$	1	.01
T1	Transformer	.149 Ns to Np	1	*
U1	Switcher	-	1	.41

**Table 6.3 Bill Of Materials, Power Circuit 2 Part 2**

Sensors, Extra Hardware, Etc

Name	Type	Value	Qty	Price (USD)
YF-S201	Flow Sensor	-	1	9.95
TMP36	Temp Sensor	-	2	1.50
-	Pump	-	1	Owned
-	Tank	-	1	Owned
-	Hot Plate	-	1	Owned
-	Fan	-	1	Owned
LB04302	LCD	-	1	29.95
RA8875	LCD Driver	-	1	34.95
STMPE610	Touch Driver	-	1	9.95
	JTAG cable	-	1	
XBee Pro	Wireless	-	1	24.95
ZedBoard	Dev Board	-	-	On Loan

**Table 6.4 Sensors, Misc Hardware**



## 6.2. PCB Vendor and Assembly

We had initially planned on using OSH Park for our PCB manufacturing based on good reviews but their capabilities do not meet our needs. OSH Park only offers 2 and 4 layer boards making it impossible to route all the pads from under the IC. At this level of complexity a home-made PCB is not an option either.

We have looked into a few more manufacturers capable of producing 10+ layer boards and found Gold Phoenix Printed Circuit Board, Sierra Circuits and Advanced Circuits. Advanced Circuit is the most attractive choice because they offer student discounts of up to \$500. Not realizing that OSH Park was not a suitable manufacturer earlier on was a serious overlook that could cost us a great deal of extra money and put us over our budget.

We will be using a mixture of SMD and through-hole parts. Undoubtedly the most challenging part of the assembly will be placing the CLG225 (basically BGA) Zynq IC. This is nearly impossible by hand but we have access to an industrial reflow oven in which we will be able to place all smd components at once. We will use a stencil to lay out the solder paste and run the board through the oven to solder the components. After that we will be able to solder by hand all through-hole components left.

## 7. Project Prototype Testing

This section details the testing hardware and procedures to be used for prototype testing. Given the similarities of the hardware between various water heater types, cross compatibility should be relatively easy to test without an overly wide range of test hardware.

### 7.1. Hardware Test Environment

The test environment for the device will be custom made. In lieu of a large, difficult to transport, and expensive water heater, a custom test-bed will be made that mimics the variables of a water heater. The test-bed will be made up of three main components: The water reservoir, a heating surface, and a water pump.

#### 7.1.1. Water Reservoir

The water reservoir will mimic the tank of the water heater, albeit in a much more manageable size.

- 5 to 10 gallons
- Able to withstand temperatures up to 175 degrees Fahrenheit

- Not electrically conductive, or electrically insulated
- Sturdy, but as lightweight as possible.

The tank will need to be easily portable, so the upper capacity will be limited to 10 gallons. Water has a weight of approximate 8 pounds per gallon, so transporting the tank at full capacity will be troublesome. The tank will need to be filled on-site, therefore a lid or porthole will be required instead of a water heater's standard plumbed-in piping. Testing will require a submersible pump (explained below) and a thermocouple to be submerged in the water. Due to the use of 120VAC, the tank will need to be electrically non-conductive or electrically isolated by an insulated shell for safety purposes. Standard electric water heaters consist of an inner metal tank, and a lighter weight outer shell. Insulation often fills the small area between the inner tank and shell. The thermal properties of the tank are not under test, so the use of thermal insulation on the test-bed will not be required beyond what is needed for environment and tester safety.

### **7.1.2. Heating Surface**

Standard electric water heaters have single or double heating elements that are inserted into the side of the tank. The usage of actual water heater elements and insertion mounting would prove troublesome and require machining tools and services that would need extra funding. In the place of electric elements, the test-bed will use a standard hot plate.

The hot plate will be modified so that the testing device will be able to control the amount of heat required by the PID controller to maintain the temperature required by the controller unit. The modifications will be kept as simple as possible to emulate a wide range of water heaters. In many older water heaters, the current flow in to the element was not regulated. The element was on, with full current flow, or off and without current. Because of the testing needs of the device, a simple on/off method will suffice in lieu of a much more complicated temperature regulation system on the heating surface.

### **7.1.3. Water Pump**

A small, submersible water pump will be used to simulate the movement of water through the water heater's tank and throughout the home's plumbing system. This will allow the flow-rate and temperature sensing hardware to be tested. As water leaves the tank, the flow-rate sensor will calculate how much is used for the learning algorithm. As the heated water exits, cooler water enters the tank to maintain a minimum water level. The cooler water will alter the temperature of the remaining water, and require the controller module to poll the thermocouple to ascertain if the PID controller needs to activate to maintain a set temperature.

The choice of the water pump will be based on cost and flow rate. The pump will not be required to have a long lifespan, and overall run-time will likely be under 15 hours. The flow rate will be chosen to emulate the flow rate of average household plumbing. This is not set at an exact rate, as the water pressure from

outside utility providers can vary by geographic locations. A submersible garden pond pump found in most hardware and home stores will suffice.

## **7.2. Hardware Specific Testing**

Since this device will be tested using a custom test-bed and not actual water heaters, the team will be limited in its ability to test the device against both standard and hybrid style water heaters. To simulate a hybrid water heater with its multiple modes, the device will be expanded to multiple control outputs. Using the flow meter to determine demand, a secondary or tertiary control will be switched on to simulate going from High Efficiency Mode (utilizing just the heat pump system) to a Hybrid Mode (which utilized heat pump systems in conjunction with electrical element systems)

This will allow the device to be tested for hybrid water heaters, while saving the team in finances and logistics (hybrid water heaters are relatively new innovations, and are higher in price than their less efficient electrical and gas counterparts).

### **7.2.1. Unit Test**

Each subsystem will be tested individually on the test-bed to confirm accuracy. After each subsystem is tested and confirmed to work correctly, the entire system will be tested as a single unit to insure that all of the subsystems work together.

#### **7.2.1.1. Temperature Sensor**

The TMP36 temperature sensor being used in this design is relatively easy to test as an individual component. To test the component, a digital multimeter, small laboratory power supply, and a breadboard will be used.

Since the voltage drop across the transistor in the sensor is linear with temperature change, the formula

$$Temp \text{ in } ^\circ C = \frac{(Vout \text{ in } mV) - 500}{10}$$

will give the temperature measured by the sensor by simply measuring the voltage across two pins. Different temperatures can be tested by using hot or cold surfaces. A non-contact infrared thermometer will give the temperature of the testing surface, and the temperature sensor will be allowed time to climb or fall to the targeted temperature.

The temperature sensor testing will utilize a calibrated digital thermometer to confirm that the water temperature relayed by the temperature sensor to the control module is the actual temperature of the water. A calibrated digital

thermometer will be used for ease of use, speed, and safety. Standard mercury thermometers pose a risk of breaking, and take more time to reach the measured temperature. A non-contact infrared thermometer could not be used, as most low-cost IR thermometers will not give accurate readings on clear liquid surfaces.

#### **7.2.1.2. Flow Rate Sensor**

The water pump used on the test bed will be a non-variable speed pump and have a set throughput. Using the diameter of the flow rate sensor, the throughput of the pump, and a calculator, we can determine the expected maximum flow rate. Using this calculation, and allowing for a small variance, the measurement from the flow rate sensor will be compared to the calculations of what is expected.

The flow sensor that will be utilized in this hardware revision will be a plastic-housed, in-line pinwheel style flow meter. As liquid flows through the meter, the pinwheel spins. Attached to the pinwheel is a very small magnet. The rotation of the pinwheel and magnet combo is measured by a hall effect magnetic sensor.

With a known liquid throughput and a known flow-rate per pulse of the magnetic sensor, accuracy of the component can be tested using a simple microcontroller, lcd display, and code made freely available on [github.com](https://github.com). The code uses the known rate of pulses per liter of liquid provided by the flow meter and calculates the rate and amount of liquid that passes through the pinwheel

#### **7.2.1.3. Wi-Fi**

Testing of the XBee wireless module will make use of a personal computer, the XBee modules, Digi's X-CTU communications software, and an XBee Universal Serial Bus adapter board. The XBee module to be tested is connected to the computer via the USB adapter board. The secondary XBee module will be put on a breadboard connected to a 3.3 VDC power supply. The DIN and DOUT pins of the remote XBee will be bridged on the breadboard, allowing the remote XBee to act as a loopback device.

In the X-CTU software, the proper COM ports need to be chosen for the USB to XBee adapter. The software's built-in test will query the modem on the tested XBee and report if communication was successful and the XBee's modem type and firmware.

When testing the broadcasting and receiving of the XBee, the user types in the X-CTU software's terminal window. In the current iteration of the software, what the user types is shown in the window with blue text, and what is received is red text, letter by letter. If the send/receive test was successful, the terminal window will look like colorful text string below.

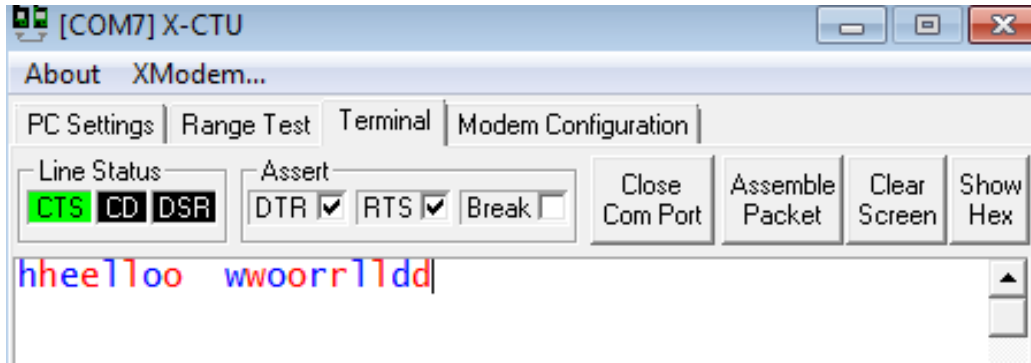


Figure 7.1 Example of Successful send/receive test

Also, during sending/receiving, the LEDs on each board will blink with each character sent or received. The X-CTU software will allow the user to bundle the test message as a single packet, which sends the test text as a single message and receives it in the same method.

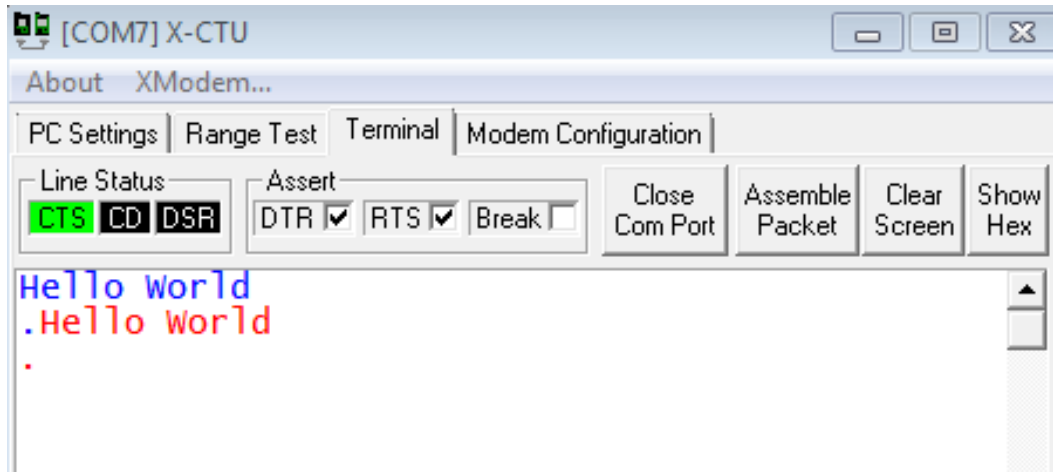


Figure 7.2 Example of successful single packet send/receive test

The X-CTU software will also test for range and Received Signal Strength Indicator value. The RSSI test will make certain that the signal will be between -40 dBm and -100 dBm. This range is the highest value and the minimum sensitivity of the XBee module.

#### 7.2.1.4. Touch Screen

The touchscreen will be another component tested with outside hardware. The touchscreen will be testing using an Arduino development board, an RA8875 TFT driver board, a STMPE610 touchscreen controller board, and open source software and libraries for the Arduino microcontroller. The touchscreen utilized in this design has some stringent requirements that the Arduino itself cannot provide. It is a raw pixel-dot-clock display and requires a constant 60Hz refresh with vertical and horizontal sync, along with a pixel clock. This is more than the Arduino development board on hand can provide, so the addition of the RA8875 driver board is necessary. The RA8875 provides the needed hardware to handle the 60Hz refresh and the 4MHz pixel clock requirements. In addition, it also provides a voltage regulator circuit that provides the voltage required for the display and backlights.

The RA8875 can also handle a 4-wire resistive touchscreen overlay, but it lacks precision. This would normally not be an issue, but the interface planned for the software application and the overall physical size of the screen require a slightly finer control precision than the RA8875 provides. The STMPE610 is added to the system to handle more precise control.

The open source test software is straightforward and provides a basic display and touch testing environment. The first battery of tests is a display test. This runs the display through full-screen color tests by making the entirety of the screen a single color for a few seconds. As color calibration is not an issue, the exact specifics of the colors are not terribly important. As long as the reds are somewhat red and the blues somewhat blue, etc, this test will suffice.

The second battery of tests prints multiple lines of text and characters. This checks to make sure the display can provide readability to text.

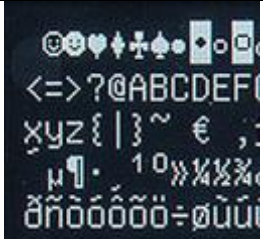

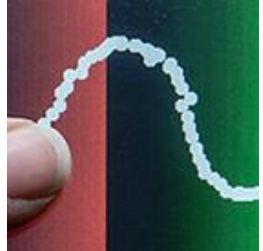
After text legibility, a combination of the first two tests is done. This test puts each character in a different colored box. Like the first test, precise color calibration is not required for this project, so as long as each colored block is different than its immediate neighbor and the character in the colored block is visible, the display will pass this test.

The final test is for the resistive touch overlay. This test allows the user to draw lines or shapes on the display using a finger or resistive stylus. A white line should be drawn on the display by the finger or stylus. As long as the drawn line corresponds to the location of the pointing device, the display will pass the test.

If the display cannot pass these basic tests, then it will be considered defective and a new display will be procured to take its place. Given the relaxed requirements of this project, it is unlikely that a more stringent set of tests will be required for the display. While the software interface of the final design will require basic accuracy, the interface will not be so complex as to require an

extreme amount of accuracy. The final software will be designed for clarity and ease of use, not examining the precision of the user's finger control.

A table giving a basic representation of the test software follows.

Test	Pass	Fail	Pass Example
Color	Each screen change is a different color than previous	No color changes displayed	Screen color is consistent across entire screen, and different color than previous screen
Text/Characters	Characters are all legible	Characters are not legible	
Combination	Each block is a different color than its neighbor, characters are still legible	No color change between blocks, illegible characters	
Touch	Drawn line follows finger or stylus with some accuracy	No drawn line appears, or does not correspond to stylus location	

**Table 7.3 Touch screen display test**

### 7.2.2. System Test

After each subsystem passes the required tests, the prototype will be subjected to tests to make certain the system as a whole is functioning properly. As each system relies on the others for the device to function properly, any interfacing problems amongst the separate subsystems will prove to be problematic.

Using an iteration of the software application being developed for this design, the system will be tested as an end user would expect it to work. The software provides the current water temperature on the display and the user instructs the

software to enact a temperature change. The software instructs the processor to poll the temperature sensor and monitor the temperature. The processor instructs the relay system to allow current to flow to the heating element. The heating element will heat the water and should stop relatively close to the desired temperature and then maintain this temperature as appropriate.

In normal operation mode, the temperature should only vary by a few degrees. The control system will cycle the element on and off as needed to maintain this small temperature range. In an energy saver mode, the allowed temperature range is widened, and the control system will allow the water to cool to a lower temperature before instructing the system to turn on the heating element.

If the system is set to away mode, then the control system will allow the water temperature to drop much further than the normal minimum temperature. This temperature is always kept above the freezing point to safeguard against appliance and pipe freeze damage.

With the system set to vacation mode, the control system will poll the temperature sensor with a minimum of frequency. The water temperature will be allowed to fall to the minimum safeguard temperature. If using the NEST, the control system can appropriate weather data from the NEST and determine if there is any possibility of low temperatures. If located in a warm climate, or during a season where there is no chance of freeze, the control system can drop to maximum power saving. This stops the polling of the temperature sensor, and the control system only maintains power to keep the time and schedule, and monitor for unplanned changes from the user.

Testing normal operation mode will give sufficient data to make certain that the hardware is working appropriately. Since the other various modes are mostly software related changes, if the hardware works in normal operation then the hardware stands a high likelihood of working in away and energy saving mode. This will also save time, as the tests for away mode would likely take hours, and vacation mode would take days to weeks.

<b>Test</b>	<b>Pass</b>	<b>Fail</b>
Local Temperature Change (Touchscreen)	Actual Temp Accurate to 1°C Relative to Set Temp	Inaccurate Temp, out of range of desired temp.
Remote Temperature Change (Android App)	Actual Temp Accurate to 1°C Relative to Set Temp	Inaccurate Temp, out of range of desired temp.
Remote Temperature Change (NEST)	Actual Temp Accurate to 1°C Relative to Set Temp	Inaccurate Temp, out of range of desired temp.
Mode Change – Hybrid	Control Unit correctly uses both electric elements and heat pump	Fails to use electric elements and heat pump

**Table 7.4 System Test 1**



<b>Test</b>	<b>Pass</b>	<b>Fail</b>
Mode Change – Away	Control Unit correctly sets power saving mode to minimum, monitors geofence	Does not set power saving mode correctly, does not monitor geofence
Mode Change – Vacation	Control Unit correctly sets power saving mode to minimum, monitors return time	Does not set power saving mode correctly, fails to monitor time of return for homeowner
Mode Change – Heat Pump	Control Unit uses only heat pump subsystem	Control Unit fails to use only heat pump system

**Table 7.5 System Test 2**

### **7.3. Software Test Environment**

Software application will be tested on all controlling devices including touch screen, NEST interface, and android devices.

#### **7.3.1. Touch screen**

The testing of touch screen application will be tested by checking the performance of all functions that are implemented inside the application embedded in the touch screen.

#### **7.3.2. NEST Interface**

Since there is no such thing as an emulator for a Nest, all the testing will be directly done on the device itself. This means that each new bit of functionality we add will be tested immediately, especially so if other functions are dependent on this.

First of all we will need to demonstrate that we have control over the hardware by displaying some sort of graphic or banner that let us know that we have injected our code into the hardware. From there we will need to show that we can switch back and forth between the stock functions and the new 'Water' mode. At this point it will perhaps be useful to show raw text data of the packet exchanges between the nest and our system. Note that all of this will be show on the built in LCD.

#### **7.3.3. Android Phone/Tablet**

We will be testing the android application on several android platform including the actual android devices and emulator. Functionality-wise there should be no difference between different android devices. The main points of concern when developing apps for a multitude of devices are layout issues. The vast differences in screen sizes and resolutions make it almost impossible to design an attractive interface that works across the board.

### 7.3.3.1 Genymotion

Genymotion is an android emulator which comprises a complete set of sensors and features in order to interact with a virtual Android environment. With **Genymotion**, we can test our Android applications on a wide range of virtual devices for development, test and demonstration purposes.

### 7.3.3.2 Samsung Galaxy S II

The Samsung Galaxy S II is a touchscreen-enabled, slate-format Android Smartphone designed, developed, and marketed by Samsung Electronics. It has additional software features, expanded hardware, and a redesigned physique compared to its predecessor, the Samsung Galaxy S. The Galaxy S II was launched with Android 2.3 "Gingerbread".

### 7.3.3.3 Google Nexus 7

The Nexus 7 is a mini tablet computer co-developed by Google and Asus that runs the Android operating system. It is the first tablet in the Google Nexus series, a family of Android consumer devices marketed by Google and built by an original equipment manufacturer partner. The Nexus 7 features a 7-inch (180 mm) display, an Nvidia Tegra 3 quad-core chip, 1 GB of memory, Wi-Fi and near field communication connectivity, and 8, 16 or 32 GB of storage. The tablet was the first device to ship with version 4.1 of Android, nicknamed "Jelly Bean".

## 7.4. Software Specific Testing

For the android application testing, we will divide the system into several parts to make sure each and every part is working properly. After that, we will do the system test where we make sure that the application is working properly after the integration.

### 7.4.1. Functionalities

Below are the tests of the main interface portion

Tests	Description	Steps	Expected Result
Application load data	Test the program to see if it successfully connects to the data base and get pass loading screen	Open the application and wait until the main screen shows up	The application should take no longer than 30 seconds to connect with the database and open up the main screen

Table 7.6 Main Interface Testing 1

<b>Tests</b>	<b>Description</b>	<b>Steps</b>	<b>Expected Result</b>
Log in	Test to see if the program recognize the user identity	Type in the valid username and password in the main screen and press login	The application should take the given information, compare it to the information on the database, then open up the menu screen after validate user identity
Main Menu Functions	Test to see if the menu connects to the appropriate functions	Press each function	Each function should direct the user into the appropriate screen

**Table 7.7 Main Interface Testing 2**

Below are the tests of the Control portion

<b>Tests</b>	<b>Description</b>	<b>Steps</b>	<b>Expected Result</b>
Turn On/Off	Test to see if Turn On/Off function works	In the main menu screen, press the control icon. Inside control screen, press "Turn On/Off." Then , press on or off and check if the water heater response properly	The water heater should turn off when "Turn off" symbol is checked and it should turn on when "Turn on" symbol is checked
Set Temperature	Test to see if Set Temperature function works	In the main menu screen, press the control icon, Inside control screen, press "Set Temperature." Then, set the temperature to specific value	The water heater should have the desire temperature within the 30 minutes

**Table 7.8 Control Testing**

Below are tests of the History and Temperature portion

<b>Tests</b>	<b>Description</b>	<b>Steps</b>	<b>Expected Result</b>
Temperature History	Test to see if temperature history function works	In the main menu screen, press the History icon. Inside history screen, press Temperature. Check the graph	The graph should show the history of temperature change within certain period of time. The data should match with the data stored in the database
Energy Usage History	Test to see if energy usage history works	In the main menu screen, press the History icon. Inside history screen, press Energy usage. Check the graph	The graph should show the history of energy usage within certain period of time. The data should match with the database
Temperature	Test to see if Temperature function works	In the main menu screen, press the temperature icon. Check the value of the temperature	The temperature shown in the screen should be the same as the current temperature of the water heater tank

**Table 7.9 History and Temperature Testing**

Below are the tests of the Setting portion

<b>Tests</b>	<b>Description</b>	<b>Steps</b>	<b>Expected Result</b>
WIFI connection	Test to see if the application indicates the right status of WIFI connectivity	In the main menu screen, press Setting icon. Inside setting screen, press WIFI connection. Check WIFI status	The WIFI status inside the application should match with the device's status

**Table 7.10 Setting Testing 1**

<b>Tests</b>	<b>Description</b>	<b>Steps</b>	<b>Expected Result</b>
Water heater status	Test to see if the application connects to the water heater	In the main menu screen, press Setting icon. Inside setting screen, press Water Heater Status. Check the status.	The application should show the appropriate connectivity status that match with actual water heater
Devices	Test to see if Devices function works	In the main menu screen, press Setting icon. Inside setting screen, press Devices.	The application should show the correct status of the devices that are connecting to the water heater

**Table 7.11 Setting Testing 2**

Below is the test for GUI portion

<b>Tests</b>	<b>Description</b>	<b>Steps</b>	<b>Expected Result</b>
GUI	Check if the application displays the expected GUI on each device	For each android device, go through every screen inside the application and check all the symbols and the layout of the application	All the symbols and layouts should display properly

**Table 7.12 GUI Testing**

## 8. Administrative Content

### 8.1. Milestone Discussion

The project management approach we have been using can be best characterized as Agile management. We subdivided our short term task, which was completing this paper, and turned in smaller works over the allotted time to complete the project before the deadline.

It can be argued that a project milestone is an essential tool in project management. A project milestone is especially useful when it comes to showing what key steps need to be taken and the order in which the tasks need to be accomplished. At the beginning of the semester after we settled on a project idea we came up with the following project milestone

September	October	November	December	January	February	March	April
Research							
	Design						
	Ordering Parts						
			Prototyping				
			Build	Finalize			
			Testing				
			Debugging/Troubleshooting				
						Finalize Documents	
							Pres entat ion

**Table 8.1 Original Project Milestone**

September	October	November	December	January	February	March	April
Research							
	Design						
		Ordering parts					
			Prototype				
			Build	Finalize			
				Test			
				Debug/Troubleshoot			
						Finalize Documents	
							Presentation

**Table 8.2 Updated Project Milestone**

## 8.2. Budget and Finance Discussion

Due to the number of devices needed for a fully working system this is a relatively expensive project to carry out. Before settling down for this idea we discussed it with Dr. Jin and he decided to lend us a Zedboard development board as well as giving us access to one of the Nest thermostats in his laboratory. Other than a (brand new) water heater, these were the two most expensive items we did not already own. We set a loose budget of \$600 dollars which would be divided evenly between the three members of the group. We assumed this would be enough to purchase two sets of breakable (or burnable, rather) parts to put together. With this money we will buy all the parts required to put together the main PCB, as well as the accompanying touchscreen display and sensors.

So far we have had any expenses and as such we do not have any way of tracking them. Printing and binding this document will be the first expense, at which time we will create a spreadsheet to document purchases.

## 9. Appendices

### 9.1. Appendix A - Copyright Permissions

#### Department of Energy

<http://energy.gov/about-us/web-policies>

#### Copyright, Restrictions and Permissions Notice

Government information at DOE websites is in the public domain. Public domain information may be freely distributed and copied, but it is requested that in any subsequent use the Department of Energy be given appropriate acknowledgement. When using DOE websites, you may encounter documents, illustrations, photographs or other information resources contributed or licensed by private individuals, companies or organizations that may be protected by U.S. and foreign copyright laws. Transmission or reproduction of protected items beyond that allowed by fair use as defined in the copyright laws requires the written permission of the copyright owners.

Images on our website which are in the public domain may be used without permission. If you use images from our website, we ask that you credit "U.S. Department of Energy" as the source. Please note that some images on our site may have been obtained from other organizations. Permission to use these images should be obtained directly from those organizations.

DOE websites have links to many other websites. Once you access another site through a link that we provide, you are subject to the copyright and licensing restrictions of the new site.

#### Fair Use

<http://www.copyright.gov/fls/fl102.html>

#### GE Appliances

<http://www.geappliances.com/plcy/tandc2.htm>

#### Terms of Use

##### 1. Proprietary Rights.

Copyrights. GEA, its Affiliates, Service Providers and Business Partners reserve copyrights in all of the content of the Site, including but not limited to design, text, software, technical drawings, configurations, graphics, and other files and their selection and arrangement (the "Content"). The Content may not be modified, copied, distributed, framed, reproduced, republished, downloaded, displayed, posted, transmitted, or sold in any form or by any means, in whole or in part,



without GEA's prior written permission, *except that you may download and print Content for uses that are not competitive with or derogatory to GEA*, its Affiliates or Service Providers, provided that you keep all copyright or other proprietary notices intact and that you do not transfer any Content to any other person unless you give them notice of, and they agree to accept, the obligations arising under these Terms of Use.

### **Whirlpool Appliances**

<http://www.whirlpool.com/>

#### Copyright/Intellectual Property

This entire World Wide Web site is copyrighted under United States law and protected by worldwide copyright laws and treaty provisions. Users of the site MAY download or print one copy of any and all materials on the site for personal, non-commercial use, provided that they do not modify or alter the materials in any way, nor delete or change any copyright or trademark notice.

### **Software License Agreement (BSD License)**

<https://github.com/adafruit/Adafruit-Flow-Meter/blob/master/license.txt>

Copyright (c) 2012, Adafruit Industries

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

### **Adafruit copyright permission**

<https://www.adafruit.com>

Adafruit releases documentation under a Creative-Commons Share-alike, Attribution license

## **Creative Commons**

<http://creativecommons.org/licenses/by-sa/2.5/>

You are free to:

- Share — copy and redistribute the material in any medium or format
- Adapt — remix, transform, and build upon the material
- The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

- Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

## **Digi-Key Terms of Use and Conditions of Order**

<http://www.digikey.com/en/terms-and-conditions>

2. Intellectual Property. The Service, the Site, and all information and/or content that you see, hear or otherwise experience on the Site (the "Content") are protected by U.S. and international copyright, trademark and other laws, and belong to Digi-Key or its partners, affiliates, contributors or third parties.

Digi-Key grants you a personal, revocable, non-exclusive, non-transferable license to use the Site, the Service and the Content and to download, print and store portions of the Content that you select, provided that: (1) you only use these copies of the Content for your own internal business purposes or your personal, non-commercial use; (2) if you are a competitor of Digi-Key, commercial aggregator of data, or other commercial user, you do not copy or post the Content on any network computer or transmit, distribute, publish or broadcast the Content in any media; and (3) you do not modify or alter the Content in any way, or delete or change any copyright or trademark notice. No right, title or interest in any downloaded Content or materials is transferred to you as a result of this license. Digi-Key reserves complete title and full intellectual property rights in any Content you download from the Site, subject to this limited license for you to make personal use of the Content as set forth herein.

## **AVNet.com Terms of Use**

<http://www.avnet.com/en-us/legal/Pages/legal-notice.aspx>

### **INTELLECTUAL PROPERTY**

All trademarks, service marks, logos, slogans, domain names and trade names (collectively "Marks") are the properties of their respective owners. Avnet disclaims any proprietary interest in Marks other than its own.

The copyright in all original material provided on this website is held by Avnet, or by the original creator of the material. Except as stated below, none of the material may be reproduced, distributed, republished, downloaded, displayed, posted, transmitted or copied in any form or by any means, without the prior written permission of Avnet, and the copyright owner. *Permission is granted to display, copy, distribute and download the materials on this website solely for personal, non-commercial use provided that you make no modifications to the materials and that all copyright and other proprietary notices contained in the materials are retained.*

## **ACEEE**

<http://www.aceee.org/terms-conditions>

### **3. Use of Publications.**

The Publications posted on this Site, except for materials that contain a copyright notice for a third party other than ACEEE, are the property of ACEEE or its licensors or affiliates and are made available for third-party copying, reproduction, distribution, modification, display, republication, transmittal, reposting, or otherwise using only upon receipt of ACEEE's prior written permission, to be granted in ACEEE's sole discretion on a case-by-case basis. To request such permission, please contact us at [aceeeinfo@aceee.org](mailto:aceeeinfo@aceee.org).

## 9.2. Appendix B - List of Tables and Figures

<b>Table/Figure Number</b>	<b>Table/Figure Name</b>	<b>Page Number</b>
2.1	Project Management	5
3.1	Whirlpool Energy Smart Water Heater Tank	6
3.2	Whirlpool Energy Smart Water Heater Control	7
3.3	Whirlpool Energy Smart Water Heater Specifications	7
3.4	GeoSpring Hybrid Water Heater Control	8
3.5	GeoSpring Hybrid Water Heater Tank	9
3.6	Nest Thermostat	11
3.7	Honeywell Lyric Thermostat	12
3.8	Smart Thermostats	14
3.9	Infrared Thermometer	15
3.10	LM35 Temperature Sensor	16
3.11	Examples Of Embedded Systems Usually Found Around Homes	17
3.12	Comparison Between Different Security Protocols	18
3.13	iOS Devices	19
3.14	Android Devices	20
3.15	Comparison between Android and iOS 1	20
3.16	Comparison between Android and iOS2	21
3.17	2012-2013 Smartphone Market Share	22
3.18	Resistive Touch Screen Technology	23
3.19	Capacitive Touch Screen Technology	23
3.20	Surface Acoustic Wave Touch Screen Technology	24
3.21	Infrared Touch Screen Technology	25

<b>Table/Figure Number</b>	<b>Table/Figure Name</b>	<b>Page Number</b>
3.22	ZedBoard Xilinx Zynq-7000	26
3.23	Zynq Z7020 CSG484 Bank Assignments	27
3.24	ZedBoard Block Diagram	28
3.25	CFAF320240F-035T Touch Screen	29
3.26	Basic Architecture of System	30
3.27	Block Diagram For Controller Board	31
3.28	Block Diagram Of Android Device	32
3.29	NEST Block Diagram	33
4.1	ZedBoard Specification 1	34
4.2	ZedBoard Specification 2	35
4.3	TMO36 Temperature Sensor Pinout	36
4.4	Suggested TMP36 Usage	37
4.5	TMP36 Temperature Sensor Specs	37
4.6	Flow Sensor Specs	38
4.7	The YF-S201 Flow Rate Sensor	3.9
4.8	Optical Relay Usage	40
4.9	Project Layout	43
4.10	NEST Data Structure 1	45
4.11	NEST Data Structure 2	46
4.12	Nest Data Structure 3	47
4.13	Wireless Communication Routes	48
4.14	XBee Specifications	48
4.15	Pinout data for XBee 1	49
4.16	Pinout data for XBee 2	49
4.17	XBee Pro S1	50
4.18	Mock-up Of Possible Water Heater Mode	51
4.19	Standard Nest Thermostat Boot Process	52
4.20	SunBoard Touch Screen Pinout 1	53
4.21	Sunbond Touch Screen Pinout 2	54

<b>Table/Figure Number</b>	<b>Table/Figure Name</b>	<b>Page Number</b>
4.22	Voltage Needs: Display	55
4.23	Power Specs: Flow Sensor	55
4.24	Power Specs: XBee Module	56
4.25	Voltage Needs: Zynq 7000 Series SOC	56
4.26	Power Circuit 1	57
4.27	Component Values, Power Circuit 1	58
4.28	Bill of Materials, Power Circuit 1	58
4.29	Power Circuit 2	59
4.30	Bill Of Materials, Power Circuit 2	60
4.31	Class Diagram of Application Functionalities	61
4.32	Android UI, Loading and Log In	63
4.33	Android UI, Main Menu and Select Control	63
4.34	Android UI, Control, Turn On, and Temperature Selection	64
4.35	Android UI, Select History, History Type, and History Sample	64
4.36	Android UI, Select Temperature and Show Temperature	65
4.37	Android UI, Select Setting and Show Setting	65
4.38	History Of Energy Usage Graph	66
4.39	History of Temperature Change	67
4.40	Sample Linear Regression	69
4.41	Optimal Seperating Hyperplane	70
5.1	Major Components	71
5.2	Device data kept in database	73

<b>Table/Figure Number</b>	<b>Table/Figure Name</b>	<b>Page Number</b>
5.3	Sensor data kept in database	73
6.1	Bill Of Materials, Power Circuit 1	74
6.2	Bill Of Materials, Power Circuit 2 Part 1	74
6.3	Bill Of Materials, Power Circuit 2 Part 2	75
6.4	Sensors, Misc Hardware	75
7.1	Example of Successful send/receive test	80
7.2	Example of successful single packet send/receive test	80
7.3	Touch Screen Display test	82
7.4	System Test 1	83
7.5	System Test 2	84
7.6	Main Interface Testing 1	85
7.7	Main Interface Testing 2	86
7.8	Control Testing	86
7.9	History and Temperature Testing	87
7.10	Setting Testing 1	87
7.11	Setting Testing 2	88
7.12	GUI Testing	88
8.1	Original Project Milestone	89
8.2	Updated Project Milestone	90

### 9.3. Appendix C - References

- [1] <http://talks.caktusgroup.com/djangocon/2014/django-project/#/19>
- [2] <https://www.blackhat.com/docs/us-14/materials/us-14-Jin-Smart-Nest-Thermostat-A-Smart-Spy-In-Your-Home-WP.pdf>
- [3] <http://www.geckoandfly.com/10380/wep-vs-wpa-vs-wpa2-comparison-table/>
- [4] [http://33sticks.com/wp-content/uploads/2014/01/face\\_6-b5871f36.png](http://33sticks.com/wp-content/uploads/2014/01/face_6-b5871f36.png)
- [5] <http://www.beachstreetnews.com/wp-content/uploads/2013/01/NEST-Thermostat1.jpg>
- [6] [http://en.wikipedia.org/wiki/Nexus\\_7\\_\(2012\\_version\)](http://en.wikipedia.org/wiki/Nexus_7_(2012_version))
- [7] [http://en.wikipedia.org/wiki/Samsung\\_Galaxy\\_S\\_II#Software\\_and\\_services](http://en.wikipedia.org/wiki/Samsung_Galaxy_S_II#Software_and_services)
- [8] [http://seismo.berkeley.edu/~kirchner/eps\\_120/Toolkits/Toolkit\\_10.pdf](http://seismo.berkeley.edu/~kirchner/eps_120/Toolkits/Toolkit_10.pdf)
- [9] <http://onlinestatbook.com/2/regression/intro.html>
- [10] [http://en.wikipedia.org/wiki/File:Linear\\_regression.svg](http://en.wikipedia.org/wiki/File:Linear_regression.svg)
- [11] <http://users.ecs.soton.ac.uk/srg/publications/pdf/SVM.pdf>
- [12] [http://www.diffen.com/difference/Android\\_vs\\_iOS](http://www.diffen.com/difference/Android_vs_iOS)
- [13] <http://appleinsider.com/articles/13/11/12/idc-data-shows-66-of-androids-81-smartphone-share-are-junk-phones-selling-for-215>
- [14] <http://tech.co/ios-vs-android-app-development-consumer-experience-comparison-2014-02>
- [15] [http://www.gartner.com/technology/research/methodologies/research\\_mshare.jsp](http://www.gartner.com/technology/research/methodologies/research_mshare.jsp)
- [16] <http://www.engineersgarage.com/articles/touchscreen-technology-working?page=1>
- [17] <http://zedboard.org/?inline=true#colorbox-inline-2>
- [18] <http://entesla.com/image/cache/data/Products/Sensors/LM35-900x900.jpg>
- [19] [http://shopbuy.org/static/category/original/infrared-thermometers/infrared\\_thermometer.jpg](http://shopbuy.org/static/category/original/infrared-thermometers/infrared_thermometer.jpg)