May 2014

Thermoelectric Management of Lithium Ion Batteries in Mobile Devices

Edward Eric Allison
Worcester Polytechnic Institute

Jeffrey Pascal Paquette
Worcester Polytechnic Institute

Liv Pasqualina Radder
Worcester Polytechnic Institute

Project Number MQP CF – EP

Thermoelectric Management of Lithium Ion Batteries in Mobile Devices

A Major Qualifying Project Report

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

in Mechanical Engineering

and

Degree of Bachelor of Science

in Electrical Engineering

by

Edward Allison

____

Jeffrey Paquette

____

Philip Radder

____

Approved:

____

Prof. Cosme Furlong, Major Advisor

Table of Contents

List of Figures

List of Tables

Table of Equations

Acknowledgements

The team would like to thank:

Our advisor, Professor Furlong of the Mechanical Engineering Department for his guidance and support throughout the duration of the project.

The Mechanical Engineering department for the use of the lab and lab equipment for performing experiments for the project.

Professor Bitar, of the Electrical Engineering Department, for his helpful advice and guidance as a member of the Electrical Engineering Department.

Professor McNeill, of the Electrical Engineering Department, for his advice on component requirements and selection for use in experimentation.

Barbara Furhman, of the Mechanical Engineering Department, for her assistance and advice concerning budget, expenses, and retailer selection.

Statia Canning, of the Mechanical Engineering Department, for her assistance in presentation day materials, poster printing details, and support during our last minute changes.

Abstract

As mobile devices continue to package an increasing number of features into smaller form factors, it is crucial to extend battery life, which could be achieved by thermal management. Efficient battery usage through overcoming thermal constraints enables the continuous improvement of device hardware and device computational power. Potential solutions to thermally manage lithium ion batteries currently exist, such as micro heat pipes, phase change materials, and nanocomposites. This project used analytical, computational, and experimental methods in parallel in order to identify characteristics of thermal management systems. These characteristics were investigated through the use of uncertainty analysis and one-dimensional models, solid modeling and finite element analysis for three-dimensional heat flow, and wireless control of a Peltier system through an Arduino board to control battery surface temperature. Results obtained can be used in the development of potential thermal management systems for incorporation in future portable devices.

1. Introduction

For consumers, battery life cycle is an important characteristic in choosing the ideal mobile phone for purchase. Battery life cycle has not grown as fast as the computational power as current generation mobile phones. Faster CPUs and larger displays coupled with high quantities of local and network communication have increased the power draw requirements placed on the phone battery. Typically, mobile phones use lithium ion batteries with their high energy density providing longer lifespan than alternative battery technology.

Lithium ion batteries, like all batteries, have an optimal temperature range in which the time of discharge is at a maximum. Too high or too low a temperature can affect the battery negatively, shortening the lifespan, or rate of discharge. It can be established that higher power draw from a phone will lead to higher internal temperatures, and potentially shorter the lifespan of the battery will be per discharge cycle. In theory, a possibly solution would be to restrict phone computational power until battery technology catches up. However, with high power consumption being a major sell for smartphones in the world today, the rate at which computational power in mobile phones is improving incredibly quickly. Separate focus needs to be placed on improvement of battery technology, in particular, a method or methods that can be used to control the temperature of the battery in order to increase the lifespan while not inhibiting the power consumption.

The goal of this project is to develop and design characteristics required of a thermal management system for lithium ion batteries for future recommendations of thermal management solutions. Potentially available passive thermal management solutions are highly complex systems, and so a thermal system was to be designed as a proof of concept, with focus placed on the thermal capabilities required in order to control the temperature of a lithium ion battery system to the optimal temperature, required for maximum discharge.

2. Objective

The objective of this project was to develop characteristics required of a thermal management system for lithium ion batteries for future recommendations of thermal management solutions. The final system design would be capable of efficiently extending battery life based on the developed characteristics of the thermal models in one and three dimensions, and thermal characteristics could then be further developed in future works.

3. Motivation

The motivation for the investigation of thermally managing lithium ion batteries stems from two arguments. Firstly, due to the demands in the market for more powerful devices, the need for longer lasting battery life becomes crucial. A battery that can last even as much as 10% longer than existing batteries today is a major improvement and shows the need for further development in the area of thermal management for batteries in mobile devices. Longer battery life means that the phone’s power consumption can be greater, which is in high demand from consumers. Secondly, there is the cost consideration that can be taken into account. As replacement batteries and phones become ever more expensive, reducing the frequency for purchase and replacement of these devices can be critical and highly beneficial for financially responsible individuals. In addition, while not present in the foreground, charging of lithium ion batteries does draw an amount of electricity from the power grid. Reducing charge time and frequency can reduce the long term costs of owning a mobile device utilizing a lithium ion battery. Even a small percent increase in battery life cycle, if charging cycles are monitored properly, can lead to significant savings.

4. Background

4.1. Lithium Ion Battery Operating Principles

4.1.1. Components of a Lithium Ion Battery

The components of a lithium ion battery and the means by which it operates during phone usage must be understood before work can continue. As a general definition, batteries are electrochemical devices which convert chemical energy into electrical energy or vice versa by means of controlled chemical reactions between a set of active chemicals. There are four key components to a lithium ion battery system: the positive electrode, also known as the anode, the negative electrode, known as the cathode, the separator and the electrolyte. A carbon material is used in the anode, and a metal oxide material containing lithium is used in the cathode. The separator-electrolyte interface in located in between the cathode and anode and separates the two charged entities. The electrolyte consists of a materials with low electronic conductivity, but notable ionic conductivity. [1] During operation, the anode and cathode have a voltage differential, and separation is required in order to prevent shorting out the battery.

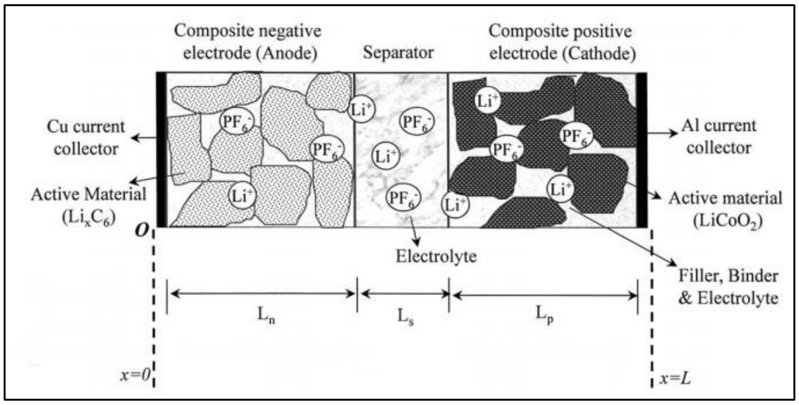

Composite negative electrode (Anode)
Separator
Composite positive electrode (Cathode)
Cu current collector
Active Material (Li_xC_6)
O
PF_6^-
Li^+
Al current collector
Active material ($LiCoO_2$)
Filler, Binder & Electrolyte
Electrolyte
L_n
L_s
L_p
x=0
x=L

Figure 1 - Schematic of typical lithium ion battery system using a carbon based anode, a cathode of $LiCoO_2$, and a separator of $LiPF_6$. [2]

In all batteries, lithium ion included, there is a charge and discharge cycle, during which the battery gains or releases power appropriately. When charging, lithium ions, charged atoms, separate from the electrons at the cathode and travel through the electrolyte due to an induced voltage differential between the cathode and electrode. Due to the high electrical resistivity of the electrolyte, electrons cannot pass through the electrolyte. Electrons therefore travel along a separate path, a wire in the case of a battery, to the anode. The lithium ions then merge between the layers of carbon at the anode. This process of the lithium ions merging into the carbon is known as intercalation. The differential storage of electrons and lithium ions creates a notable *chemical* potential difference between the cathode and anode. [1] During discharge, the potential is utilized, and electrons and lithium ions flow in the opposite direction. As stated earlier, electrons travel through a separate conductor to the cathode. Simultaneously, lithium ions separate themselves from the graphite layers at the anode, and return through the electrolyte to the cathode where they reform with the metal oxide material. The flow of electrons through the separate conductor generates electric current. The chemical equations for charge and discharge in the lithium ion battery system at the anode and cathode are shown below:

$$At\ anode: C + xLi_{+} + xe_{-} \leftrightarrow CLi_{x}$$

$$At\ cathode: LiCoO_{2} \leftrightarrow Li_{1-x}CoO_{2} + xLi_{+} + xe_{-}$$

Equation 1 - Chemical equations for charging and discharge at the cathode and anode

The motion of electrons between cathode and anode is the root generation of electrical current within a battery. The flow of ions and electrons is controlled within the system, but exterior influences can play a role in operational efficiency. Various environmental factors influence the flow of charged particles in the battery system.

4.1.2. Modeling the Electrochemistry of the Lithium Ion Battery System

In order to understand which environmental factors can influence battery efficiency, results were obtained from a modified lithium ion battery model contained in the COMSOL Multiphysics software package. Additions to the included lithium ion battery model were created by Long Cai and Ralph E. White of the Department of Chemical Engineering at the University of South Carolina [3]. The new model couples a two-dimensional representation with thermal and electrochemical models.

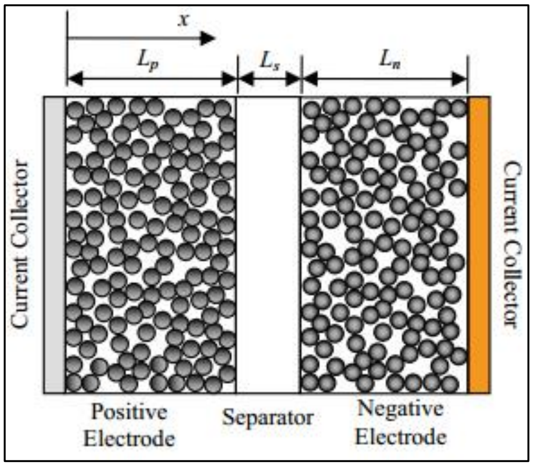


Figure 2 - Lithium Ion battery schematic utilized in COMSOL modeling software with appropriate components [3].

Contained in the model are components standard in any lithium ion battery: a positive and negative current collector, a positive electrode, a negative electrode, a separator, and an electrolyte. The negative current collector is copper and the positive current collector is aluminum. The negative electrode is Li_xC_6, a lithium graphite composite, and the positive electrode is $Li_xMn_2O_4$, or lithium manganese oxide [4] [5]. The separator-electrolyte interface is composed of $LiPF_6$, lithium hexafluorophosphate, and EC/DMC, ethylene carbonate with dimethyl carbonate [6]. An organic solution treated with lithium serves as the electrolyte. These material choices are reflective of lithium ion batteries used in mobile devices.

Cai and White's model makes a number of assumptions and references various thermodynamic and heat transfer laws in order to acquire the desired end result [3]. The mathematics begins with Fick's second law in spherical coordinates, serving as the mass balance:

$$\frac{\partial c_{s,i}}{\partial t} = D_{s,i} \frac{1}{r^2} \frac{\partial}{\partial r} \left(r^2 \frac{\partial c_{s,i}}{\partial r} \right)$$

Equation 2 - Mass balance

That is, the change in the concentration of lithium ions in a solid material particle $\left[\frac{mol}{s \cdot m^3}\right]$ is a function of the diffusion coefficient of lithium ions in a solid electrode $\left[\frac{m^2}{s}\right]$, the distance from the center of the particle $[m]$, and the change in concentration of lithium ions with respect to distance from the center of the particle $\left[\frac{mol}{m \cdot m^3}\right]$. For the positive electrode or negative electrode, $i = p$ or $i = n$, respectively. The mass flux at the center of the particle is assumed to be zero and the mass flux on the surface of the particle is equal to the rate at which lithium ions are produced or consumed due to the electrochemical reaction occurring $\left[J_i, \frac{mol}{m^2 \cdot s}\right]$, shown below in Equation 3.

$$\text{At } r = 0\text{: } -D_{s,i} \frac{\partial c_{s,i}}{\partial r} = 0 \text{ and at } r = R_{s,i}: -D_{s,i} \frac{\partial c_{s,i}}{\partial r} = J_i$$

Equation 3 - Mass flux constituents

The mass balance for the electrolyte exhibits similar properties, dependent on the porosity of the electrode and separator, the specific interfacial area $\left[\frac{m^2}{m^3}\right]$, and the mass flux of lithium ions from the electrode $\left[\frac{mol}{m^2 \cdot s}\right]$. At the two ends of the cell, $x = 0$ and $x = L_p + L_s + L_n$, there is assumed to be no mass flux. At the boundaries between the electrodes and separator, both the concentration of lithium ions and the flux of lithium ions are assumed continuous.

In tandem with the mass balances, the charge balance in the solid phase is represented by an expansion of Ohm's law:

$$\sigma_{eff,i}\frac{\partial^2\phi_{1,i}}{\partial x^2}=a_iFJ_i$$

Equation 4 - Ohm's law expansion for current flow in lithium ion battery system

The electrical conductivity of the electrode $\left[\frac{S}{m}\right]$ multiplied by the second derivative of the potential $[V]$ with respect to x is equivalent to the specific interfacial area $\left[\frac{m^2}{m^3}\right]$, Faraday's constant $\left[\frac{C}{mol}\right]$, and the flux of ions $\left[\frac{mol}{m^2\cdot s}\right]$.

Between the current collector and the positive electrode, the charge flux is equivalent to the applied current density:

$$-\sigma_{eff,p}\frac{\partial\phi_{1,p}}{\partial x}=I_{app}$$

Equation 5 - Current density parameters

The interface between both electrodes and the separator have a net zero charge flux. The potential at the negative electrode is set to zero and the potential at the positive electrode is equivalent to the cell voltage, E_{cell}.

In the liquid electrolyte, the charge balance is also based on Ohm's Law:

$$-\frac{\partial}{\partial x}\left(\kappa_{eff,i}\frac{\partial\phi_{2,i}}{\partial x}\right)+\frac{2RT(1-t_+^0)}{F}\frac{\partial}{\partial x}\left(\kappa_{eff,i}\frac{\partial lnc_i}{\partial x}\right)=a_iFJ_i$$

Equation 6 - Further expansion of Ohm's law to include charge balance

Here it is important to note that $\kappa_{eff,i}$, the specific conductivity of the electrolyte, is a function of the concentration of the electrolyte and temperature:

$$\kappa_{eff,i} = \kappa_i \varepsilon_i^{brugg_i}$$

where

$$\kappa_i = c_i(-10.5 + 0.668 \times 10^3 c_i + 0.494 \times 10^6 c_i^2 + 0.074T - 17.8c_iT - 8.86 \times 10^2 c_i^2 T - 6.96 \times 10^{-5} T^2 + 2.80 \times 10^{-2} c_i T^2)$$

Equation 7 - Specific conductivity of the electrolyte showing temperature dependence

It is at this stage in the electrochemistry that the temperature dependence of battery operation is observed. Variations in the temperature of the battery will affect the rate at which lithium ions will flow between the cathode and anode. This will affect the outputted current, and ultimately will influence the duration of discharge. This relation is the ultimate electrochemical motivation for thermal management.

Rounding out the electrochemistry, the final conservation equation, the energy balance, is also of particular interest:

$$\rho C_p \frac{dT}{dt} = \lambda \frac{\partial^2 T}{\partial x^2} + Q_{rxn} + Q_{rev} + Q_{ohm}$$

Equation 8 - Final energy balance with heat generation terms

The heat generation terms[$Q_{rxn}, Q_{rev}, Q_{ohm}$] correspond to the total reaction heat generation rate, the total reversible heat generation rate, and the total ohmic heat generation rate. They are defined by:

$$Q_{rxn} = FaJ(\phi_1 - \phi_2 - U)$$

$$Q_{rev} = FaJT \frac{\partial U}{\partial T}$$

$$Q_{ohm} = \sigma_{eff} \left(\frac{\partial \phi_1}{\partial x}\right)^2 + \kappa_{eff} \left(\frac{\partial \phi_2}{\partial x}\right)^2 + \frac{2\kappa_{eff} RT}{F}(1 - t_+^0)\frac{1}{c}\frac{\partial c}{\partial x}\frac{\partial \phi_2}{\partial x}$$

Equation 9 - Expansion of heat generation terms within the lithium ion battery system

The ohmic heat generation term is of the utmost importance. In current smartphones, increased RAM and processor speed allows for additional processes. However, this means additional current draw from the battery, leading to increased ohmic heat generation. These heat generation terms are a component of the overall heat generation within the mobile phone system. It is important to note, that the heat generation within the battery is comparatively small to heat generation in other components. However, it is the sum of all heat generation terms that ultimately influence the rate of discharge in the It is the overall heat generation that is to be opposed and overcome in order to maintain optimal battery temperature.

The modified COMSOL model utilized the aforementioned equations with simple geometry to represent the electrodes and separators. The model discharges the battery first for 300 seconds at C/2 rate, then at 3C rate until the cell voltage reaches 2.5V, as shown in Figure 3.

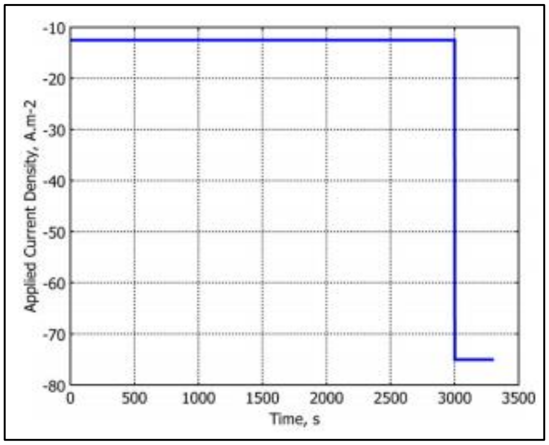


Figure 3 - Current density profile for discharge process [3]

Among the various outputs of the COMSOL simulation is the relation between time and cell surface temperature. Reported were five separate trials which varied the cooling conditions on the cell surface. The cell was simulated with three different cooling conditions, with the heat transfer coefficient between the cell surface and the environment valued at 10.0, 1.0 and 0.1 $\frac{W}{m^2K}$.

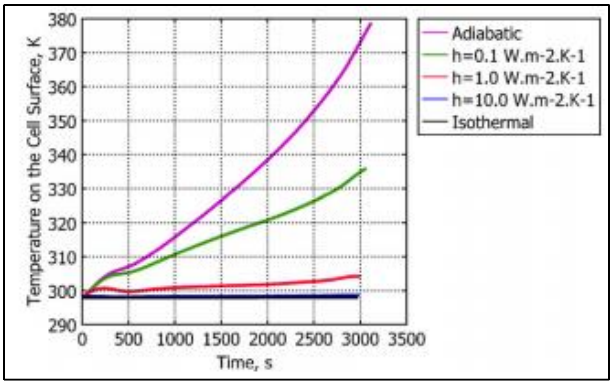


Figure 4 - Temperature on the cell surface during discharge under different cooling conditions [3]

Adiabatic as well as isothermal conditions were also tested as limiting conditions. As shown, in the adiabatic condition, as expected, no heat is transferred to the system surroundings and the cell temperature increases rapidly, nearing a maximum of 380K. In the isothermal simulation, the cell surface temperature stays constant. With increasing heat transfer coefficient, more heat can be dissipated to the environment and the cell peak temperature is reduced. By this data, is can be deduced that with an improved cell surface cooling system, additional heat will be dissipated to the surroundings, and the internal temperature of the cell can be reduced during discharge.

However, it should be noted that during discharge, it is important to increase cell temperature quickly in order to maximize diffusion of ions. The diffusion coefficient, as previously mentioned, is dependent on temperature and achieving a maximum diffusion coefficient provides for the greatest flux of ions, and therefore the maximum possible cell voltage. As seen above in Figure 4, under adiabatic conditions, the cell surface temperature is always above that of isothermal, and higher heat transfer coefficient conditions. With these higher cell surface temperatures, the diffusion coefficient reaches a maximum much faster. This leads to improved concentration of the electrolyte at the end of the discharge cycle, as shown in Figure 5.

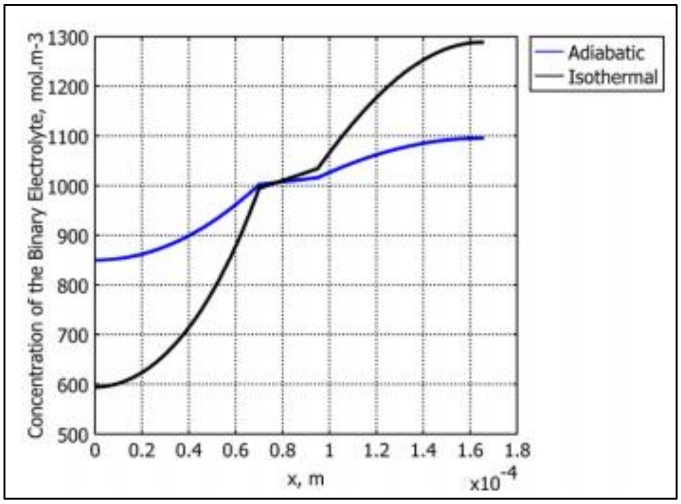


Figure 5 - Concentration profiles of the electrolyte at the end of the 3000s discharge [3]

The concentration profile for the isothermal case has a much higher range, illustrates reduced diffusion of ions. The adiabatic curve with its smaller range indicates improved diffusion, resulting in more similar concentrations across the two electrodes and the separator.

The higher temperatures under adiabatic conditions increase in ionic diffusion leads to increased cell voltage. By reaching a critical temperature faster, ion diffusivity reaches a maximum. However, the cell temperature should not be allowed to surpass certain temperatures. When cell temperatures reach specific levels, the lithium ions can interact with the positive electrode material and change phase. A set of experiments conducted by Di Chen of the

Karlsruher Institut für Technologie investigated the mechanical interactions that occur within the spinel structure in lithium ion batteries.

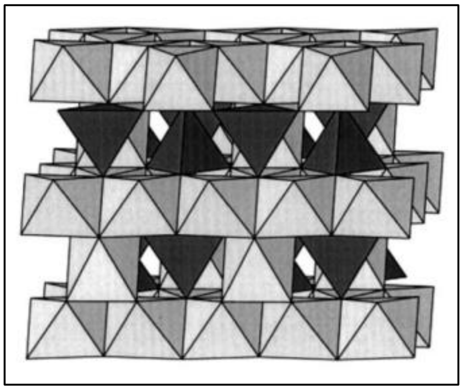


Figure 6 - Spinel structure of LiMn2O4 [7].

The spinel structure shown in Figure 6 is that of $LiMn_2O_4$, the exact material used in Cai and White’s model. Chen’s study examined the microscopic interactions in the spinel structure during the discharge and life cycle of a lithium ion battery utilizing $LiMn_{1.95}Al_{0.05}O_4$ [7]. It was found that during discharge, the lithium ions react with the tin during insertion into the lattice. The Li-Sn combination reacts with oxygen in the electrode material, changes phase, and expands in volume, up to 3.6 times the original volume. This volume expansion stresses the spinel structure and eventually leads to cracks and failure.

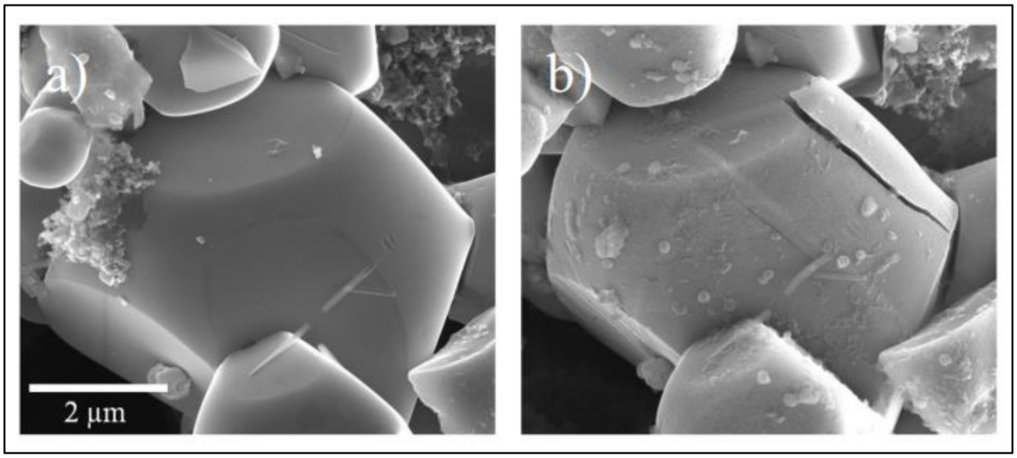


Figure 7 - Images of spinel particles (a) uncycled and (b) after 800 cycles [7].

Chen cycled a cell 800 times and observed the mechanical failures in the lattice due to volume expansion of the Li-Sn combination. Shown in Figure 7, after 800 cycles cracks appeared in the spinel structure. These cracks reduce the efficiency the cell discharge by reducing the available space for lithium ion insertion.

In combination with Cai and White's COMSOL model, it can be concluded that higher cell temperatures provide more efficient discharge, but can lead to mechanical failure due to spinel structure cracking. Ideally, a cell's peak temperature will be reached quickly, but not be allowed to surpass the threshold required for phase change during lithium insertion. This could result in maximum diffusion coefficient, and therefore maximum cell discharge voltage, while reducing the damaging effects of volumetric expansion during lithium ion insertion.

The electrochemistry points towards a temperature dependence in the flow of ions and electrons in the lithium ion battery system. This is the stepping off point for this project and directs the project towards incorporation of a thermal management system in order to maximize battery life cycle per discharge.

4.2. Potential Passive Thermal Management Systems

With the established temperature dependency, attention is turned towards potential thermal management solutions. Selection of a passive solution was chosen in order to not place additional loads on the battery. Currently available are three notable systems that are effectively used in high density electronic packaging: micro heat pipes, nanocomposite materials, and phase change materials.

4.2.1. Micro Heat Pipes

Micro heat-pipe (μHP) technology works on the principle of conducting heat through the evaporation and condensation cycle of vapor within a hollow cylinder. A thermally conductive tube encases the liquid and vapor forms of a fluid of choice. At one end of the pipe is the heat source and the other end becomes the desired dissipation location. Heat influx at one end evaporates the interior liquid into a vapor form. The vapor relocates to the alternate end and condenses into a liquid, releasing heat energy. [8]

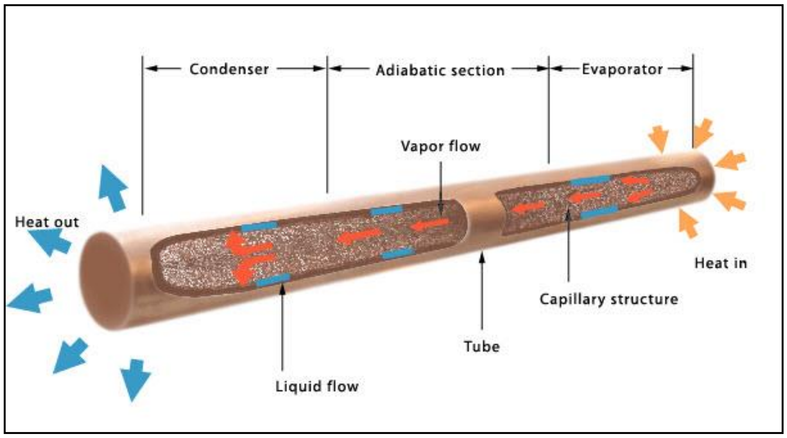


Figure 8 - Heat pipe showing evaporation and condensation sections, fluid/vapor flow, and heat transfer locations.

Heat pipes are ideal systems for thermal management in mobile devices. Due to the nature of the evaporation-condensation, the system can operate on a small scale, more appropriate for mobile devices. Micro scale heat pipes have diameters of 100-1000 μm and lengths of 10-20 mm. This size allows a full heat pipe system to fit comfortably on the rear surface of a mobile phone. The small diameter could even allow internal installation within the phone or battery casing.

The heat extraction of a heat pipe system is a complex thermofluid system, controlled by many attributes of the fluid, pipe material, and pipe geometry. Shown below in Equation 10 is the theoretical maximum heat extraction of a heat pipe system.

$$\dot{Q}_{max} = cf \frac{0.16 \cdot \beta\sqrt{K_l K_v}}{8\pi(0.5L_e + L_a + 0.5L_c)} \frac{\sigma h_{fg}}{v_l} \sqrt{\frac{v_l}{v_v}} A^{\frac{3}{2}}$$

Equation 10 - Maximum possible heat extraction capable by a micro heat pipe system

The system is highly complex, as expected from a system governed by the change of a phase of a liquid within a mechanical pipe system. Notable variables include β, the liquid shape parameter, K_l and K_v, the dimensionless liquid and vapor shape factors, σ, the surface tension, h_{fg}, the latent heat of vaporization, v_l and v_v, the liquid and vapor kinematic viscosities, cf, a correction factor, as well as geometric dimensions, A, L_e, L_a, and L_c, the cross sectional area, and lengths of the evaporator, adiabatic, and condenser sections appropriately. [8]

While governing by a number of highly complicated equations, micro heat pipes are still thermally capable of dissipating the heat generated in mobile systems. At working temperatures nearing the boiling point of water, 100°C, micro heat pipe arrays approach dissipation levels of 20 watts. Heat pipes are a passive system, as required, drawing no additional power from the battery. In addition, heat pipes are static systems with no moving components, operate with no noise generation, and are highly reliable. [8]

4.2.2. Nanocomposite Materials

A second passive thermal management option is nanocomposite materials. Thermal management in nanocomposites operates via the phenomena of increasing thermal conductivity. This can work with existing structures in a mobile device casing to increase heat flow away from the components and battery, and direct heat flow into the environment. Single carbon nanotubes can reach thermal conductivities of $103\,\frac{W}{m\cdot K}$, well above typical polymer levels, falling closer to $0.2\,\frac{W}{m\cdot K}$. [9] Differing materials and nanostructures can further increase thermal conductivities of nanocomposite material systems. Operating in nanoscale form factors, shown in Figure 9, nanocomposites can be integrated into many locations in a mobile device.

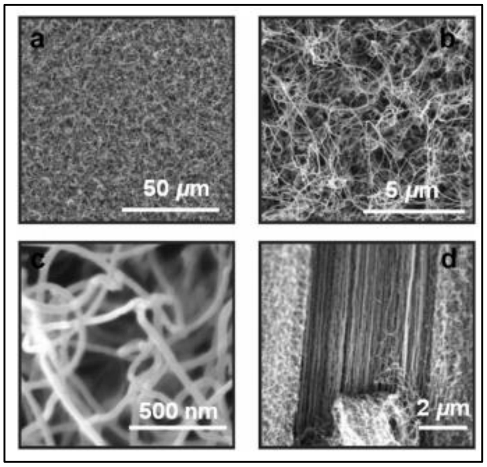


Figure 9 - SEM images of carbon nanotube structures. Top view at multiple scales (a) (b) (c) show entanglement of tubes. Side view (d) showing interior vertical alignment. [10]

Similar to micro heat pipe arrays, nanocomposites are a passive system, pulling no power from the mobile device's internal battery. Nanocomposite materials is a new and emerging technology, and its methods of operation are not entirely understood. However, eventual development of nanomaterial systems can be highly effective as thermal management solutions.

4.3.3. Phase change materials

The third and final potential thermal management system targeted was phase change materials. These systems operate on the principles of the change of phase of various materials for heat storage. Similar to the fashion by which vapor condenses into a liquid form in micro heat pipes, phase change materials alternate between solid and liquid forms. As heat energy is generated in mobile system components, the energy is slowly absorbed by a phase change material. In modern mobile phone systems, this can work to decrease the heat flow to the lithium ion battery. The specific heat capacity of phase change materials can be quantitatively high, reaching up to 45 $\frac{kJ}{kg \cdot K}$. [11]

Again, as required, phase change materials are a passive system. While needed a larger volume and mass as compared to micro heat pipes and nanocomposites, phase change materials are less expensive and could possibly be the ideal thermal management solution required to combat the temperature differential generating in a mobile device system.

4.3. Phone Power Draw Parameters

Within a mobile phone, the temperature differential is created within the system itself. Nearly all of the components on the printed circuit board, and located elsewhere in a mobile phone produce some quantity of heat energy. The amount of energy generated is dependent on component function, utilization by applications and the operating system, and hardware efficiency.

4.3.1. CPU Power Consumption

A large percentage of heat is generated by a phone's central processing unit (CPU). The CPU is responsible for many of a mobile phones calculations and data processing. The CPU manages processes such as the operating system, back end code for applications, and communication between applications and other components. In some mobile phones, a separate graphics processing unit is not present, and the CPU is also tasked with creating and rendering all graphics.

Power consumption in CPUs is broken down into three main components. Dynamic power is representative of power that is consumed when logic gates within the CPU are opened and closed during operation. The opening and closing of logic gates is how the CPU interprets and interacts with data. Interactions between logic gates and associated capacitors are the basis for how power is consumed and dissipated. Opening and closing a logic gate generates a finite amount of power. However, logic gates are opened and closed many times during operation. As such, the frequency at which the CPU is operating must be taken into account. [12]

CPU frequency is an integral part of CPU computing power and processor speed. Measured in hertz, or computations per second, CPU frequencies have become increasingly powerful. Many modern mobile device CPUs have frequencies above two gigahertz. That is, two times 10^9 operations per second. [13] With high clock speeds, and millions of logic gates integrated in CPU assembly, dynamic power is responsible for a significant portion of power in the CPU. Dynamic power can be represented as follows:

$$P_{dynamic} = C_L V_{DD}^2 f_{o\to 1}$$

Equation 11 - Dynamic power in CPU power consumption

C_L is the total capacitance on the chip, V_{DD} is the voltage consumed during the discharge of a capacitor, and $f_{0\to 1}$ is referred to as the “frequency of energy-consuming transitions,” and is directly related to the clock speed of the CPU. With advances in CPU manufacturing technology, C_L and $f_{0\to 1}$ continue to increase. Appropriately, $P_{dynamic}$ continues to rise with the increase in CPU power. [12] [14]

Second to dynamic power in CPU power consumption is static power. When logic gates are not being switched, power still flows within the circuitry. Also referred to as leak power, static power is ideally zero in a CPU system, and is governed by the following equation:

$$P_{static} = I_{Static} V_{DD}$$

Equation 12 - Static power in CPU power consumption

I_{static}, also referred to as I_{leak}, is the current that flows while the system is idle. This equation is a simple extension of Ohm’s Law. [12]

The final portion of CPU power consumption is called direct-path power dissipation. Dynamic power consumption assumes zero time for the opening and closing of logic gates. Unfortunately, this is not the case in a real system, and the rise and fall time of the voltage differential is represented in direct-path power dissipation. [14] Being similar in nature to dynamic power, the equation for direct-path power dissipation is dependent on similar conditions:

$$P_{dp} = t_{sc}V_{DD}I_{peak}f_{0\to 1}$$

Equation 13 - Direct-path power dissipation in CPU power consumption

Similar to dynamic power, the direct-path power is directly related to the speed at which the processor operates, $f_{0\to 1}$. V_{DD} , the voltage consumed during discharge, is again present, and I_{peak} represents the peak current that is to flow through the logic gate. The t_{sc} term is referred to as the transition time, and represents the rise and fall of the current flowing through the logic gate. [12]

The total power consumption within a CPU can now be expressed as a combination of dynamic power, static power, and direct-path power. [12]

$$P_{total} = P_{dynamic} + P_{static} + P_{dp}$$

$$P_{total} = C_L V_{DD}^2 f_{0\to 1} + I_{static}V_{DD} + t_{sc}V_{DD}I_{peak}f_{0\to 1}$$

Equation 14 - CPU total power consumption

Actual CPU power consumption varies by processor. Processor technology is incredibly varied and faster, more efficient CPUs are available in current generation mobile phones.

4.3.2. Screen Power Consumption

Along with the CPU, a mobile device's screen contributes a high portion of power consumption. Screen power consumption is related to brightness level, and the primary color distribution. Appropriately, higher brightness levels consume more power. Shown in Figure 10, an exponential relation is present. [15] While relatively simple in nature, screen power consumption is very high quantitatively.

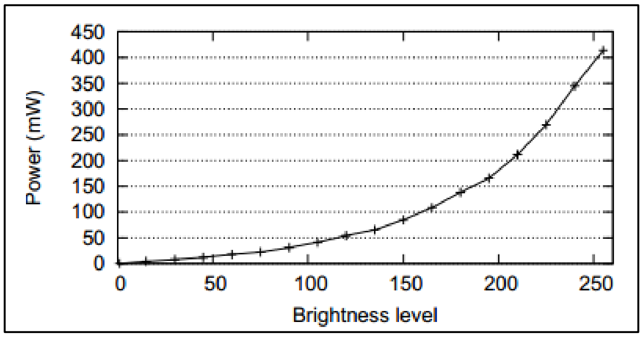
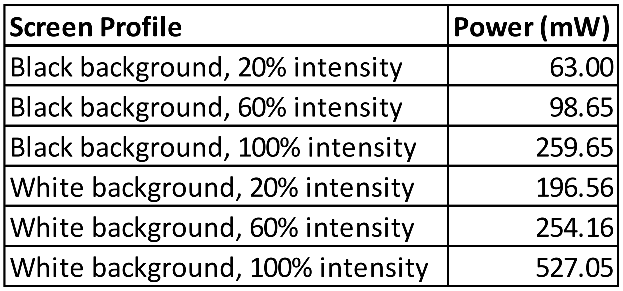


Figure 10- Display backlight power consumption as a function of brightness

In addition to being dependent on brightness, screen power consumptions is dependent on the general color scheme. Differences between pixel display colors require different amounts of energy: a darker background requires less energy than a lighter background. Shown in Table 1 is a general breakdown of screen power consumption as a function of different background colors. Indicated is that white backgrounds require more power. This is connected to the operating principle of the pixels contained within the display, as darker colors can be achieved by keep select pixels entirely deactivated, drawing little to no power in the process. [16]

Table 1 - Screen power consumption



Screen Profile	Power (mW)
Black background, 20% intensity	63.00
Black background, 60% intensity	98.65
Black background, 100% intensity	259.65
White background, 20% intensity	196.56
White background, 60% intensity	254.16
White background, 100% intensity	527.05

4.3.3. Other Components Power Consumption

Other components in a mobile phone system also draw power from the battery. For wireless communication there is the GSM antenna, for satellite communication, as well as 2G, 3G, 4G, and 4G LTE antennas for connecting to satellite networks with higher transfer speeds.

For local connections there is the WiFi antenna and the BlueTooth module. Hardware components include storage, audio codecs, memory, and sensors for acceleration or orientation. Some high end mobile phones also have components for displaying 3D images, higher end cameras, more powerful speakers, and near field communication. All of these specialized components draw power on a usage basis. For example, during video playback, the CPU and screen power draw will stay relatively static, but storage and audio will receive increased power in order to carry out their respective functions. All the power draw from these components must be treated dynamically, in accordance with the functionality of the device.

4.4. Preliminary Experimentation

Before in depth experiments were to be performed on the battery, preliminary experiments were needed in order to verify the link between time of discharge and temperature of the lithium ion battery. In preliminary experiments, a lithium ion battery within a mobile phone was to have its temperature finely controlled using a ThorLabs TED 350 Temperature Controller. The controller used a thermoelectric material, or Peltier surface, with an internal thermistor to actively control temperatures as governed by PID control. Further explanation of PID control will be developed later in this project. In order to conduct these experiments, knowledge of where to place a Peltier surface on the battery was necessary. The most efficient location was determined to be the exterior surface of the battery, relative to the screen and remainder of components. This was directly adjacent to the CPU-frame-battery interface, and works well to dissipate the heat generated by the CPU. With the rear casing of the mobile phone removed, it was simple to attach a Peltier at the desired location. Upon establishing this information, a Peltier could then affixed using thermal paste to increase thermal conductivity.

Next, the ThorLabs TED 350 temperature controller’s exterior wiring was modified to use an external thermistor and Peltier surface. A ThorLabs Peltier-thermistor module was unavailable, and modifications were only required upon a component connection wire. Using the CAB420-15 db15 cable that connected the TED350 to outputs, the wire was stripped down and the internal wires were exposed. Proper wire selection was required for attaching the chosen

Peltier surface and external thermistor, the LM35, one of two types accepted by the TED 350. Important wiring diagrams are shown below in Figure 11, Figure 12, and Figure 13.

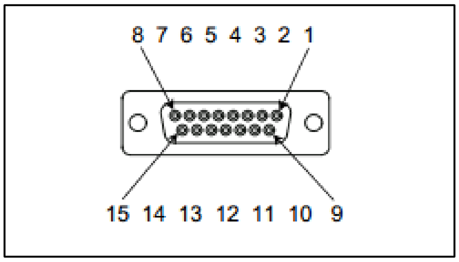


Figure 11 - 15-pole female D-SUB on CAB420-15 db15 cable, rear panel view.

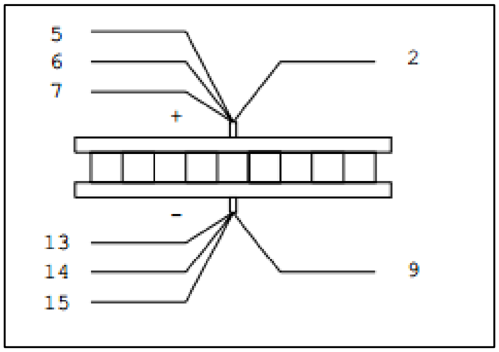


Figure 12 – Peltier surface connection map.

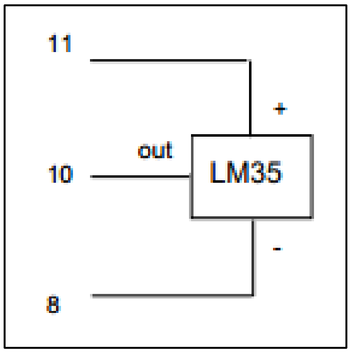


Figure 13 - LM35 thermistor connection map.

The TED 350 output current to the Peltier surface was set to 1.202 Amperes, and the target temperature was arbitrarily set to 25.0, 26.0, 26.5, 27.0, 27.5, 28.0, 28.5, 29.0, 30.0, 31.0, 33.5, and 36.0 degrees Celsius. The PID settings on the TED350 were then adjusted to ensure that the TED 350 could accurately manage the battery temperatures within 0.5 degrees Celsius of the target temperature. The 0.5°C tolerance falls within the range of +/- 0.6 the LM 35 can record in relations to the surface temperature of the battery. The purpose of the thermistor was to monitor the external temperature of the battery during the experiments. The thermistor was placed on the opposite side of the battery, also attached with thermal paste, in order to identify the external temperature once the Peltier was activated during the experiments.

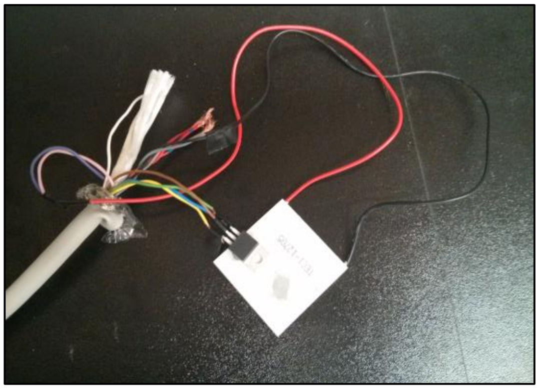


Figure 14 - Modified Peltier control surface showing Peltier (white) and LM 35 thermistor (black).

The experiments provided the discharge time for the battery as a function of battery temperature. This information was critical to the project as these relations would be necessary once more in-depth tests were performed. Those discharge times, along with the corresponding temperature are provided in Figure 15.

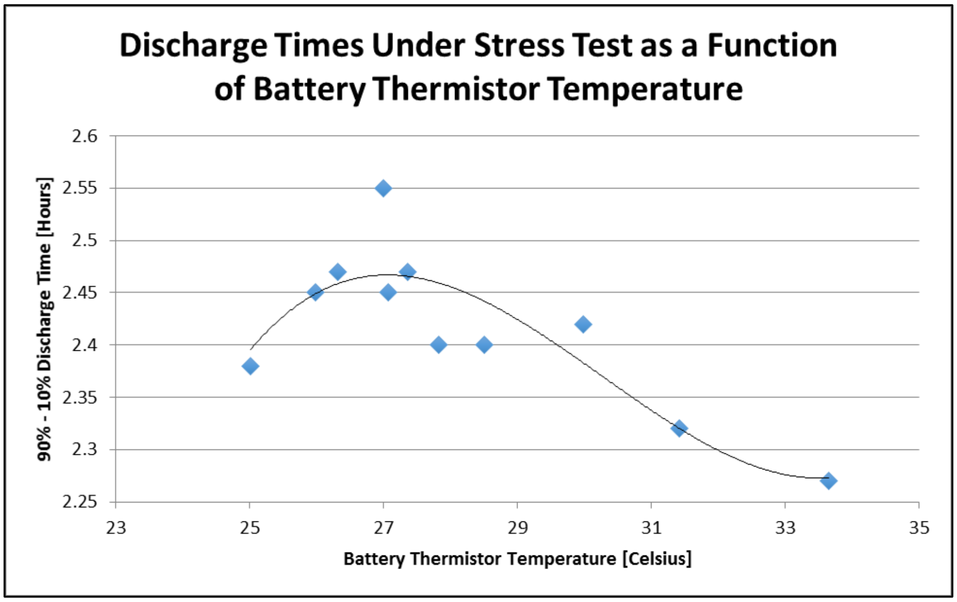


Figure 15 - Discharge times under stress test as a function of battery thermistor temperature

The tests indicated that at the temperature setting 27 degrees +/- 0.5 degrees Celsius, the battery sustained the longest time to discharge. The peak discharge time was recorded at 2.55 hours, or 175 minutes. Compared to the worst time observed, of 2.28 hours, or 139 minutes at 33.5 degrees, it was clear that a thermal management solution would be required in order to regulate the temperature of the lithium ion battery in order to achieve maximum discharge time. It should be noted, that there was error that was taken into account with preliminary experiments. Error is present in the LM35 thermistor's internal error due to the inherent tolerance in temperature readings. Error is also present in the precision of the method of the measurement of discharge times, and the mobile phone operating system was only capable of recording data every minute. In addition, error was caused by ambient temperature effecting the uninsulated sensor temperature reading. With a verified correlation between battery temperature and discharge, further research on lithium ion battery systems was required. Per manufacturer's specifications and recommendations, the optimal temperature for maximum discharge was chosen to be 30 degrees +/- 1.0 degree Celsius.

5. Project Goal

The goal of this project was to develop characteristics required of a thermal management system for lithium ion batteries in order to properly regulate battery temperature to the optimal 30°C. The final system design would be capable of efficiently extending battery life based on the developed characteristics of thermal models in one and three dimensions, and thermal characteristics could then be further developed in future works.

6. Constraints

Initial work began with establishing constraints for a potential thermal management system. The system had to practical for use in a mobile phone system as well as capable of dissipating the appropriate heat requirements. In addition, restrictions were in place in terms of the hardware and software capabilities of mobile phone systems, should they be required for use in an active thermal management system.

6.1. Thermal Constraints

Thermal constraints pertained to the heat capacity required of the thermal cooling system. The system had to be capable of dissipating the required amount of heat in a reasonable amount of time. The system would need a capacity of a precise level. Overcooling or undercooling the battery system could lead to ineffective operating temperatures, restricting the ability to reach maximum discharge time.

6.2. Electronic Constraints

Electronic constraints consisted of the restrictions in place by the mobile phone's operating system and ability to control the heat generation in the componentry. Precise control of the heat generation would remove variables during experimentation and could be used to obtain more accurate results. As the CPU was targeted as producing the highest percentage of thermal energy, managing the clock speed, or frequency, of the CPU was chosen as the best method for stabilizing heat generation. This control was achieved by applying various loads to the phone via a series of stress test applications and written back end code. For this project, a maximum load, 50% load, and minimum load were selected.

6.3. Physical Constraints

Physical constraints pertained to the dimensions of the battery surface and the physical area adjacent to the battery. The thermal management solution applied to the battery needed to be within controllable physical dimensions in order to be a feasible attachment for a mobile system. If thermal characteristics determined would require use of a thermal management system with an exceedingly large volume, an alternative system would need to be developed. A thermal management system also needed to be placed at the location where the highest quantity of heat was being generated. This would maximize the efficiency of the solution, dissipating the required quantify of heat while requiring the least physical volume.

7. Methodology

Due to the complex nature of phase change materials, nanocomposites, and micro heat pipes as potential thermal management solutions, the methods utilized in the project are not focused around specific solutions. Instead, methods targeted the characteristics required of such thermal management systems in order to provide important data that would be necessary to designing a thermal management solution utilizing any of the potential solutions. In addition, experiments that were highly controlled, and more precise were done in order to obtain a verified connection between battery temperature and discharge time.

Obtaining discharge times with decay in milliampere-hours (mAh) requires measuring the current through a known load for a number of hours and performing a few simple calculations. This would involve measuring the battery output voltage via connecting the positive lead of a multi-meter to the A port on the battery, and then connecting the positive terminal of the battery to a resistor which then loops to the positive lead of the multi-meter. The negative multi-meter lead connects to the negative battery terminal. Recording the current several times over an hour for several hours, depending on the desired accuracy, can give the average current. Alternatively, the Battery Monitor Widget application available on the Google Play Store performs this calculation automatically and graphs the battery decay over time. Due to higher accuracy, and ease of use, the Battery Monitor Widget was chosen over use of a multi-meter. The battery monitor widget is also safer, as an incorrect resistor value may lead to fire if not under constant observation.

Consideration was also given due the fact that cell phones often report charge percentage with a relatively major degree of error because the computer chip within lithium ion batteries must be calibrated by fully discharging to ensure that the percentage shown is accurate. Note that a discharge of a lithium ion battery is not a full discharge because this would permanently damage the battery's computer chip, but it does allow for calibration. In addition, when a cell phone plugged into a charger reaches capacity, the battery stops accepting charge until it drops a few percent – which could cause a large margin of error when measuring the overall decay of a battery over time. To bypass this potential error, a battery charge cycle can be stopped when the battery is at "capacity" then the charge cycle can be restarted several times to reach a true maximum charge. This can reduce a battery's overall lifetime but will aid testing in determining accurate results with minimum error. With proper equipment to determine time of discharge and a thorough understanding of battery operation and sources of error, work began on methods to design and develop characteristics required for a feasible, effective thermal management solution.

7.1. Analytical Methods: 1D and 2D Modeling

Before proceeding with experimentation, it was first necessary to investigate the lithium ion battery system from a two-dimensional standpoint, or more appropriately, understanding a one-dimensional heat flow model. In analyzing the battery's one-dimensional heat transfer characteristics, it was necessary to choose the proper equations to produce the desired results. For the purposes of this project, a transient analysis of the lithium ion battery is required. The battery temperature is not only a function of position in the battery system, but also a function of time. When inactive, phone components generate significantly less heat energy. As such an initial temperature and initial conditions were taken into account. Since battery warms up to an equilibrium temperature, a second order differential equation was necessary.

$$\rho c_p\left(\frac{\partial T}{\partial t}\right)=\left(\frac{\partial}{\partial x}\right)*k\left(\frac{\partial T}{\partial x}\right)+\dot{q}$$

Equation 15 - Second order differential governing heat flow in a transient solution

In the above equation, ρ represents the density of the control volume, the lithium ion battery, c_p represents the specific heat; multiplied by the partial of temperature over time equaling the partial with respect to x, multiplied by the thermal conductivity of the battery, k, multiplied by the partial of temperature over the partial of x, signifying the change in temperature over a specific point, plus the heat generation, labelled as $\dot{q}$. With this equation as the basis, next was to define the boundary conditions for the lithium battery, which are shown in Figure 16.

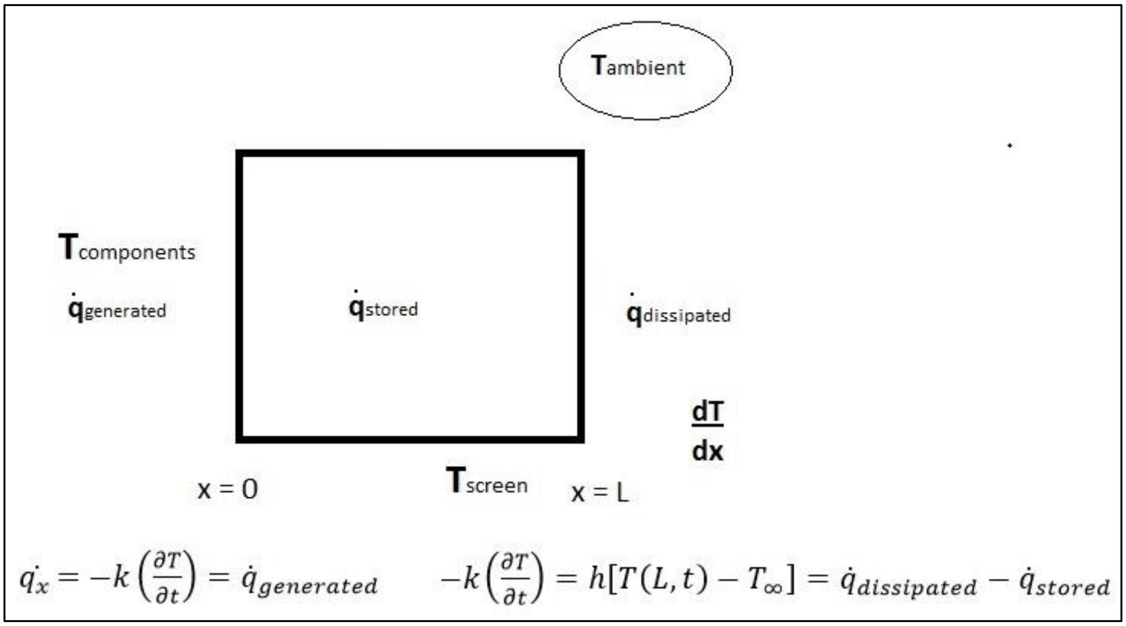

$T_{ambient}$
$T_{components}$
$\dot{q}_{generated}$
$\dot{q}_{stored}$
$\dot{q}_{dissipated}$
$\frac{dT}{dx}$
x = 0
T_{screen}
x = L
$\dot{q_x}=-k\left(\frac{\partial T}{\partial t}\right)=\dot{q}_{generated}$
$-k\left(\frac{\partial T}{\partial t}\right)=h[T(L,t)-T_\infty]=\dot{q}_{dissipated}-\dot{q}_{stored}$

Figure 16 - Differential Element of Battery with Boundary Conditions

As indicated from the differential element of the two-dimensional view of the battery, at x equal to 0, the thermal conductivity multiplied by the partial of temperature over time is equal to the heat generated. At x equal to L, the conductivity equates to the convection, which equates to the stored heat in the battery subtracted from the dissipated heat from the side. These boundary conditions fit the purposes of the experiment, so the next step is to further investigate the second order equation to find this temperature change over time. The equation first approached to solve the time necessary for the battery to reach 30 degrees is as follows.

$$T(x,t) = T_\infty + \dot{q}_i/k * \left[2\sqrt{\frac{\alpha t}{\pi}} * \mathrm{e}^{\left(-\frac{x^2}{4\alpha t}\right)} - x\left(1 - \mathrm{erf}\left(\frac{x}{2\sqrt{\alpha t}}\right)\right)\right]$$

Equation 16 - Transient heat equation for the lithium ion battery system

The initial temperature set to 23 degrees Celsius, the generated heat, labeled as $\dot{q}_i$ equating to 1.284 W from mathematical analysis, the thermal conductivity of k equating to 0.66 $\frac{W}{m \cdot K}$, and the thermal diffusivity α equating to 2.163x10^{-5} $\frac{m^2}{s}$. For the purposes of the analytical work, *t*, the time it would take for the battery to reach 30 degrees, was isolated and an arbitrary value for x was selected to represent a point on the battery. From numerous trials using this equation, it was evident that the value for t to satisfy the problem could not be reached, as heat loss at the end of the battery, when x = L, had to be taken into account, thus all data would indicate a gradual rise in temperature before a sharp increase, contradicting analysis performed in prior experimentation. The issue arose with analytical work was that in the two dimensional system of equations for one dimensional heat flow, there was not enough available required information to solve the equation accurately. It was clear that another method would be needed for solving.

Upon establishing that prior methods would not be satisfactory, a lumped capacitance method was approached, which considered the battery reaching a temperature uniformly. The benefits of the lumped capacitance method is that it greatly simplifies the problem to be solved, which in this case was the heat flow of the battery. The equation used to solve for the temperature is as follows.

$$T(t) = T_\infty + (T_0 - T_\infty) \cdot \exp\left(\frac{-hA_s}{\rho c_p V} t\right)$$

Equation 17 - Lumped capacitance method temperature as a function of time

Upon rearranging to solve for *t*, the amount of time in seconds required for the battery to reach the specific temperature, the equation then becomes as follows:

$$t = \frac{\rho \cdot c_p}{h} \cdot \frac{V}{A_s} \cdot \ln\left[\frac{T_\infty - T_0}{(T_\infty - T)}\right]$$

Equation 18 - Time to reach equilibrium in the lumped capacitance method

Initial temperature, T_0, was chose to be 23 degrees Celsius, and the maximum temperature $T_\infty$was chosen to be 32 degrees Celsius. The temperature that was desired, *T*, was chose to be 30 degrees Celsius. The remaining values were all obtained from data researched on lithium ion batteries, including the battery involved in the experiment: a density of 2100 $\frac{kg}{m^3}$, a specific heat of 795 $\frac{J}{kg \cdot m^3}$, a volume of the battery of 1.2948x10^{-5} m^3, a surface area of the battery of 2.099*10^{-4} m^2, and finally the convection coefficient estimated at 5 $\frac{W}{m^2 \cdot K}$. Initial solving revealed a time of 2250 seconds for the battery to reach 30 degrees. Knowing that there was freedom to manipulate values, such as the density, specific heat, and the convection coefficient, some parameters were changed. The density was then chosen to be 1900 $\frac{kg}{m^3}$, and the convection coefficient to 12.0 $\frac{W}{m^2 \cdot K}$, two and a half times the previous amount. All other values remained the same. Solving with these parameters, rise time was solved to be near 600 seconds, or 10 minutes for the battery to reach 30 degrees. This was very similar to preliminary experimentation. The hypothesis was then established that if the convection value, h, could be increased, the time to reach a desired temperature would decrease.

Upon finding a time that closely resembled the effects we desired, it was then determined to perform an uncertainty analysis to indicate which of the parameters that could be changed would have the greatest impact on the convection of the battery. The equation used to perform the uncertainty analysis for the battery was as follows.

$$\delta Q_c = \left[\left(\frac{\partial Q_c}{\partial h}\delta h\right)^2 + \left(\frac{\partial Q_c}{\partial A_c}\delta A_c\right)^2 + \left(\frac{\partial Q_c}{\partial T_s}\delta T_s\right)^2 + \left(\frac{\partial Q_c}{\partial T_e}\delta T_e\right)^2\right]^{\frac{1}{2}}$$

Equation 19 – Uncertainty analysis equation of choice

In this equation, uncertainties for the convection coefficient, the area of the battery where the heat is being dissipated, the surface temperature and the environmental temperature are measured in relation to the heat dissipated, $\dot{q}_{dissipated}$. Using values pertaining to experimentation, the convection coefficient was shown to contribute the largest percentage to the overall uncertainty. Convection was measured to be on the order of 8.318 mW. The analysis showed that total uncertainty was equal to 12.303%, with the convection uncertainty making up almost two thirds of the uncertainty, at 64.81%. The surface temperature contributed the second largest percent uncertainty, totaling at 33.067%. The uncertainty of these two parameters make up the majority of the complications for this project, as there is no clear way of knowing the free convection that occurs in a mobile phone during experimentation. The uncertainty analysis further suggests that using finite element methods in a three dimensional model would better solidify these values.

Finally, it is important to investigate the one dimensional heat flow with a thermal management system attached. The diagram for this model is shown in Figure 17.

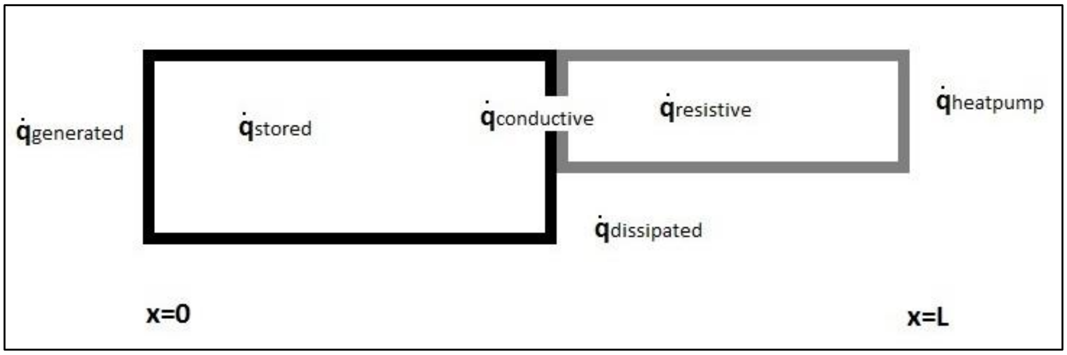


Figure 17 - One-dimensional flow with battery and thermal management solutions

With the governing equation of the system as follows:

$$\dot{q}_{generated} + \dot{q}_{conductive} = \dot{q}_{dissipated} + \dot{q}_{heatpump} - \dot{q}_{stored} - \dot{q}_{resistive}$$

Equation 20 – Heat equation for one dimensional system

From Figure 17, it is evident that the heat generated at x = 0, travels through the battery and then through the thermal management system, indicated in the light gray box, where it exits as $\dot{q}_{heatpump}$, and also through the remainder of the battery, labelled as $\dot{q}_{dissipated}$. Depending on the desired temperature, the thermal management system requires a higher heat capacity to dissipate out generated heat. For the purposes of the experiment, the heat capacity that is necessary to thermally manage the lithium ion battery to 30 degree Celsius is between 10 and 12 Watts.

It is evident from the one-dimensional analysis that there is a significant relationship between the convection conditions and the time required for the battery to reach to a desired temperature. It is also clear that the two dimensional system and one-dimensional heat flow analysis, while useful in providing numerical data, doesn't efficiently provide enough accurate information to understand the heat transfer that fully occurs. The mobile phone system is complicated, three dimensional system with many inputs, outputs, and heat transfer parameters, not solvable in one and two dimensions, and a more advance method was required.

7.2. Computational Methods: Finite Element Method Thermal Modeling

With one and two dimensional models being unsolvable and ineffective at providing required data, an alternative method was needed. A 3D solid model was created in order to perform thermal simulations on the entire phone system. Using transient analysis with proper initial and boundary conditions, the battery reaction to heat parameters could be observed over the discharge cycle.

7.2.1. Solid Modeling

The solid model created was based off the phone to be used for experiments: the HTC One S. This phone has the battery and main PCB on the same general plane. This means that the heat flow from the PCB components would be directed along the long axis of the battery. Heat would enter at one end of the battery, and dissipation would be directed perpendicular to the heat flow.

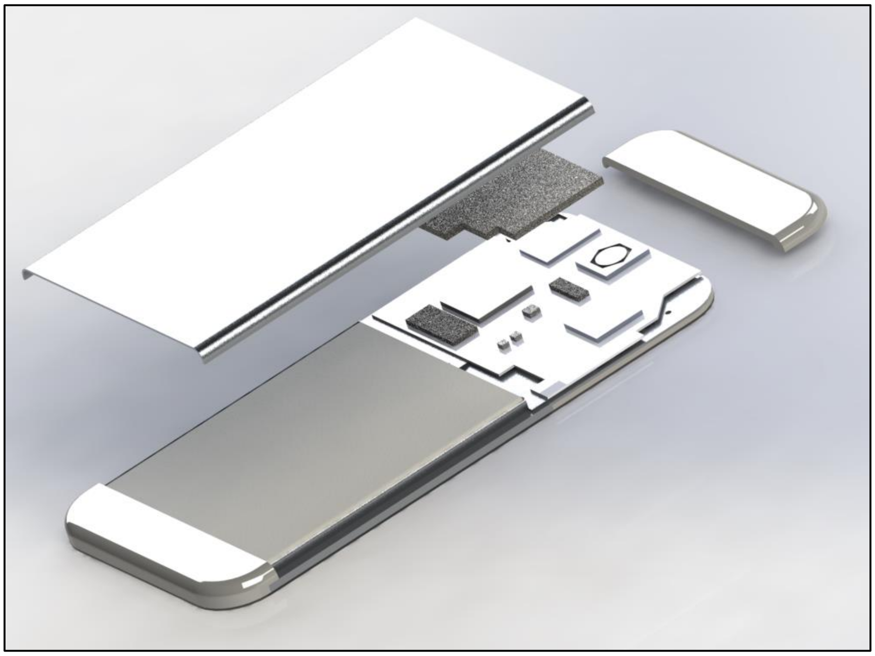


Figure 18 - Exploded phone showing interior components.

Shown in Figure 18 is the entire model with the rear cover and component housing removed to show the heat generation components on the PCB. The model was created at actual size, with accurate dimensions and materials for all important components. Since the test phone had to remain functional for experimental work, some interior components are likely not present as the component housing was not removed on the experimental phone. In addition, many small components including individual transistors, capacitors, and small microprocessors were eliminated. These components generate irrelevant amounts of heat and would only cause thermal simulations to require extended time periods to be solved by the software.

In the thermal simulation package, boundary and initial conditions were to be set. The initial temperature was chosen as room temperature, as all experimental results were to begin at room temperature as well. All heat generation components were given appropriate heat loads shown in red in Figure 19.

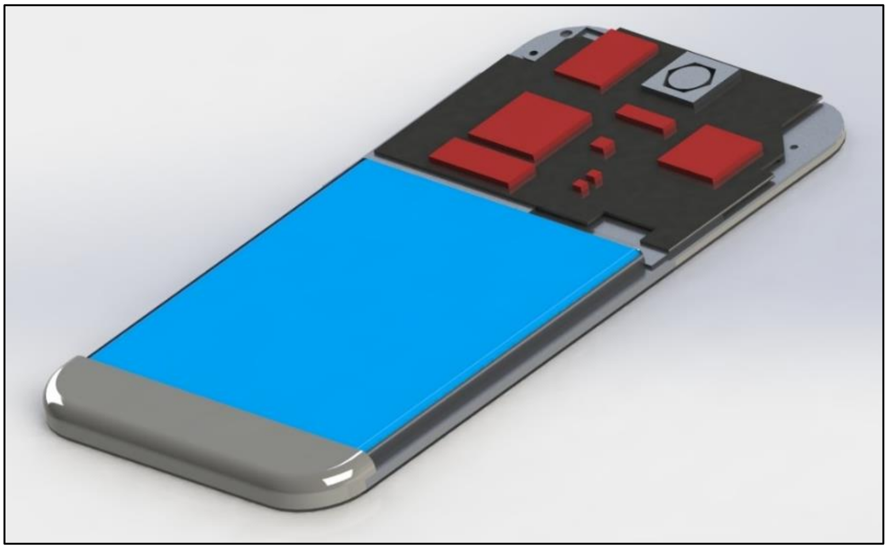


Figure 19 - SolidWorks model showing heat generation components (red) and battery (blue).

For the purposes of simplification of model parameters, on the components contributing the highest percentages of heat were given loads: the central processing unit (CPU), graphics processing unit (GPU), GSM antennae, audio module, battery, and screen. Contributions by the storage, memory, camera module, and WiFi were all removed. During experimentation, in order to eliminate variables, these components would not be in use or would be disabled. Similar to the experimental device, the heat generation components were all placed in plane with the battery. All heat loads on components were considered maximum and static. During experimental trials, benchmarks would enable similar conditions.

Boundary conditions were set in accordance with the parameters developed in 1D and 2D modeling. Conduction parameters were handled by the simulation software, and conduction is governed by the materials present and their dimensions. Convection and radiation parameters were user inputted. This involved conditions inside the phone as well as on the exterior surface.

Heat extraction for the thermal management system was put in place on the rear surface of the battery, directed opposite the screen. This heat extraction could be precisely controlled in order to mimic extraction by various thermal systems.

7.2.2. Heat Extraction Simulations

Simulations were run for various periods and conditions. Since the electrical interactions between the battery and phone components were not present, all simulations were conducted until battery temperature had adequately stabilized with the phone components under load.

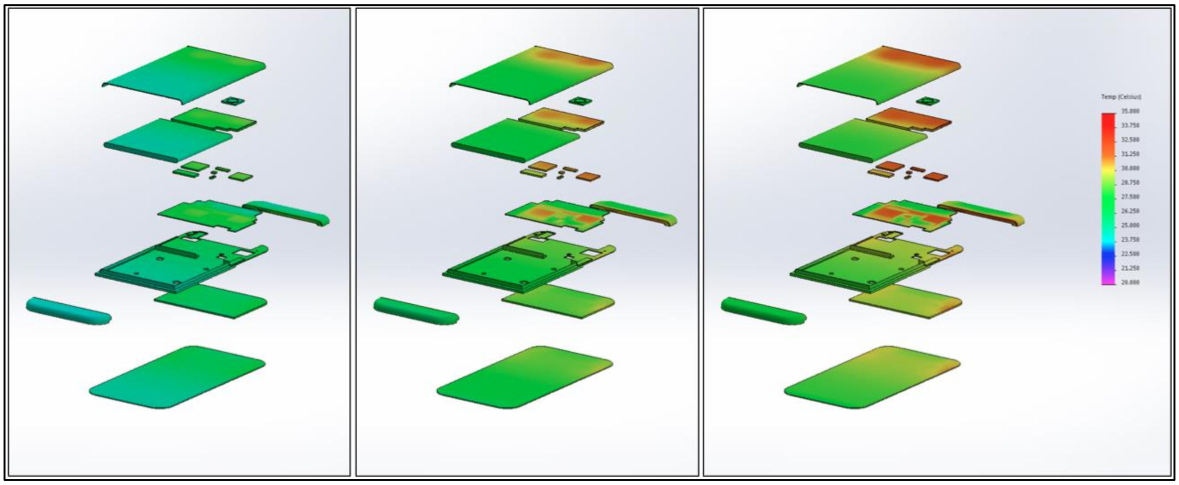


Figure 20 - Time lapse of simulation

Simulations performed admirably and as expected. During the early stages of the discharge time period, the heat generation components heated up fastest and to the highest temperatures. Heat flow spread outward from the components towards the battery, screen, and the various housings and spacers. This verified idea placement of a thermal cooling solution: adjacent to the CPU-housing-battery interface.

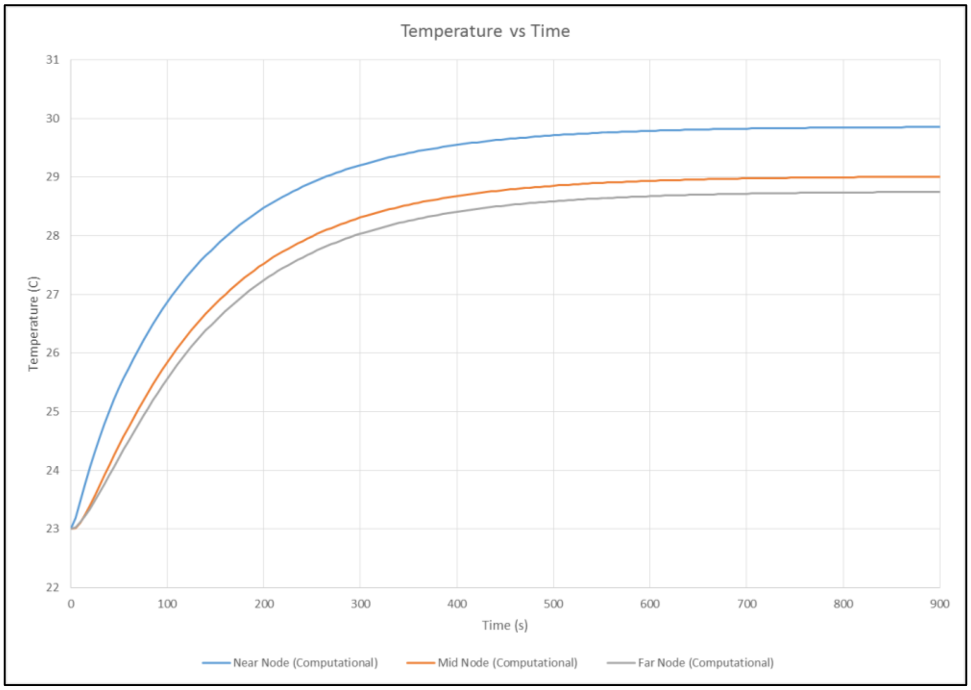


Figure 21 - Temperature vs. time data for computational results at multiple battery nodes

Results of thermal simulations gave insight into how the phone system would react under load. Various temperature vs. time curves could be generated. Depending how the real phone system would perform under load, simulation results would be tailored in order to keep results parallel.

7.3. Experimental Methods: Proof of Concept with Peltier Control System

Before the experimental analysis portion of the methodology was completed, the accuracy of the internal thermistor application on the phone had to be compared to the actual surface temperature readings on the battery. An LM35 thermistor was applied on the battery of the phone nearest the CPU and was monitored by the TED350. The internal temperature was compared to the temperature read by the thermistor and the error was noted. The goal was to show the temperature scale between surface and internal. This showed why it was chosen to use the internal battery NTC thermistor as a means of monitoring despite not being able to be certain of error.

The external thermistor was not as accurate due to positioning on the battery, contact with the surface of the battery, and ambient air in the room where the experiments were performed. It was expected that at lower temperatures, they would match relatively closely, but as the phone battery heated up on the phone side, the external thermistor was lagging behind in readings. As the phone heated up well above ambient temperature, the dissipation and temperature difference between ambient and the battery caused significant error in the LM35 thermistor. A total of 5 experiments were performed, monitoring the temperature and the rate of temperature change for the battery. These experiments used the ThorLabs TED 350 temperature management with the Peltier control surface attached during preliminary experimentation. Experiments were as follows:

1. Maximum CPU load with no Peltier control.
2. Half CPU load with no Peltier control.
3. Maximum CPU load with Peltier control.
4. Average load experiment with Peltier control.
5. Setting maximum load temperature, then Peltier application to note change in temperature over time.

The computational analysis indicated where to place the Peltier optimally during testing, as well as the size and power necessary to cool the battery.

In the first test, a maximum load stress test on both cores was performed without Peltier control, and the time constant from a cold boot of the battery was found. Next, an ordinary load of 50% on both cores was applied, still without Peltier control, and the time constant was recorded. In addition, the time required for the battery to heat up to an equilibrium temperature was recorded, and compared in both of the first two experiments.

The next series of tests were with Peltier control, once again with maximum load and an ordinary load. Peltier control was used to ensure that the battery maintained a temperature of 30°C throughout the duration of the experiment. During experiments, the time to discharge was observed, providing new time constants each time. The data from these experiments was compared to the data observed from the preliminary experiments performed, and indicated an increase in the lifespan of the battery and time required to reach a certain temperature, which for our purposes was the aforementioned 30°C, proving the hypothesis that application of a thermal management system to the battery will increase the lifespan of the battery per discharge cycle to be true.

Finally, a high load was applied and after the temperature settled above the optimal 30°C, the Peltier control was activated and the time to settle at 30°C was observed. In addition, all values recorded during these experiments were compared to the values from the dimensional analysis performed in order to ensure that the values found during the experiments matched the theoretical work done.

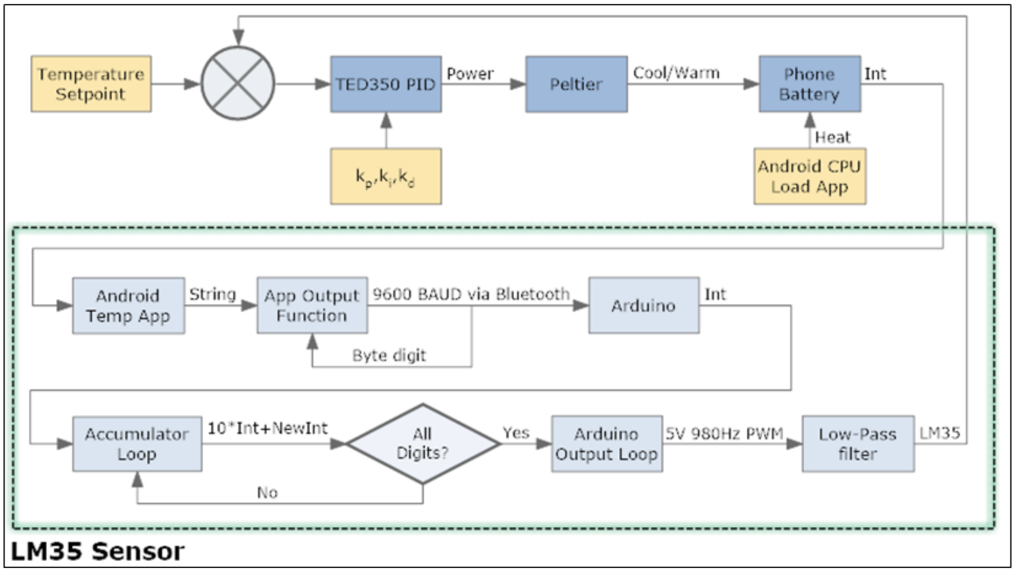


Figure 22 - Experimental control system showing the interactions between the components and the emulated LM35 sensor, outlined in the blue dotted line.

In order to gather data and give weight to the theoretical predictions, an experimental PID system was designed for proof of concept that a thermal management system would improve battery life. An active system was used, purely to prove the possibility of a thermal management system providing relevant battery life cycle. With developed thermal characteristics, a passive solution could be developed.

The TED 350 controller was again utilized in an active PID loop in order to regulate the temperature of the battery to approximately 30°C. Multiple CPU load levels were set to vary the temperature of the battery. The realization of this system gave data crucial to determine the heat capacity of the battery and showed the temperature relation to lifecycle, demonstrating the effect a thermal management system would have on battery life. Also shown is the impressive self-managing capability of smartphones by communicating between multiple systems. Again, note that this is an active system representing an ideal thermal management system acting on a lithium ion battery. The actual implementation of this active system with Peltier consumes significant power, but passive solutions exist as covered in the background of this project. This system shows the potential for battery life elongation with one of these passive solutions implemented properly, under heavy phone usage.

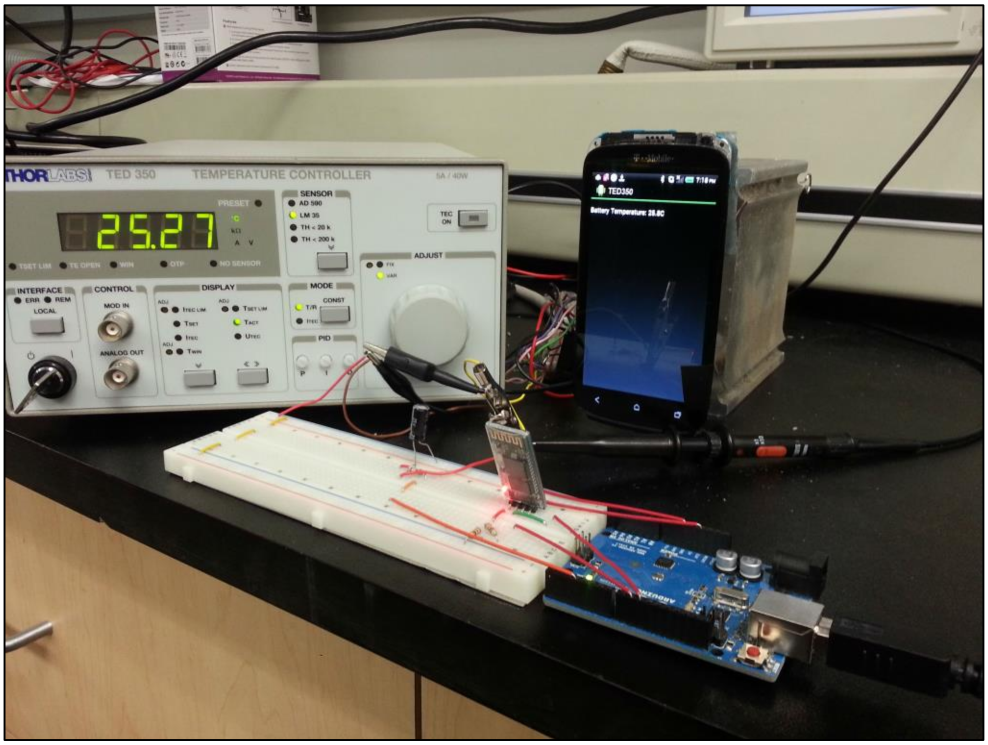


Figure 23 - Experimental setup showing ThorLabs TED350 Temperature Controller (left), Arduino board and BlueTooth module (center foreground), and phone running Android application (right).

As the battery temperature was found to be dependent almost entirely by the heat generation of components acting upon it, the battery temperature during experimentation was controlled by varying the CPU load. This was done by running a background busyloop on one or both cores of the Snapdragon S4 MSM8260A chip in the HTC One S. In order to vary the CPU load percentage and the temperature of the battery, the duty cycle was varied from .5 to 1, for corresponding 50% to 100% load.

7.3.1. Component Rationale

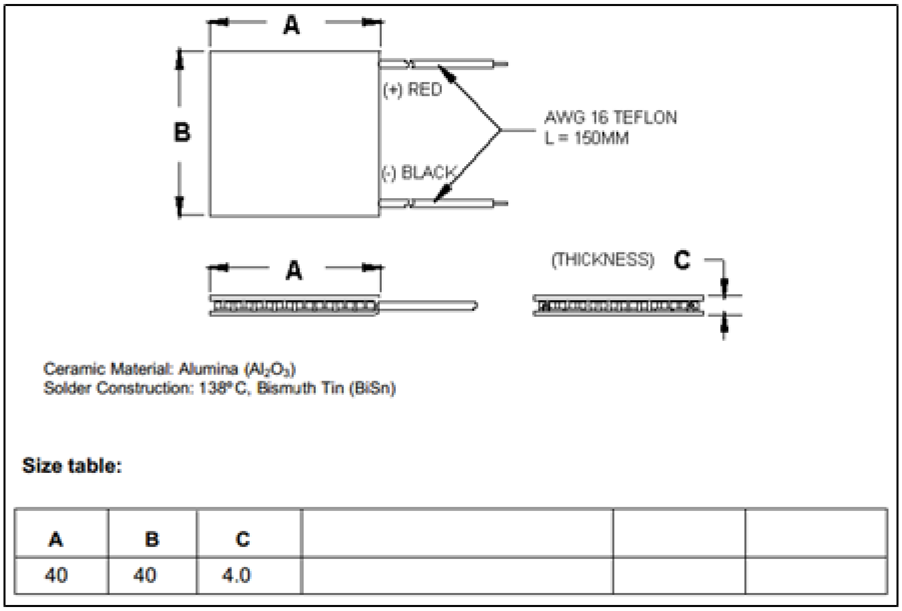
Several choices were made in the development of this system that may not be immediately obvious or require explanation. For one, a Peltier surface was chosen as the control element because it is a cheap and mature technology with significant documentation, that the TED 350 supports it without modification. The Peltier element also fit on the backside of the battery which was necessary for the experiment. While Peltier surfaces are inefficient, this model was developed as a proof of concept that a cooling system would work to increase battery life of a cell phone even in aftermarket case design. Furthermore, efficiency of thermoelectric materials is currently increasing at a high rate, and may be a feasible solution in several years, especially utilized in passive mode to generate power and recharge the battery during use.

Secondly, the LM35 thermistor that was previously used, was not utilized, but emulated with code and the voltage output of an Arduino microcontroller. The necessity of this design became apparent after preliminary tests revealed inaccuracy with the sensor physically mounted on the backside of the battery. Relatively uninsulated, the sensor's reading of the battery surface was skewed significantly by the ambient air temperature. In addition, the heat generated by the components took a long time to affect the opposite surface temperature of the battery, so the sensor lagged behind the actual temperature of the battery. Using the internal thermistor of the lithium ion battery, these issues were solved. As the TED350 is only capable of taking input from one of four specified temperature sensors, emulating the LM35 was chosen for its linear nature that would be practical for simulation.

Selection and Purchase

With the system design logic decided upon, the next step was to research components for actual implementation. These components consisted of the Peltier surface for temperature control and the programs used for the coding for the Android and Arduino communication system. While each component was available in many different forms, it was essential that these components would meet our desired function effectively and fall under necessary constraints.

First researched was a Peltier that would be able to operate at the desired level of heat dissipation under the TED350 limitations for the amount of current that could be supplied. Power and dimensions were two of the parameters that were investigated. As explained in the constraints section under physical and thermal constraints, the Peltier must not only handle the maximum power (12W) and current that could be applied to it (5A), but also fit for the dimensions of the battery to allow for sufficient cooling over the area. Some more advanced Peltier surfaces offer higher efficiency and quality of thermoelectric material, but can cost upwards of $40 to 50 USD. As efficiency was not the goal of experimentation, with the Peltier surface power externally, a less effective Peltier surface would serve the same purpose. The Hebei TEC1-12705 purchased was both sufficient in terms of cooling, and cost-effective.

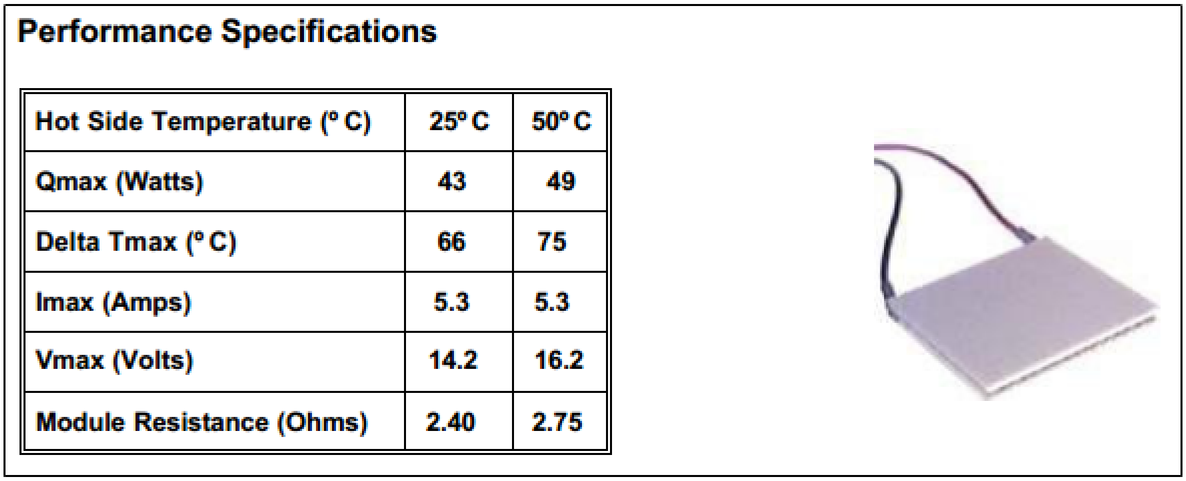


Ceramic Material: Alumina (Al_2O_3)
Solder Construction: 138° C, Bismuth Tin (BiSn)

Size table:

A	B	C			
40	40	4.0			

Figure 24 - Peltier data sheet showing important dimensions.



Performance Specifications

Hot Side Temperature (° C)	25° C	50° C
Qmax (Watts)	43	49
Delta Tmax (° C)	66	75
Imax (Amps)	5.3	5.3
Vmax (Volts)	14.2	16.2
Module Resistance (Ohms)	2.40	2.75

Figure 25 - Peltier data sheet showing important thermal characteristics.

Secondly, a thermistor or sensor was needed to verify the accuracy of the temperature readings of the emulated thermistor. There are many sensors that could have sufficed for these experiments, but ultimately, the LM35 was selected. There are a few reasons behind this selection. Firstly, the TED 350 only accepts the input of four types of sensors: the AD590 and 592 series, and the LM35 and 135 series. As a result, our selection was narrowed by brand and series. The LM35 was rated for a temperature accuracy of +/- 0.6 degrees C, which sufficed for the experiments, as it was known that error would arise and it was desirable for the readings of the surface temperature to be as accurate as possible. Comparison to thermistors of the AD 590 and 592 series did not offer as close to a range as desired. Additionally, the flat surface of the thermistor allowed for easier application to the surface of the battery as opposed to thermistors with round dome tops. Also, the LM 35 was non-insulated so the temperature readings would be more accurate. Finally, there was cost. Thermistors in general are relatively inexpensive, though some can range upwards of $15 USD or so. The LM 35 thermistor was the least expensive after reviewing the two prior reasons. Thus, it was chosen for the experiments.

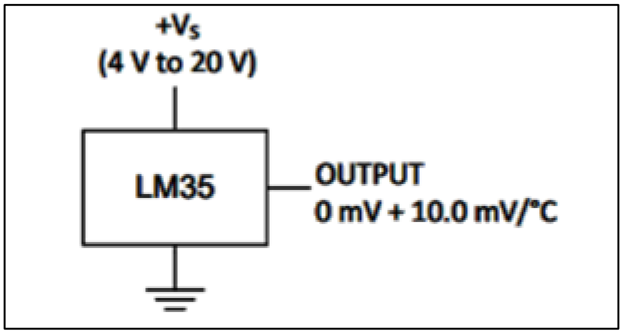


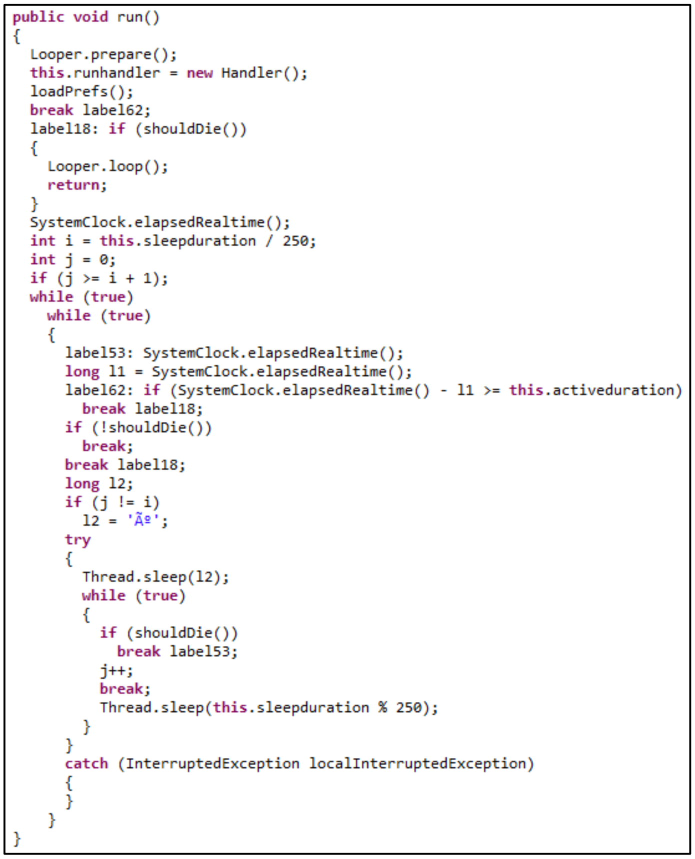
Figure 26 – LM35 temperature sensor and connections.

7.3.2. Android Program

An application was developed on the Android phone platform with the goal of applying a variable load on the CPU, and transmitting the battery temperature information to the Arduino microcontroller for LM35 voltage emulation. This was done in multiple steps of coding and the implementation of PID controls via the TED350 temperature controller.

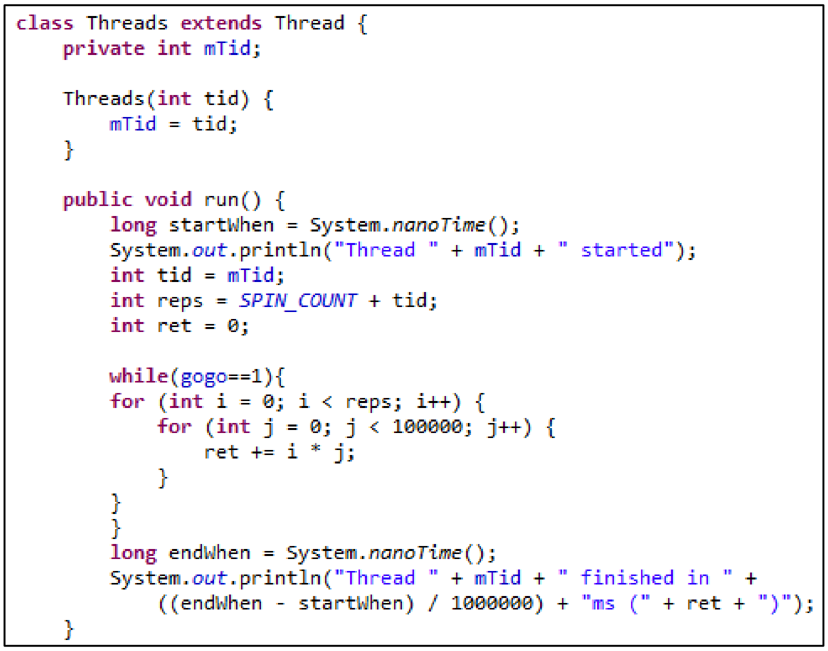
7.3.3. Load

This part of the application was developed due to a perceived lack of options in stress and benchmarking applications currently available on the market. It creates an artificial load on the CPU by stressing threads individually via busyloops. This means that one can apply a 25%, 50%, 75%, or 100% load on a quad core CPU, or 50% and 100% for our purposes testing the dual core HTC One S. Realizing that battery temperature is mostly dependent on the power generation of components -- of which the CPU plays the largest role -- by analyzing battery lifetimes at different CPU loads the temperature relationship to overall lifetime is proven. Furthermore, with this data the temperature relationship of the battery can observed with respect to the CPU frequency. This part of the application is currently working but the excellent yet obscure application, CPUBurn. This application works much as this part of the application began development, by creating busyloops, but they are fully active as background services and run at user-set duty cycles. This allowed greater customization and accuracy in CPU load setting for later experimental testing.



```
public void run()
{
  Looper.prepare();
  this.runhandler = new Handler();
  loadPrefs();
  break label62;
  label18: if (shouldDie())
  {
    Looper.loop();
    return;
  }
  SystemClock.elapsedRealtime();
  int i = this.sleepduration / 250;
  int j = 0;
  if (j >= i + 1);
  while (true)
    while (true)
    {
      label53: SystemClock.elapsedRealtime();
      long l1 = SystemClock.elapsedRealtime();
      label62: if (SystemClock.elapsedRealtime() - l1 >= this.activeduration)
        break label18;
      if (!shouldDie())
        break;
      break label18;
      long l2;
      if (j != i)
        l2 = 'Ãº';
      try
      {
        Thread.sleep(l2);
        while (true)
        {
          if (shouldDie())
            break label53;
          j++;
          break;
          Thread.sleep(this.sleepduration % 250);
        }
      }
      catch (InterruptedException localInterruptedException)
      {
      }
    }
}
```

Figure 27 - CPUBurn application



```
class Threads extends Thread {
    private int mTid;

    Threads(int tid) {
        mTid = tid;
    }

    public void run() {
        long startWhen = System.nanoTime();
        System.out.println("Thread " + mTid + " started");
        int tid = mTid;
        int reps = SPIN_COUNT + tid;
        int ret = 0;

        while(gogo==1){
        for (int i = 0; i < reps; i++) {
            for (int j = 0; j < 100000; j++) {
                ret += i * j;
            }
        }
        }
        long endWhen = System.nanoTime();
        System.out.println("Thread " + mTid + " finished in " +
            ((endWhen - startWhen) / 1000000) + "ms (" + ret + ")");
    }
```

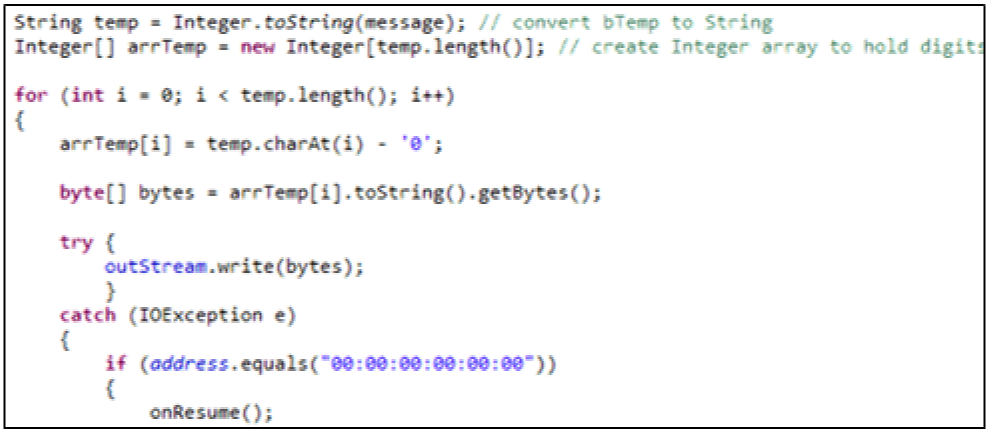
Figure *28* - Alpha version of Android application.

7.3.4. Bluetooth

The primary purpose of this application was for the specific need to control the TED350 – an analog PID temperature control system – via a smartphone's digital reading of the lithium ion battery's NTC thermistor. The application converts phone's current battery information into a serial data stream over Bluetooth, which is then processed and translated by an Arduino board into an output voltage that mimics an LM35. Overall, the reading from the phone's battery's internal thermistor is being processed and translated through multiple systems, in order to act as a specific thermistor input for the TED350, which only takes specific thermistor or temperature sensor inputs.

It creates an intent-filter for use within a broadcast-receiver which parses the phone for battery information at one-minute intervals, an Android limitation. The temperature is received as an Integer, which is the wrapper class for a 32bit int, allowing it to be acted on as an object which is necessary for future manipulation of this value. This Integer is then sent to the output function of the application, created with the goal of sending the battery temperature as serial data to the Arduino.

The output function receives the integer and converts it into a string, then creates an integer array of the same length of the string. Next, a for-loop is run that sets the values of the array to the individual digits of the string, minus char '0' to convert the data into an integer. Next, a byte is created by converting the current digit value of the integer array into a char, then into a byte. A serial output stream sends these digits in sequential order via Bluetooth, to be processed by the Arduino microcontroller.

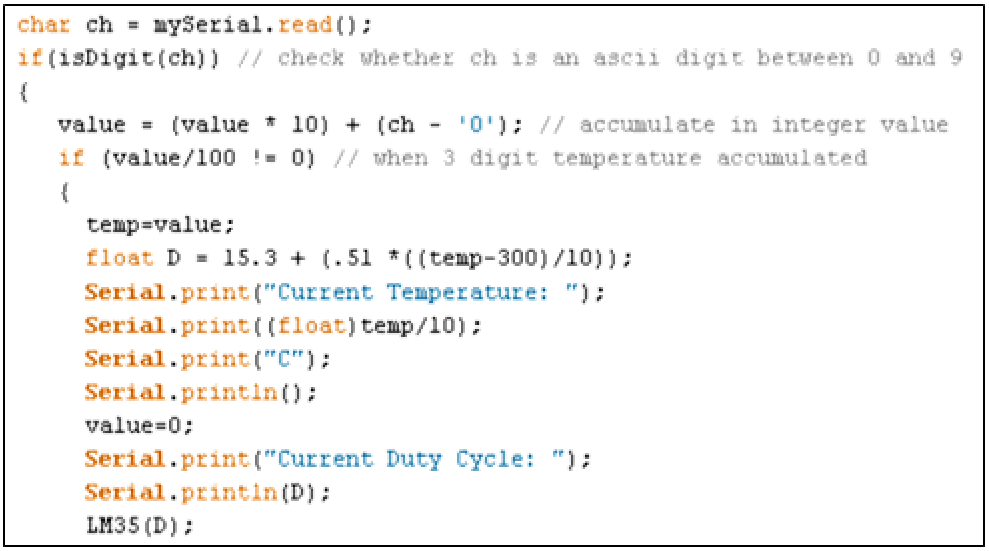


```
String temp = Integer.toString(message); // convert bTemp to String
Integer[] arrTemp = new Integer[temp.length()]; // create Integer array to hold digits

for (int i = 0; i < temp.length(); i++)
{
    arrTemp[i] = temp.charAt(i) - '0';

    byte[] bytes = arrTemp[i].toString().getBytes();

    try {
        outStream.write(bytes);
        }
    catch (IOException e)
    {
        if (address.equals("00:00:00:00:00:00"))
        {
            onResume();
```

Figure 29 – Conversion of integer to string operation

7.3.5. Arduino Program

The Arduino reads the serial data as a char at a baud rate of 9600 and is checked whether it is indeed an ASCII digit between 0 and 9. If so, it is accumulated by converting and adding it to an integer variable. When another digit is read, the current value of the integer is multiplied by ten, shifting the zero to the right, then the value of the new digit is added. When the value of this integer returns a value greater than 1 when divided by 100, it is theoretically three digits long for our purposes. Next, this value is converted into a value between 0 and 255 to output a PWM voltage equal to that an LM35 would at the given temperature. This is output as a PWM then put through a low pass filter to give an approximate DC output voltage. The value of the integer is reset, the loop is broken, and data is accumulated again. The ted350 reads this output voltage as a temperature as it would an LM35. Using PID, the TED350 controls the Peltier to keep the battery temperature constant.



```
char ch = mySerial.read();
if(isDigit(ch)) // check whether ch is an ascii digit between 0 and 9
{
   value = (value * 10) + (ch - '0'); // accumulate in integer value
   if (value/100 != 0) // when 3 digit temperature accumulated
   {
     temp=value;
     float D = 15.3 + (.51 *((temp-300)/10));
     Serial.print("Current Temperature: ");
     Serial.print((float)temp/10);
     Serial.print("C");
     Serial.println();
     value=0;
     Serial.print("Current Duty Cycle: ");
     Serial.println(D);
     LM35(D);
```

Figure 30 - Arduino program showing steps for verification of accurate temperature data.

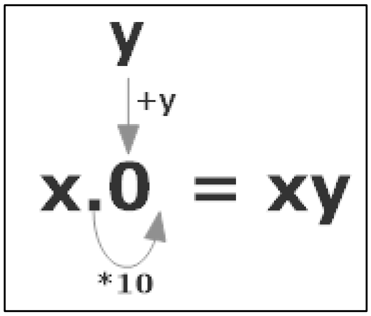


Figure 31 - Algorithm contained in Android application for interpreting of integer inputs and translating to output temperatures.

7.3.6. ThorLabs TED350 Parameters and Control

The Proportional, Integral, and Differential controller is used widely as a means to quickly and accurately bring a fluctuating system to a given stable set point. In order to maintain a PID loop, the proportional, integral, and differential parameters must be set, and the set point reference temperature must be defined. The PID parameters were set using the heuristic Ziegler-Nichols method, and the set point was chosen to be 29C.

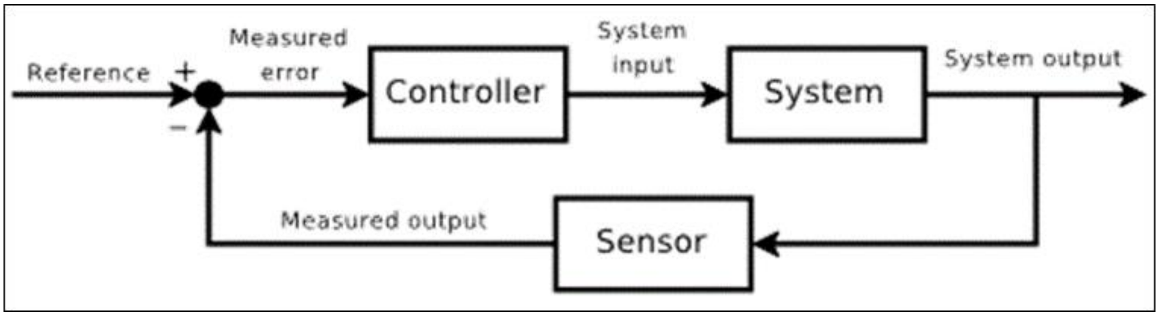


Figure 32 - PID system control loop.

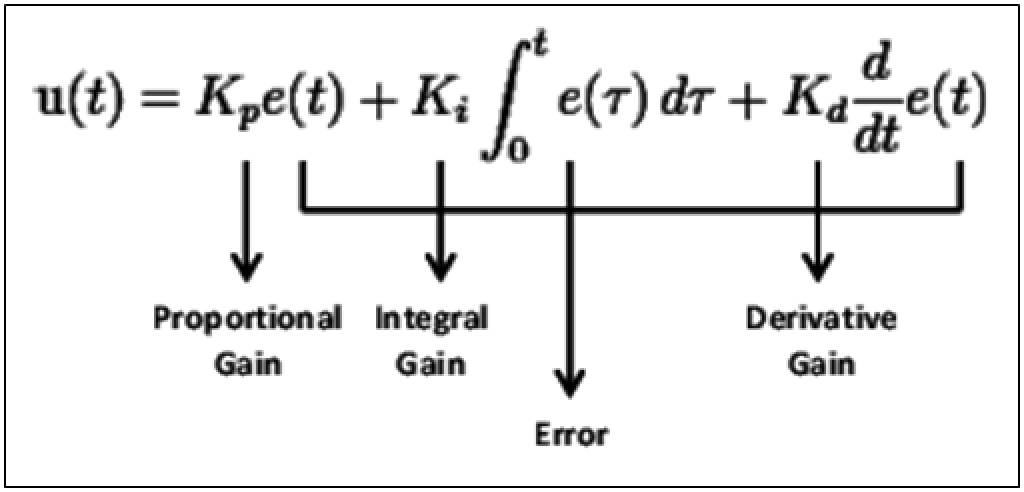

$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{d}{dt} e(t)$
Proportional Gain
Integral Gain
Error
Derivative Gain

Figure 33 - PID system governing equation.

7.3.7. PID

PID works by comparing the actual state of a system as read by a sensor, to the desired set point. The control is then applied based upon the proportional, integral, and derivative gain, which represent the present, past, and future error, respectively. Figure 34 below shows the impact of each parameter on the system at different gains.

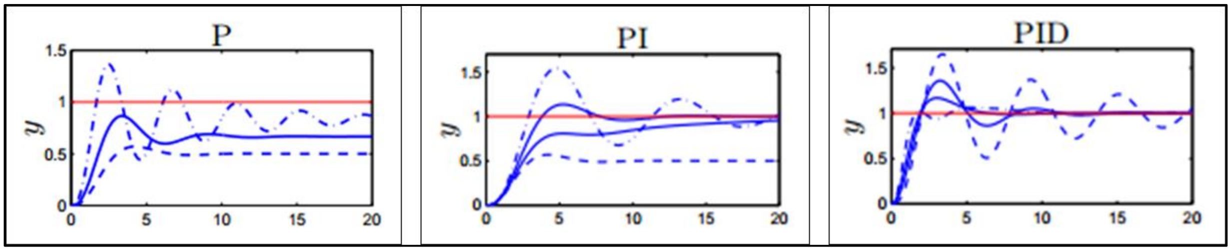


Figure 34 - Graphical illustration of PID control systems for increasing P gain (left), I gain (center), and D gain (right).

By increasing the proportional gain, rise time increases, steady state error is reduced as the system settles closer to the set point value, but the system becomes more oscillatory. The purpose of the proportional gain in most PID systems is to increase the rise time to an acceptable speed, as steady state error is typically handled by the integral portion of the controller. By increasing the integral gain, the function accumulates error over time, building the system drive voltage to reach a very high gain, eliminating steady state error and increasing oscillations. The general purpose of the integral gain is to eliminate steady state error. Increasing the derivative gain causes a counteraction of the current error but amplifies unwanted noise and disturbances in the signal. The derivative of the system at its most positive value is at that point at its most negative slope. As this function swings negative as the error is positive, the derivative function yields a damped system with faster stabilization time, but can be tricky to tune in systems prone to disturbances.

7.3.8. Functional Analog Implementation

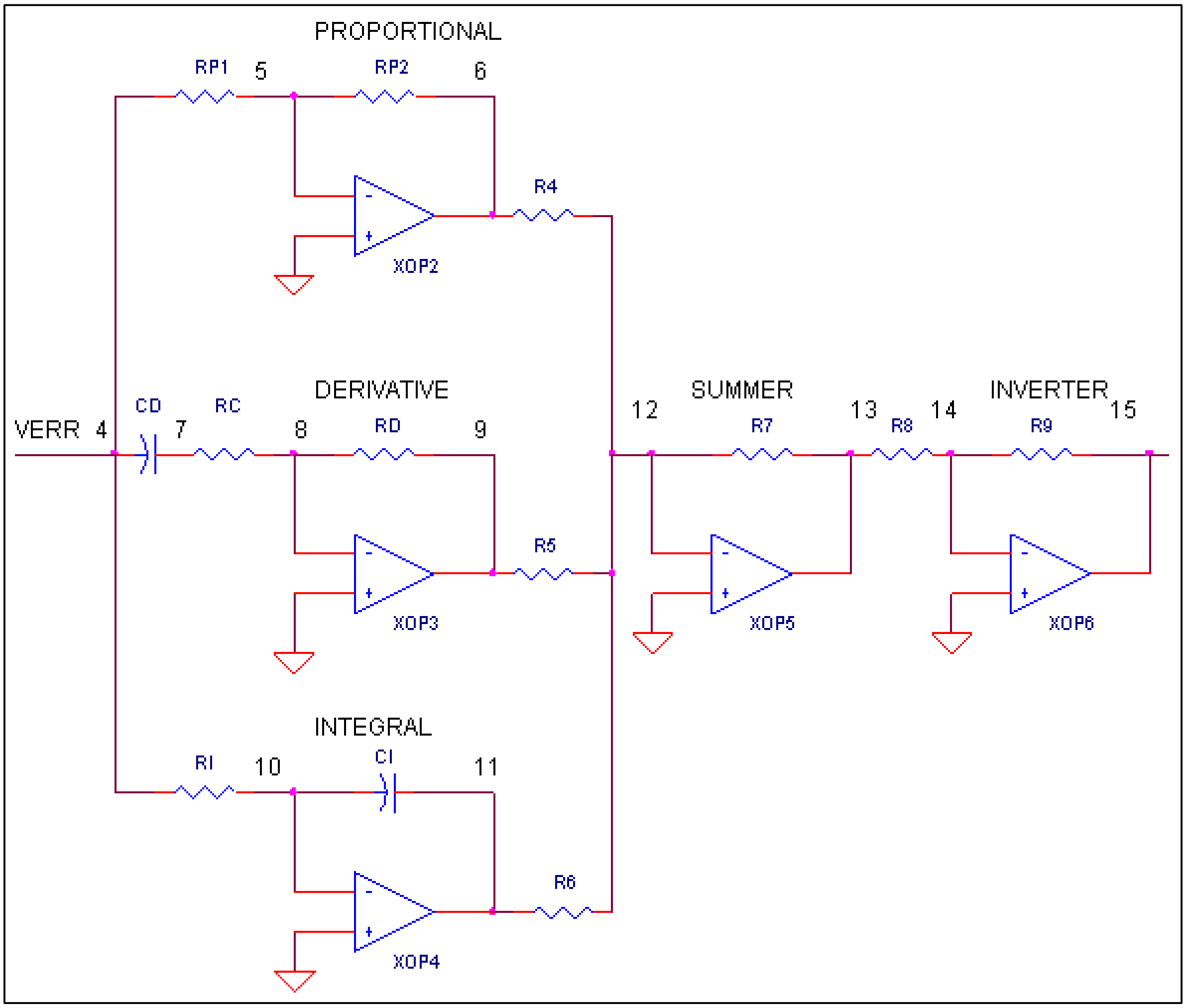


Figure 35 – Op-amp setup

An op-amp setup, as shown above in Figure 35 via a PSpice simulation, can be used to implement a variable PID controller. As shown, the proportional gain is implemented with an inverting, negative feedback op-amp. The integral gain is shown in this circuit with the use of an op-amp in an integrator configuration. Similarly, the derivative gain in this circuit is handled by an op-amp in derivative configuration. The resistor RC is used to reduce phase shift caused by RD and CD at high frequencies, improving system stability. Below are equations for the relative magnitude of each of these functions.

$$k_p = Verr\left(\frac{RP_2}{RP_1}\right)$$

$$k_i = \left(\frac{1}{RI} \cdot CI\right) integral(Verr \cdot dt)$$

$$k_d = RD \cdot CD\left(\frac{dVerr}{dt}\right)$$

Equation 21 - Governing equations for circuit development

To alter the parameter gain magnitudes, resistor values can be changed -- or, more simply, potentiometers may be used to tune each component quickly and accurately. This application utilizes three separate op-amps in different configurations for each parameter, which is easy to tune and accurate, but a fixed-gain PID controller can be implemented with a single op-amp and careful tuning calculations.

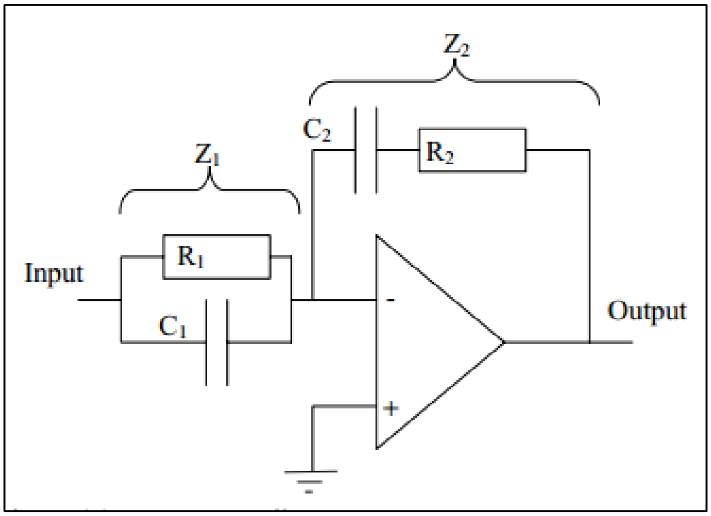


Figure 36 - Op amp circuit

Pictured above is a single op-amp circuit that handles PID control on its own. In order to determine the gain of the PID parameters for this circuit, the resistor and capacitor values must be calculated. As the equations for the proportional, integral, and derivative gains all affect one another, it is important to carefully determine the required components so that each value is of the correct magnitude. In addition, the mutually dependent nature of the gains in this circuit makes perpetual adjustment computationally tedious, and heuristic tuning a challenge. This circuit is thus typically fixed in gain and applied when the gain values necessary have already been found. Below are the equations necessary to calculate the PID control parameters for this implementation.

$$k_p = \frac{R_2}{R_1} + \frac{C_1}{C_2}$$

$$k_i = R_2 \cdot C_1$$

$$k_d = \frac{1}{R_1 \cdot C_2}$$

Equation 22 - Equations for calculating PID control parameters

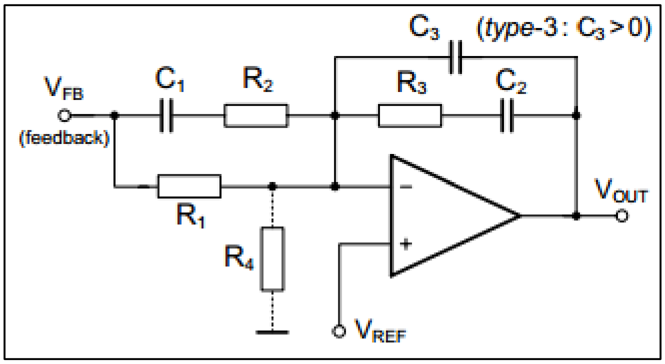


Figure 37 - Revised circuit

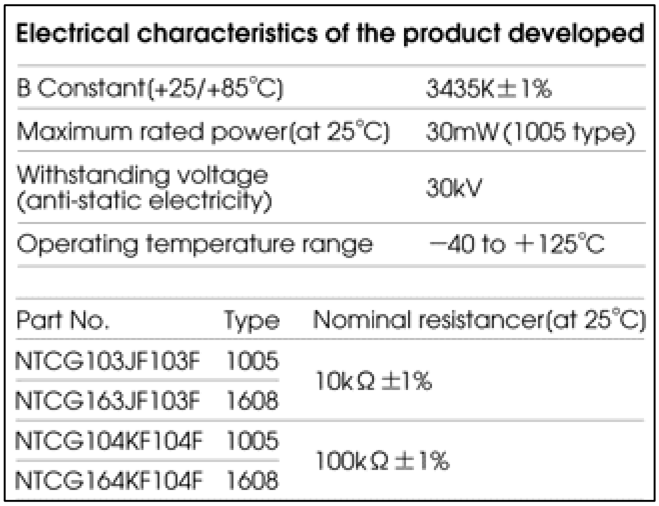
This circuit is different from the typical op-amp PID by the inclusion of resistor R4 and capacitor C3. The resistor R4 allows for additional gain control without affecting the closed-loop stability of the system. The capacitor C3 is added for derivative filtering and pole-stability of the system. By using C3 to adjust the roll-off frequency, loop noise induced by the derivative function is reduced. Below is the transfer function of this circuit.

$$\frac{V_{OUT}}{V_{FB}} = \frac{-1}{R_1(C_2 + C_3)} \frac{(R_3C_2s + 1)(C_1(R_2 + R_1)s + 1)}{s(R_2C_1s + 1)\left(\frac{C_2C_3}{C_2 + C_3}R_3s + 1\right)}$$

Equation 23 – Transfer function of circuit

7.3.9. Error Analysis

Error was analyzed from the coding perspective. The inherent error in the Li-Ion battery's thermistor is difficult to measure experimentally, but typical mobile NTC thermistor error at nominal temperature 25C is approximately 1%, which equates to +-.25C. In addition to this error, the Arduino is not completely accurate in its emulation of the LM35 so there is a loss of accuracy within the code as well. For the Arduino to output its PWM voltage, it takes an input integer between 0 and 255. However, with the required duty cycle of .06 for an average output voltage of .06*5V=.3V relative to the LM35 at 30C, the step size of the input integer is .51/C. This results in an error of +-1C in the Arduino output to the TED350. Overall system error is approximately +-1.25C. Because of this, and the fact that many manufacturers rate Li-Ion battery optimal temperature at 28-30 C, the temperature set point on the TED350 was set to an average 29 C.



Electrical characteristics of the product developed

B Constant(+25/+85°C)	3435K±1%
Maximum rated power(at 25°C)	30mW(1005 type)
Withstanding voltage (anti-static electricity)	30kV
Operating temperature range	−40 to +125°C

Part No.	Type	Nominal resistancer(at 25°C)
NTCG103JF103F	1005	10kΩ ±1%
NTCG163JF103F	1608	
NTCG104KF104F	1005	100kΩ ±1%
NTCG164KF104F	1608	

Figure 38 - Electrical characteristics

7.4. Model Verification

With the implemented thermal management system, the experimental results could be compared to the computational results. Since the computational results could run for an infinite amount of time, the area of interest was the rise profile. Both experiments and thermal simulations began at room temperature and were allowed to rise to optimal temperature. The temperature vs. time curves could be compared in order to verify model accuracy.

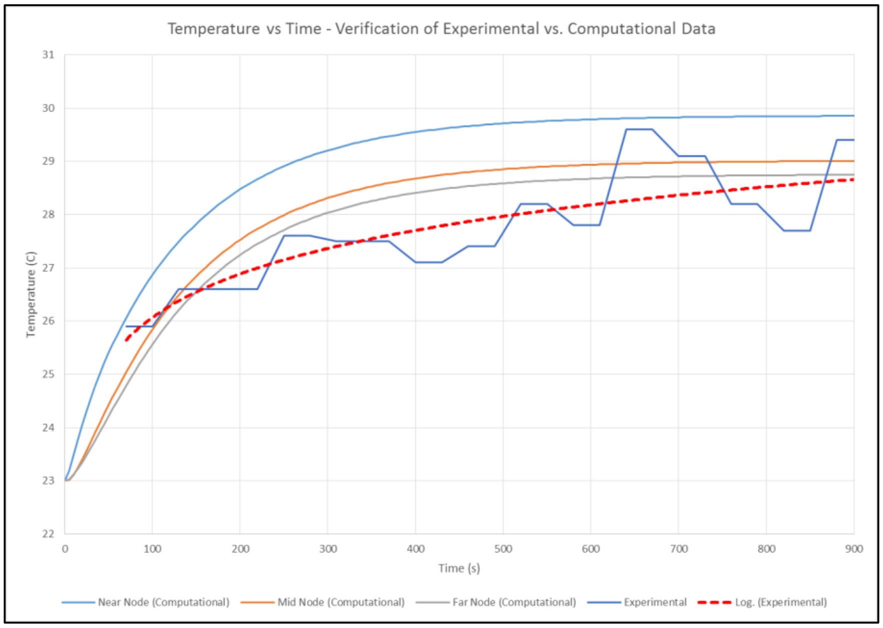


Figure 39 - Comparison of experimental and computational temperature vs. time curves.

Comparison of results showed an accurate modeling of the real world system. Shown in Figure 39 are the various temperature curves for the experimental and computational data. The blue, orange, and grey curves are outputs from the computational model. The blue jagged line is the output from the Android application during rise time. The red trend line was comparatively close to the computational results. This verified the accuracy of the computational model, and allowed for further evolution of the computational and experimental models in order to determine ideal thermal management system characteristics.

8. Results

Initial experimentation without a thermal management system indicated that the lithium ion battery system rose to a temperature between 40 and 45°C. At this temperature, the mobile phone had a discharge time of just over 2 hours. With incorporation of a thermal management system, the battery temperature could be controlled to 30°C. With a difference of 15°, the lifespan of the battery was shown to increase, illustrated in Figure 40.

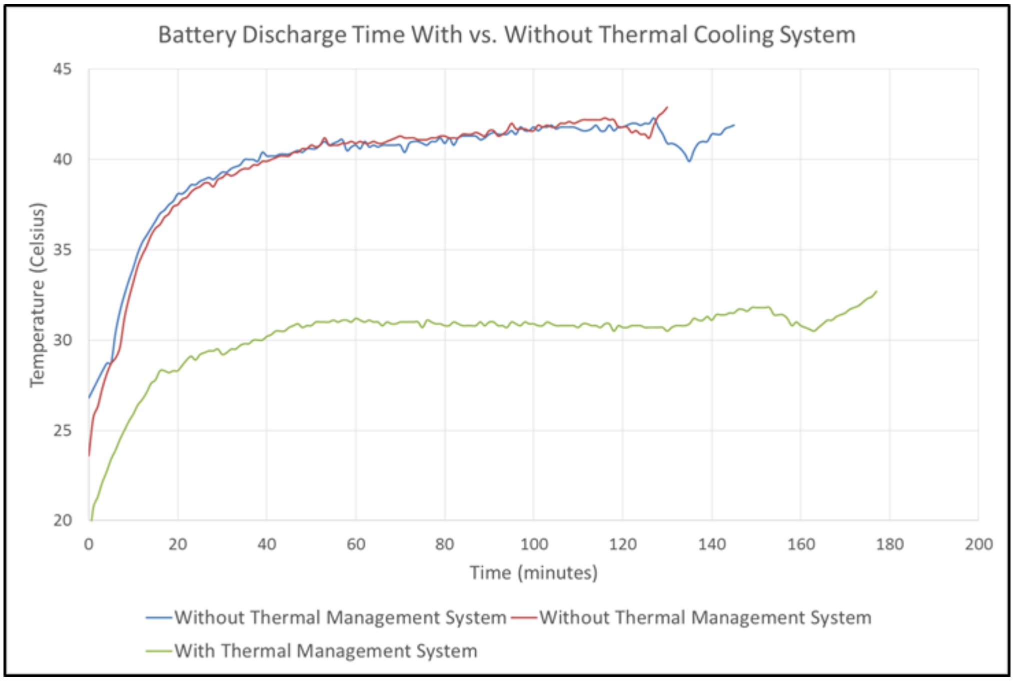


Figure 40 - Battery life cycle with thermal management solution (green) compared to that without thermal management solutions (blue, red).

The upper curves indicate the battery reached to the aforementioned temperature of 45°C and discharge lasted under 150 minutes. The lower curve displays experimentation with the installed proof of concept thermal management system. With the lithium ion battery controlled to the optimal 30°C, discharge lasted approximately 180 minutes, an increase of 23%. This is substantial evidence that the thermal management system extended the battery life when the battery was operating at the optimal temperature. Finally, the amount of power that needed to be dissipated to control the battery to the optimal temperature was investigated.

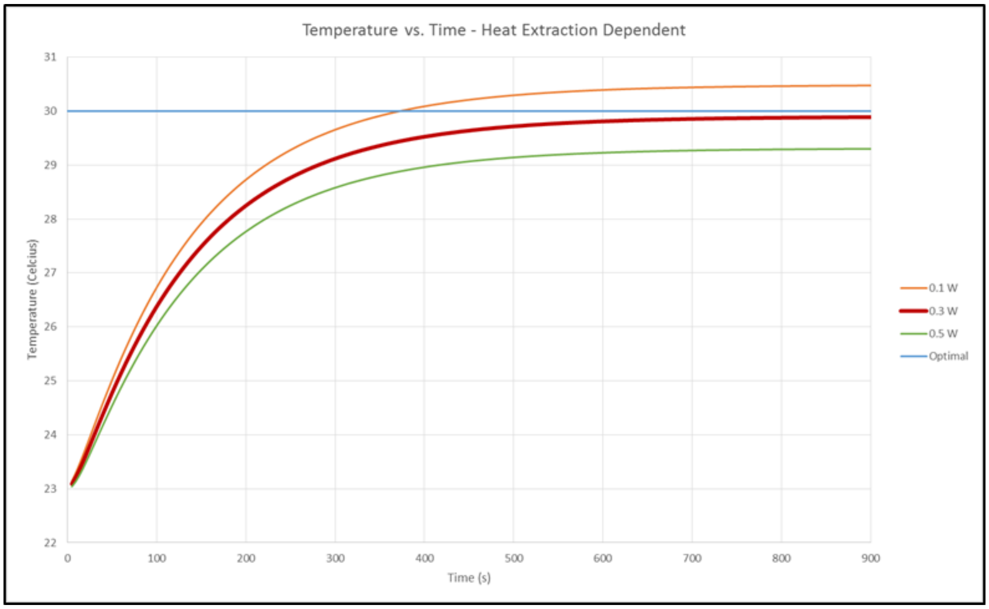


Figure 41 - Temperature vs. Time plot showing ideal heat capacity for stabilization at 30 degrees Celsius in red - 0.3W.

For the thermal management system to manage the battery temperature to 30°C, prolonging the battery life, a heat capacity of 0.3 Watts was required. For the experiments performed this equated to $0.001\frac{W}{m^2}$. While this value seems small, it should be noted that the dimensions of the battery are comparatively small when expressed in meters, thus this value is appropriate for the area of the rear surface of the lithium ion battery. Additionally, it should be noted that all data gathered does contain some error that needs to be taken into consideration. The temperature of 30° has a ± 1.5°C of error that was taken into consideration during experimentation. This means that temperatures recorded landed between 28 and 32 degrees C, which was observed by temperature readings from the Arduino app on the phone.

9. Conclusions and Recommendations

The results from the analytical, computational and experimental analysis provided us the thermal capabilities of the system for the lithium ion battery of the mobile phone. This data provided parameters for which future experimentation could be furthered on potential passive cooling systems: micro heat pipes, nanocomposite materials, and phase change materials. These complex systems are capable of the required heat capacity of 0.3 W, or $0.001\frac{W}{m^2}$, and are therefore capable of thermally managing lithium ion batteries in high density electronic packaging to the optimal temperature of 30°C. Physical constraints together with developed thermal requirements can be used to select the most effective passive solution based on merits of cost, physical space requirements, and availability.

Further work utilizing an active solution could be developed in the area of software development. A more powerful phone could be programmed to control heat generation to an even fine degree, turning off select components for more accurate results. Additional work in thermal modeling could be undertaken to investigate potential alternate locations of cooling solutions, as well as the effects of multiple smaller solution locations on key components.

Through proper thermal management, it was shown that battery lifespan can be significantly extended. A result of 23% increase was achieved, showing the potential gains from implementing a thermal management system. A thermal management system could potentially be used to extend battery life in real world applications as the demand for higher computational power continues to increase. These thermal management systems, as shown from the work completed in this project, can provide the consumer longer battery life usage, leaving more time for productivity in mobile devices in both professional and entertainment applications.

References

[1] R. D. Deshpande, "Understanding and Improving Lithium Ion Batteries Through Mathematical Modeling and Experiments," University of Kentucky, 2011.

[2] P. M. Gomadam, J. W. Weidner, R. A. Dougal and R. E. White, "Mathematical modeling of lithium-ion and nickel battery systems," *Journal of Power Sources,* vol. 110, pp. 267-284, 2002.

[3] L. Cai and R. E. White, "Mathematical modeling of a lithium ion battery with thermal effects in COMSOL Inc. Multiphysics (MP) software," *Journal of Power Scources,* vol. 196, pp. 5985-5989, 2011.

[4] J. R. Dahn, "Phase diagram of LixC6," *Physical Review B,* vol. 44, no. 17, pp. 9170-9177, 1991.

[5] G. E. Grechnev, R. Ahuja, B. Johansson and O. Eriksson, "Electronic structure, magnetic, and cohesive properties of LixMn2O4," *Physical Review B,* vol. 65, no. 17, p. 174408, 2002.

[6] Q. Wang, J. Sun and C. Chen, "Thermal stability of LiPF6/EC+DMC+EMC electrolyte for lithium ion batteries," *Rare Metals,* vol. 25, no. 6, pp. 94-99, 2006.

[7] D. Chen, "Microscopic Investigations of Degradation in Lithium-Ion Batteries," Karlsruher Institut fur Technologie, 2012.

[8] S. S. Singh, "Optimal micro heat pipe configuration on high performance heat spreaders," San Jose State University, San Jose, 2009.

[9] W. Park, K. Lafdi, C. Yu and K. Choi, "Influence of Nanomaterials in Polymer Composites on Thermal Conductivity," *Journal of Heat Transfer,* vol. 134, no. 4, 2012.

[10] T. Tong, Y. Zhao, L. Delzeit, A. Kashani, M. Meyyappan and A. Majumdar, "Dense Vertically Aligned Multiwalled Carbon Nanotube Arrays as Thermal Interface Materials," in *IEEE Transactions on Components and Packaging Technologies,* 2007.

[11] O. Zmeskal and L. Dohnalova, "Thermal Properties of Phase Change Materials," *Int J Thermophys,* 2013.

[12] M. J. Rabaey, A. Chandrakasan and B. Nikolic, Digital Integrated Circuits: A Design Perspective, Upper Saddle River, NJ: Pearson Education, 2003.

[13] "Qualcomm Snapdragon," Qualcomm, 2014. [Online]. Available: http://www.qualcomm.com/snapdragon/processors/800. [Accessed 15 April 2014].

[14] K. De Vogeleer, G. Memmi, P. Jouvelot and F. Coelho, "The Energy/Frequency Convexity Rule: Modeling and Experimental Validation on Mobile Devices," Cornell University, Ithaca, 2014.

[15] A. Carroll and G. Heiser, "An Analysis of Power Consumption in a Smartphone," in *USENIX Annual Technical Conference*, Berkeley, 2010.

[16] G. P. Perrucci, F. H. P. Fitzek and J. Widmer, "Survey on Energy Consumption Entities on the," in *Vehicular Technology Conference (VTC Spring)*, Yokohama, 2011.

[17] D. Bernardi, E. Pawlikowski and J. Newman, "A General Energy Balance for Battery Systems," *Journal of The Electrochemcail Society,* vol. 132, no. 1, pp. 5-12, 1985.

[18] "Arduino to Android - Turning an LED On and Off," 14 May 2012. [Online]. Available: http://digitalhacksblog.blogspot.com/2012/05/arduino-to-android-turning-led-on-and.html.

[19] M. Margolis, "Receiving Serial Data in Arduino," inkling, [Online]. Available: https://www.inkling.com/read/arduino-cookbook-michael-margolis-2nd/chapter-4/recipe-4-3.

[20] S. P. Bhattacharyya, A. Datta and L. H. Keel, "PID Control," in *Linear Control Theory,* CRC Press, 2010, pp. 201-222.

[21] "How to force maximum CPU usage," Stackoverflow, January 2014. [Online]. Available: http://stackoverflow.com/questions/19997064/how-to-force-maximum-cpu-usage.

[22] "TtsSetup_Java," GitHub, 2014. [Online]. Available: https://github.com/gregko/TtsSetup_Java.

[23] *Operation Manual Thorlabs Instrumentation Temperature Control TED350(-IEEE),* Germany: Thorlabs, 2007.

[24] "Thermoelectric Material the Best at Converting Heat Waste to Electricity," McCormick. Northwestern Engineering., 19 September 2012. [Online]. Available: https://www.mccormick.northwestern.edu/news/articles/2012/09/vinayak-dravid-thermoelectric-material-world-record.html.

[25] "Op Amp PID Controller," eCircuit Center, 2004. [Online]. Available: http://www.ecircuitcenter.com/Circuits/op_pid/op_pid.htm.

[26] V. Michael, C. Premont, G. Pillonnet and N. Abouchi, "Single Active Element PID Controllers," in *Radioelektronika 2010 20th International Conference*, Brno, 2010.

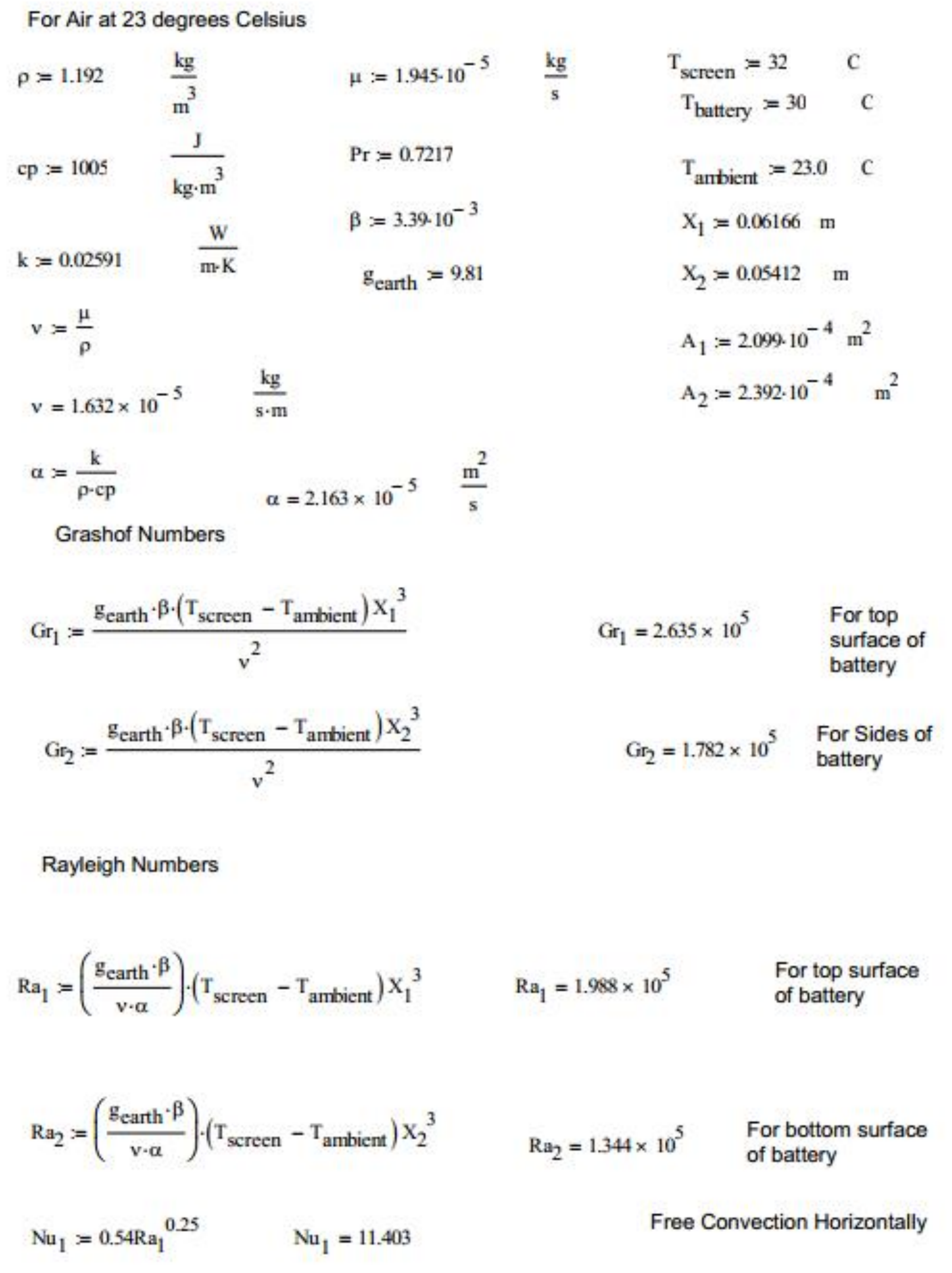
[27] "HFAN-08.2.0: How to Control and Compensate a Thermoelectric Cooler (TEC)," Maxim Integrated, 16 September 2004. [Online]. Available: http://www.maximintegrated.com/app-notes/index.mvp/id/3318.

[28] Y. Stauffer, "Analog PID module," 2004.

[29] "Multilayer Chip NTC Thermistors for Recharageable Battery Packs," TDK, [Online]. Available: http://www.tdk.co.jp/tfl_e/sensor_actuator/ntcg2/.

Appendices

Appendix I: Numerical Analysis of Lithium Ion Battery System



For Air at 23 degrees Celsius

$\rho := 1.192 \quad \frac{kg}{m^3}$ $\qquad \mu := 1.945 \cdot 10^{-5} \quad \frac{kg}{s}$ $\qquad T_{screen} := 32 \quad C$

$T_{battery} := 30 \quad C$

$cp := 1005 \quad \frac{J}{kg \cdot m^3}$ $\qquad Pr := 0.7217$ $\qquad T_{ambient} := 23.0 \quad C$

$\beta := 3.39 \cdot 10^{-3}$ $\qquad X_1 := 0.06166 \quad m$

$k := 0.02591 \quad \frac{W}{m \cdot K}$ $\qquad g_{earth} := 9.81$ $\qquad X_2 := 0.05412 \quad m$

$\nu := \frac{\mu}{\rho}$ $\qquad A_1 := 2.099 \cdot 10^{-4} \quad m^2$

$\nu = 1.632 \times 10^{-5} \quad \frac{kg}{s \cdot m}$ $\qquad A_2 := 2.392 \cdot 10^{-4} \quad m^2$

$\alpha := \frac{k}{\rho \cdot cp}$ $\qquad \alpha = 2.163 \times 10^{-5} \quad \frac{m^2}{s}$

Grashof Numbers

$$Gr_1 := \frac{g_{earth} \cdot \beta \cdot \left(T_{screen} - T_{ambient}\right) X_1^{\,3}}{\nu^2} \qquad Gr_1 = 2.635 \times 10^5$$

For top surface of battery

$$Gr_2 := \frac{g_{earth} \cdot \beta \cdot \left(T_{screen} - T_{ambient}\right) X_2^{\,3}}{\nu^2} \qquad Gr_2 = 1.782 \times 10^5$$

For Sides of battery

Rayleigh Numbers

$$Ra_1 := \left(\frac{g_{earth} \cdot \beta}{\nu \cdot \alpha}\right) \cdot \left(T_{screen} - T_{ambient}\right) X_1^{\,3} \qquad Ra_1 = 1.988 \times 10^5$$

For top surface of battery

$$Ra_2 := \left(\frac{g_{earth} \cdot \beta}{\nu \cdot \alpha}\right) \cdot \left(T_{screen} - T_{ambient}\right) X_2^{\,3} \qquad Ra_2 = 1.344 \times 10^5$$

For bottom surface of battery

$Nu_1 := 0.54 Ra_1^{\,0.25}$ $\qquad Nu_1 = 11.403$

Free Convection Horizontally

$$Nu_2 := 0.68 + \frac{\left(0.67 \cdot Ra_2^{.25}\right)}{\left[1 + \left(\frac{0.492}{Pr}\right)^{\left(\frac{9}{16}\right)}\right]^{\left(\frac{4}{9}\right)}} \qquad Nu_2 = 10.545$$

Free Convection Vertically

Convection Coefficients for Top Surface and Sides of Battery

$$h_1 := \frac{Nu_1 \cdot k}{X_1} \qquad h_1 = 4.791 \qquad \frac{W}{m^2 \cdot K}$$

$$h_2 := \frac{Nu_2 \cdot k}{X_2} \qquad h_2 = 5.048 \qquad \frac{W}{m^2 \cdot K}$$

Energy Balance

$$Energy_{exhausted\_to\_cooling\_solution} = Energy_{generated\_within\_battery} + Energy_{input}$$

Components of the Phone

$Q_{cpu} := 0.750$ $\quad Q_{gsm} := 0.057$ $\quad Q_{gps} := 0.143$ $\quad$ Watts (W)

$Q_{audio} := 0.028$ $\quad Q_{wifi} := 0.006$ $\quad Q_{sim} := 0$

$$Q_{comp} := Q_{cpu} + Q_{gsm} + Q_{gps} + Q_{audio} + Q_{wifi} + Q_{sim}$$

$Q_{comp} = 0.984$ Watts

Energy Produced by Screen $\quad Q_{screen} := 0.30$ W

Total Energy Generated

$Q_{total} := Q_{comp} + Q_{screen}$ $\quad Q_{total} = 1.284$ W

Convection

Top Surface

$T_{surface} := 30.0$ C

$Q_{top} := h_1 \cdot A_1 \cdot \left(T_{surface} - T_{ambient}\right)$

$Q_{top} = 7.04 \times 10^{-3}$ Watts

Sides

$Q_{side} := h_2 \cdot A_2 \cdot \left(T_{surface} - T_{ambient}\right)$

$Q_{side} = 8.453 \times 10^{-3}$ Watts

Total Convection

$Q_{totalcv} := Q_{top} + 2 \cdot Q_{side}$ $\quad Q_{totalcv} = 0.024$ Watts

$Q_{correct} := h_2 \cdot A_1 \cdot \left(T_{surface} - T_{ambient}\right) = 7.418 \times 10^{-3}$

Uncertainty Analysis Using the Heat Dissipated (convection)

$Q = hA\left(T_s - T_e\right)$

$Q_{correct} := h_2 \cdot A_1 \cdot \left(T_{surface} - T_{ambient}\right) = 7.418 \times 10^{-3}$ Watts

$U_h := 0.5$

$U_A := \left[(0.055 \cdot 0.00005)^2 + (0.0043 \cdot 0.0005)^2\right]^{\left(\frac{1}{2}\right)} = 3.491 \times 10^{-6}$

$U_{ts} := 0.5$

$U_{te} := 0.05$

$$Q_{correcth} := \left[U_h \cdot A_1 \cdot \left(T_{surface} - T_{ambient}\right)\right]^2 = 5.397 \times 10^{-7}$$

$$Q_{correcta} := \left[h_2 \cdot U_A \cdot \left(T_{surface} - T_{ambient}\right)\right]^2 = 1.522 \times 10^{-8}$$

$$Q_{correctts} := \left[h_2 \cdot A_1 \cdot \left(U_{ts}\right)\right]^2 = 2.807 \times 10^{-7}$$

$$Q_{correctte} := \left[-h_2 \cdot A_1 \cdot \left(U_{te}\right)\right]^2 = 2.807 \times 10^{-9}$$

$$Q_{correcttot} := \left(Q_{correcth} + Q_{correcta} + Q_{correctts} + Q_{correctte}\right)^{\left(\frac{1}{2}\right)} = 9.157 \times 10^{-4}$$

Percent Uncertainty

$$\%Q := \frac{Q_{correcttot}}{Q_{correct}} \cdot 100 = 12.345$$

$$\%Qh := \frac{Q_{correcth}}{\left(Q_{correcttot}\right)^2} \cdot 100 = 64.37$$

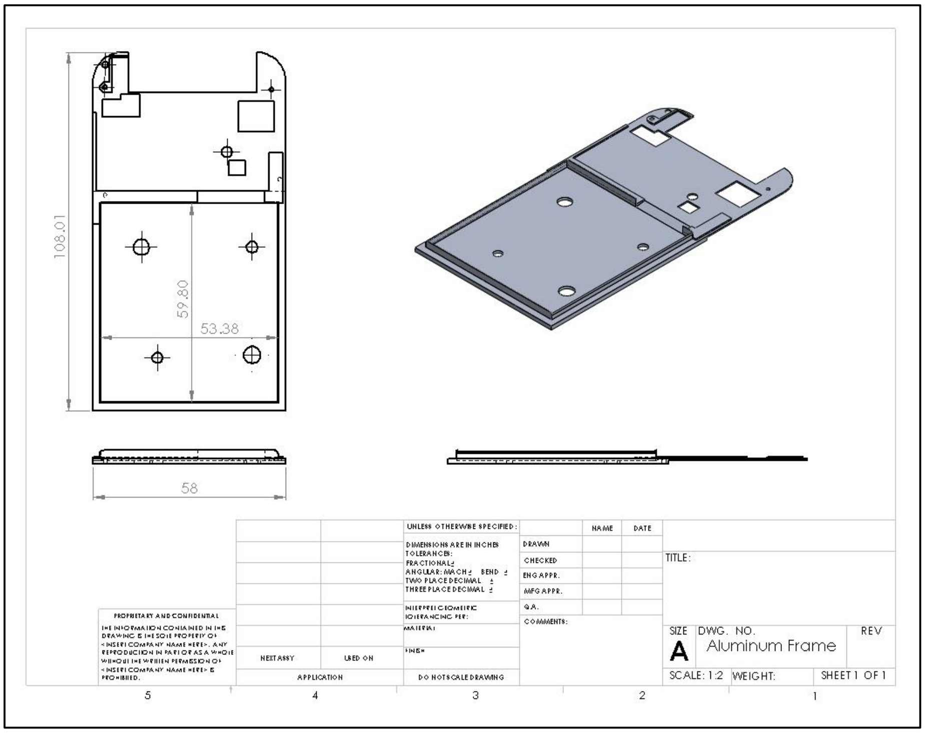
$$\%Qa := \frac{Q_{correcta}}{\left(Q_{correcttot}\right)^2} \cdot 100 = 1.815$$

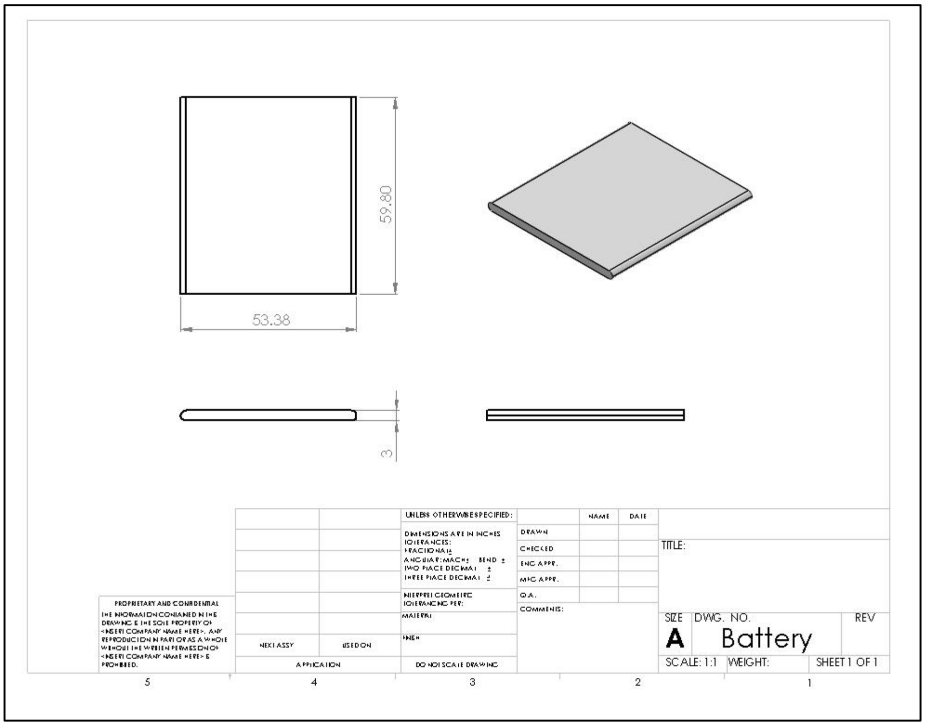
$$\%Qts := \frac{Q_{correctts}}{\left(Q_{correcttot}\right)^2} \cdot 100 = 33.48$$

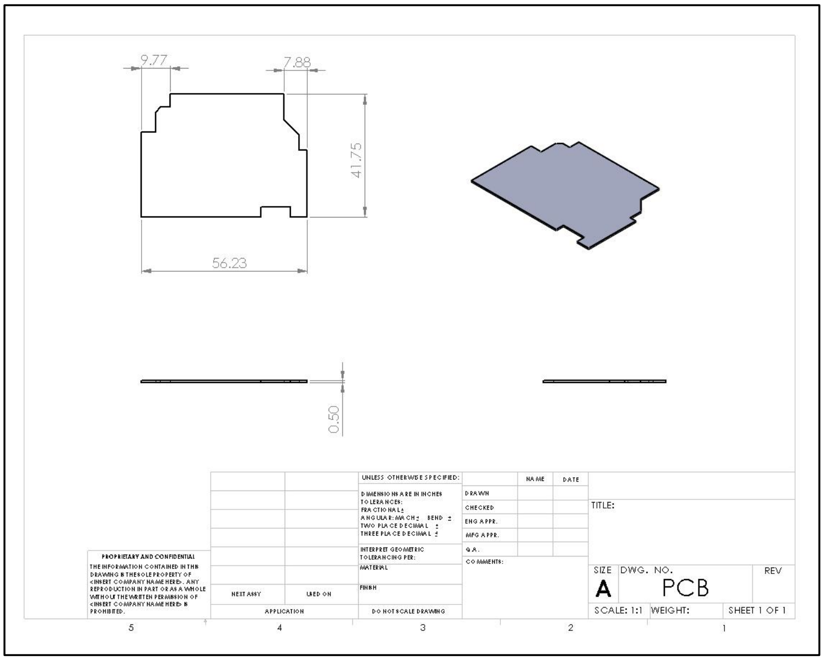
$$\%Qte := \frac{Q_{correctte}}{\left(Q_{correcttot}\right)^2} \cdot 100 = 0.335$$

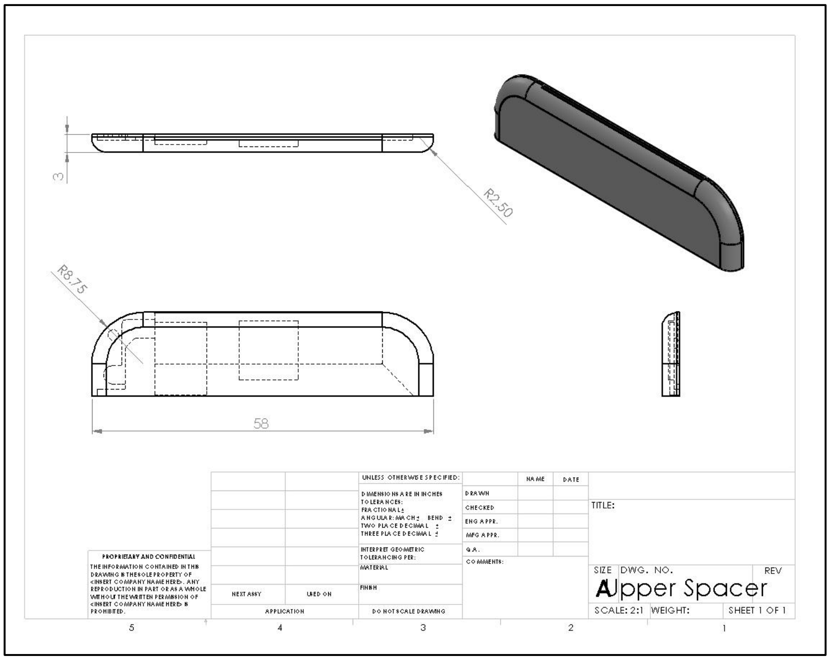
$$\%Qh + \%Qa + \%Qts + \%Qte = 100$$

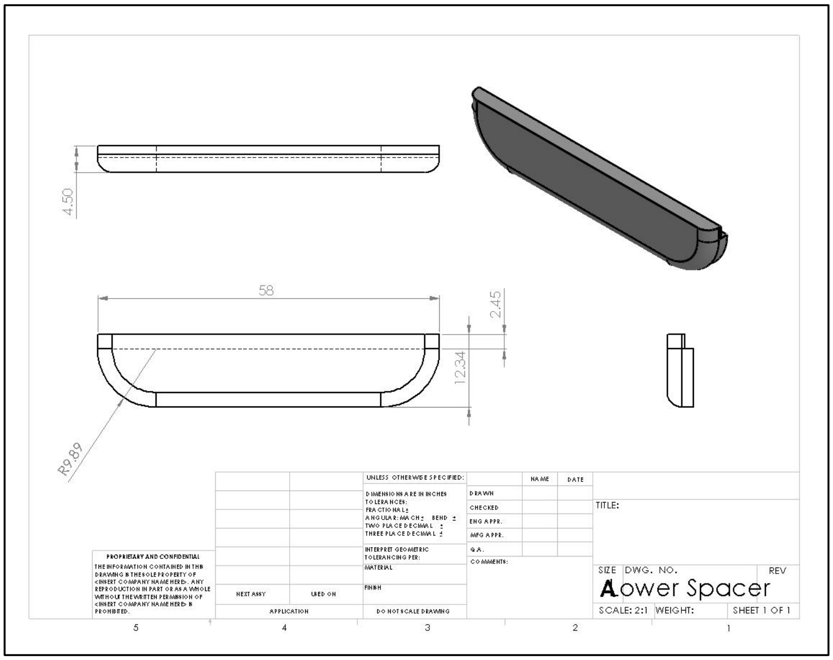
Appendix II: SolidWorks Model and Schematics

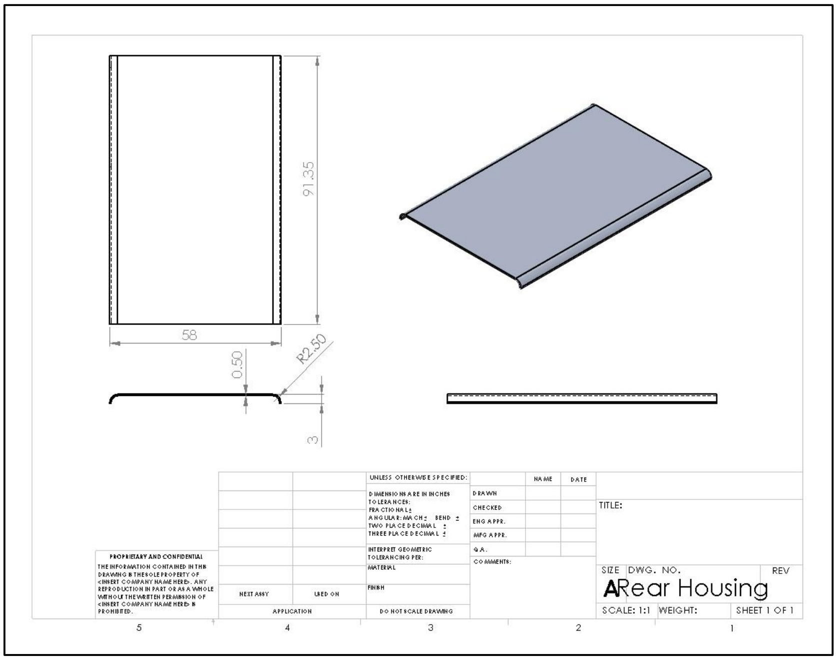



59.80
53.38
3
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL±
ANGULAR: MACH± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF <INSERT COMPANY NAME HERE>. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF <INSERT COMPANY NAME HERE> IS PROHIBITED.
NEXT ASSY
USED ON
APPLICATION
TITLE:
SIZE
A
DWG. NO.
Battery
REV
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1
5
4
3
2
1


9.77
7.88
41.75
56.23
0.50
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL±
ANGULAR: MACH± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC
TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
SIZE
A
DWG. NO.
PCB
REV
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF <INSERT COMPANY NAME HERE>. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF <INSERT COMPANY NAME HERE> IS PROHIBITED.
NEXT ASSY
USED ON
APPLICATION
5
4
3
2
1


3
R2.50
R8.75
58
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL±
ANGULAR: MACH± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC
TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
SIZE
A
DWG. NO.
Upper Spacer
REV
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF <INSERT COMPANY NAME HERE>. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF <INSERT COMPANY NAME HERE> IS PROHIBITED.
NEXT ASSY
USED ON
APPLICATION
5
4
3
2
1


4.50
58
2.45
12.34
R9.89
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
SIZE
DWG. NO.
REV
A
Lower Spacer
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1
NEXT ASSY
USED ON
APPLICATION
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF <INSERT COMPANY NAME HERE>. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF <INSERT COMPANY NAME HERE> IS PROHIBITED.
5
4
3
2
1


91.35
58
0.50
R2.50
3
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
SIZE
DWG. NO.
REV
A
Rear Housing
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1
NEXT ASSY
USED ON
APPLICATION
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF <INSERT COMPANY NAME HERE>. ANY REPRODUCTION IN PART OR AS A WHOLE WITHOUT THE WRITTEN PERMISSION OF <INSERT COMPANY NAME HERE> IS PROHIBITED.
5
4
3
2
1

Appendix III: Arduino Code

```
/*
 * SerialReceive sketch
 * Blink the LED at a rate proportional to the received digit value
*/
#include <SoftwareSerial.h>
#include <Time.h>

const int out = 3; // pin the TED350 is connected to
int value;
int temp;
int D;
SoftwareSerial mySerial(10,9);

void setup()
{
  Serial.begin(9600); // Initialize serial port to send and receive at 9600 baud
  mySerial.begin(9600);
  setTime(12,0,0,1,1,11); // set time base
  pinMode(out, OUTPUT); // set this pin as output
}

void loop()
{
  if(mySerial.available())
  {

    char ch = mySerial.read();
    if(isDigit(ch)) // check whether ch is an ascii digit between 0 and 9
    {
      value = (value * 10) + (ch - '0'); // accumulate in integer value
      if (value/100 != 0) // when 3 digit temperature accumulated
      {
        temp=value;
        float D = 15.3 + (.51 *((temp-300)/10));
        Serial.print("Current Temperature: ");
        Serial.print((float)temp/10);
        Serial.print("C");
        Serial.println();
        value=0;
        Serial.print("Current Duty Cycle: ");
```

```
         Serial.println(D);
         LM35(D);
         DisplayVal();
       }
    }
  }
  D=0;
}


// output voltage emulation of LM35
void LM35(float Duty)
{
  analogWrite(out,Duty);
  delay(1);
}

void DisplayVal(){
  // digital clock display of the time
  Serial.print(hour());
  printDigits(minute());
  printDigits(second());
  Serial.println();
}

void printDigits(int digits){
  // utility function for clock display: prints preceding colon and leading 0
  Serial.print(":");
  if(digits < 10)
    Serial.print('0');
  Serial.print(digits);
}
```

Appendix IV: Java Code

```
package com.example.TED350;

import java.io.IOException;
import java.io.OutputStream;
import java.util.UUID;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.util.Log;
import android.view.WindowManager;
import android.widget.TextView;
import android.widget.Toast;

import android.view.View;
import android.widget.Button;

public class TED350 extends Activity {

  private static final String TAG = "TED350";
  String sTemp;

  //for stress
  private static final int SPIN_COUNT = 2000;
  public int stop;
  long grandTotal = 0;
  long grandTotalWithFiles = 0;
  int numRuns = 0;
  //end stress

  private TextView Temp;
  public int onStartCommand;
  private static final int REQUEST_ENABLE_BT = 1;
  private BluetoothAdapter btAdapter = null;
  private BluetoothSocket btSocket = null;
  private OutputStream outStream = null;

  // Well known SPP UUID
  private static final UUID MY_UUID =
      UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

  // Bluetooth device MAC address
  private static String address = "20:13:12:05:11:06";

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    Log.d(TAG, "In onCreate()");
    setContentView(R.layout.main);
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);

    btAdapter = BluetoothAdapter.getDefaultAdapter();
    checkBTState();

            Temp = (TextView)findViewById(R.id.temp);

            this.registerReceiver(this.batteryReceiver, new
IntentFilter(Intent.ACTION_BATTERY_CHANGED));

            // CPU load
    Button testBtnMed = (Button) findViewById(R.id.testButtonMed);
    testBtnMed.setOnClickListener(onTestClickMed);

    Button testBtnMax = (Button) findViewById(R.id.testButtonMax);
    testBtnMax.setOnClickListener(onTestClickMax);


    Button stopBtn = (Button) findViewById(R.id.stopButton);
    stopBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            stop=1;
        }
    });
```

```
}

View.OnClickListener onTestClickMax = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int numThreads = 2;
        stop=0;
        System.out.println("Starting " + numThreads + " threads");
        long startWhen = System.nanoTime();

        SpinThread threads[] = new SpinThread[numThreads];
        for (int i = 0; i < numThreads; i++) {
            threads[i] = new SpinThread(i);
            threads[i].start();
        }

        for (int i = 0; i < numThreads; i++) {
            try {
                threads[i].join();
            } catch (InterruptedException ie) {
                System.err.println("join " + i + " failed: " + ie);
            }
        }

        long endWhen = System.nanoTime();
        System.out.println("All threads finished in " +
            ((endWhen - startWhen) / 1000000) + "ms");

    }
};


View.OnClickListener onTestClickMed = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int numThreads = 1;
        stop=0;
        System.out.println("Starting " + numThreads + " threads");
        long startWhen = System.nanoTime();

        SpinThread threads[] = new SpinThread[numThreads];
        threads[0] = new SpinThread(1);
        threads[0].start();
        try {
            threads[0].join();
        } catch (InterruptedException ie) {
            System.err.println("join " + 1 + " failed: " + ie);
        }

        long endWhen = System.nanoTime();
        System.out.println("All threads finished in " +
            ((endWhen - startWhen) / 1000000) + "ms");

    }
};

class SpinThread extends Thread {
    private int mTid;

    SpinThread(int tid) {
        mTid = tid;
    }

   public void run() {
        long startWhen = System.nanoTime();
        System.out.println("Thread " + mTid + " started");
        int tid = mTid;
        int reps = SPIN_COUNT + tid;
        int ret = 0;

        while(stop==0){
        for (int i = 0; i < reps; i++) {
            for (int j = 0; j < 100000; j++) {
                ret += i * j;
            }
            Temp = (TextView)findViewById(R.id.temp);

          }
        }
        long endWhen = System.nanoTime();
        System.out.println("Thread " + mTid + " finished in " +
            ((endWhen - startWhen) / 1000000) + "ms (" + ret + ")");
    }
}
```

```
        private BroadcastReceiver batteryReceiver = new BroadcastReceiver(){
                @Override
                public void onReceive(Context arg0, Intent arg1){
                        Integer bTemp = arg1.getIntExtra("temperature", 0);
                        Temp.setText("Battery Temperature: "+String.valueOf((float)bTemp/10)
+ "C");

                        sendData(bTemp);
                }
        };


  @Override
  public void onResume() {
    super.onResume();

    Log.d(TAG, "...In onResume - Attempting client connect...");

    address = "20:13:12:05:11:06";

    // Set up a pointer to the remote node using it's address.
    BluetoothDevice device = btAdapter.getRemoteDevice(address);
    // Two things are needed to make an android connection:
    //   A MAC address, above
    //   A UUID, SPP
    try {
      btSocket = device.createRfcommSocketToServiceRecord(MY_UUID);
    } catch (IOException e) {
      errorExit("Fatal Error", "In onResume() and socket create failed: " + e.getMessage() + ".");
    }

    // Stop discovery mode to free resources
    btAdapter.cancelDiscovery();

    // Establish the connection.  This will block until it connects.
    Log.d(TAG, "...Connecting to Remote...");
    try {
      btSocket.connect();
      Log.d(TAG, "...Connection established and data link opened...");
    } catch (IOException e) {
      try {
        btSocket.close();
      } catch (IOException e2) {
        errorExit("Fatal Error", "In onResume() and unable to close socket during connection
failure" + e2.getMessage() + ".");
      }
    }

    // Create a data stream so we can talk to server.
    Log.d(TAG, "...Creating Socket...");

    try {
      outStream = btSocket.getOutputStream();
    } catch (IOException e) {
      errorExit("Fatal Error", "In onResume() and output stream creation failed:" + e.getMessage()
+ ".");
    }
  }

  @Override
  public void onPause() {
    super.onPause();

    Log.d(TAG, "...In onPause()...");

    if (outStream != null) {
      try {
        outStream.flush();
      } catch (IOException e) {
        errorExit("Fatal Error", "In onPause() and failed to flush output stream: " + e.getMessage()
+ ".");
      }
    }

    try     {
      btSocket.close();
    } catch (IOException e2) {
      errorExit("Fatal Error", "In onPause() and failed to close socket." + e2.getMessage() + ".");
    }
  }

  private void checkBTState() {
    // Check for Bluetooth support and then check to make sure it is turned on
  // Emulator doesn't support Bluetooth and will return null
  if(btAdapter==null) {
    errorExit("Fatal Error", "Bluetooth Not supported. Aborting.");
  } else {
    if (btAdapter.isEnabled()) {
      Log.d(TAG, "...Bluetooth is enabled...");
    } else {

      //Prompt to turn on Bluetooth
      Intent enableBtIntent = new Intent(btAdapter.ACTION_REQUEST_ENABLE);
      startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
    }
  }
}

private void errorExit(String title, String message){
  Toast msg = Toast.makeText(getBaseContext(),
      title + " - " + message, Toast.LENGTH_SHORT);
  msg.show();
  finish();
}

private void sendData(Integer message) {

        // convert bTemp to String
  String temp = Integer.toString(message);
  // create Integer array to hold digits
  Integer[] arrTemp = new Integer[temp.length()];

  for (int i = 0; i < temp.length(); i++)
  {
      arrTemp[i] = temp.charAt(i) - '0';

      byte[] bytes = arrTemp[i].toString().getBytes();

      //output the current digit via serial
      try {
          outStream.write(bytes);
          }
      catch (IOException e)
      {
          // attempt to reconnect if an address error occurs
      if (address.equals("00:00:00:00:00:00"))
      {
        onResume();
      }
    }
  }
 }
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:id="@+id/temp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <Button
        android:id="@+id/testButtonMed"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="50% Load" />

    <Button
        android:id="@+id/testButtonMax"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="100% Load" />

    <Button
        android:id="@+id/stopButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Stop" />

</LinearLayout>
```