Section	
Bench No.	

ECE110 Introduction to Electronics

Pre-Lab 6: Modeling Your Battery

Name: _____

Student ID: _____

Instructor: _____

Date: _____

Teammate/Student ID: _____

This part is reserved for your instructor

Score	
Instructor Signature	
Date	

Pre-Lab 6: Modeling Your Battery

Learning Objectives

- Learning basic knowledge of Matlab
- Review Thevenin Theory
- Find Thevenin-equivalent models through curve-fitting real data; scientific computing using MATLAB or Python.
- Extract information from datasheets.
- Reconcile conservation-of-energy equations
- Apply Thevenin-equivalent-circuit theory to the lab.
- Prepare for lab by finding the IV characteristic equation of the battery in series with a current-limiting potentiometer.

Thevenin Equivalent Circuit

You should be familiar with solving circuits containing sources and resistors using KVL and KCL. It is beneficial to know that there are other very powerful tools to analyze circuits. One of these tools is the Thevenin Equivalent Circuit. The theory of Thevenin Equivalent Circuit starts with a black box with 2 terminals. Black box is a term used for a device with unknown construction. It refers neither to the color or enclosure! Assume this black box only contains ideal sources and resistors. However, you have no information on how many sources and resistor are present, nor do you know the values of these components. Thevenin's theory, can transform a circuit, as viewed through two terminals, into a Thevenin Equivalent Circuit. A Thevenin Equivalent circuit is an ideal voltage source in serials with a resistor. By applying the Thevenin's theory, you can actually find a simple equivalent circuit for the circuit composed by the black box. An equivalent circuit is a circuit that functions identically to the original circuit. Any circuit that performs mathematically-equivalently to another circuit is considered an equivalent to that circuit.

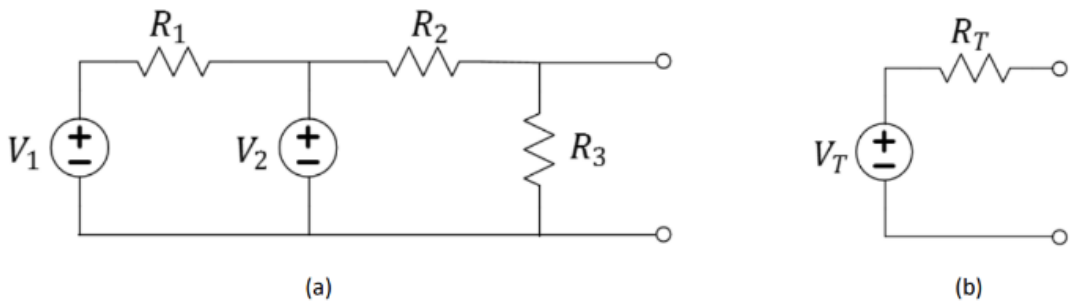


Figure 1: A complex DC circuit (a) and the Thevenin Equivalent Circuit (b) to which all DC circuits can be reduced.

In order to find V_T, open the two terminals and measure the open-circuit voltage (connect a voltage meter to the two terminals). In order to find R_T, short the two terminals and measure the short circuit current I_{SC}, then apply Ohms Law. $R_T = V_T / I_{SC}$.

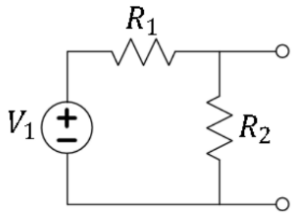
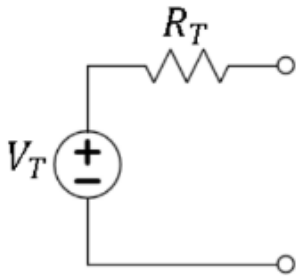


Figure 2: A practice circuit.

Question 1: For the practice circuit in Figure 2, find the Thevenin Equivalent Circuit when $V_1 = 5V$, $R_1 = 2.2$ kΩ, and $R_2 = 3.3$ kΩ.



$V_T =$ $R_T =$

Figure 3: The Thevenin equivalent of the practice circuit.

When given a circuit containing voltage sources, current sources, and resistors, it is possible to determine the Thevenin equivalent circuit using circuit analysis methods. Most real devices are not so simple, so we use empirical methods to determine the Thevenin equivalent circuit. In particular, we'll use graphical analysis of a device's IV curve.

Consider the IV curve of a resistor. The "curve" roughly follows the equation of a line. The slope of that line is equal to the inverse of the resistance as expressed by Ohm's Law. Many devices that are not resistors (combinations of ideal resistors, voltage sources and current sources) will also produce linear IV characteristics. The slope of the IV line is the inverse of the Thevenin resistance. In addition, the open-circuit voltage of these kinds of circuits will provide the value of the voltage of the Thevenin-equivalent voltage source (open-circuit forces the current to be equal to zero). In many cases, the IV curve is not perfectly linear (devices do not follow our simple models) but we can still approximate the Thevenin circuit by conducting a linear curve-fit over fixed ranges of data.

Non-ideal Voltage Source

Earlier in this course, we found the DC power supply to behave nearly as an ideal voltage source. For most practical applications, however, the use of a power supply is prohibitive as the DC power supply is too large and requires a wall outlet making it less than-mobile. Batteries provide a versatile option, but we must understand their non-ideal behaviors. Batteries present an engineering tradeoff. While they are less-than-ideal as a constant-voltage source, they can be made small and portable. What do we engineers do with this tradeoff? We figure out how to make the best of it by modeling the non-ideal behavior of the battery and accounting for it in our designs.

Let's begin by considering a very simple linear model of a battery, the Thevenin equivalent circuit. In this model, we assume that the battery is equivalent to an ideal voltage source (V_B) and source resistance (R_B) in series. In this prelab, we'll explore this model further by using DMM measurements (premeasured by your instructor) to determine appropriate values of V_B and R_B.

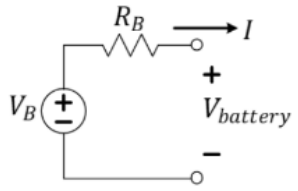


Figure 4: A first-order battery (Thevenin-equivalent) model.

Below, we are providing you with a table of real-life measurements from the 2-pack of 3.6-volt, Lithium batteries used in the laboratory. It will be your task to plot the data and to estimate the internal resistance (according to a Thevenin model) in the region where the battery is not severely loaded (forced to source a lot of current).

Current (mA)	Voltage (V)	Comments
0.0006	8.837	Using a potentiometer to control current via Ohm's Law. We did not record the resistance setting, but only the current and voltage. Voltmeter measures the voltage; Ammeter measures the current.
87.99	8.772	
93.60	8.759	
131.5	8.707	
148.9	8.688	
188.8	8.688	
215.2	8.627	
215.2	8.627	
258.4	8.586	
347.7	8.514	As the voltage dropped below 8 volts during the experiment, the current went to 800 mA, but then decreased wildly as the battery was forced well outside of its "linear" mathematical behavior!
534.9	8.375	
800	7.99	

Table 1 : IV data collected for the battery by the instructors. This is from one specific battery and not necessarily average for what might be expected.

Question 2: Plot the IV curve of a battery using the real-life data of Table 1.
*Note: Following the instruction below (**Starting MATLAB** to learn how to plot in MATLAB)*

Question 3: Use MATLAB and fit a linear curve to the data of Table 1(it is acceptable to use a pencil for the curve fit but doing it in software is preferred). Attach your graph to this assignment (**plot Q2 and Q3 in one figure**).
Note: Following the instruction in the section: *Introduction to Linear Fit using MATLAB*

Question 4: Determine the Thevenin equivalent circuit of the battery using your linear curve-fit and draw the equivalent circuit below. Label it as Figure 5. Explain the process of finding the Thevenin values.

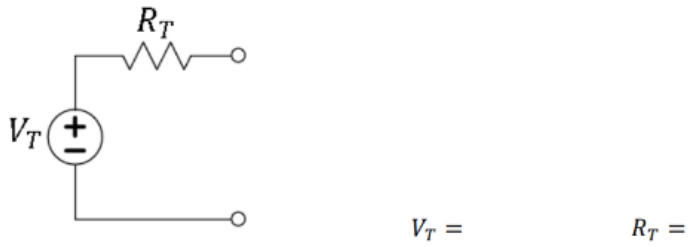


Figure 5: The Thevening equivalent circuit of the battery

Starting MATLAB

MATLAB is a very popular program used in a wide range of scientific computing and engineering applications. For our purposes we will use it to collect, manipulate, and plot the data we've measured in lab. You can think of it as the best graphing calculator

you've ever owned! It is accessible from all the lab computers (like the one installed on our lab benches). You can also access it on your personal computer via the Zhejiang University Software Platform. Visit http://ms.zju.edu.cn and choose *Download Matlab*, you will find the full instructions. You can save your MATLAB scripts, labeled as .m files, on your bench computers, or on your own personal computers.

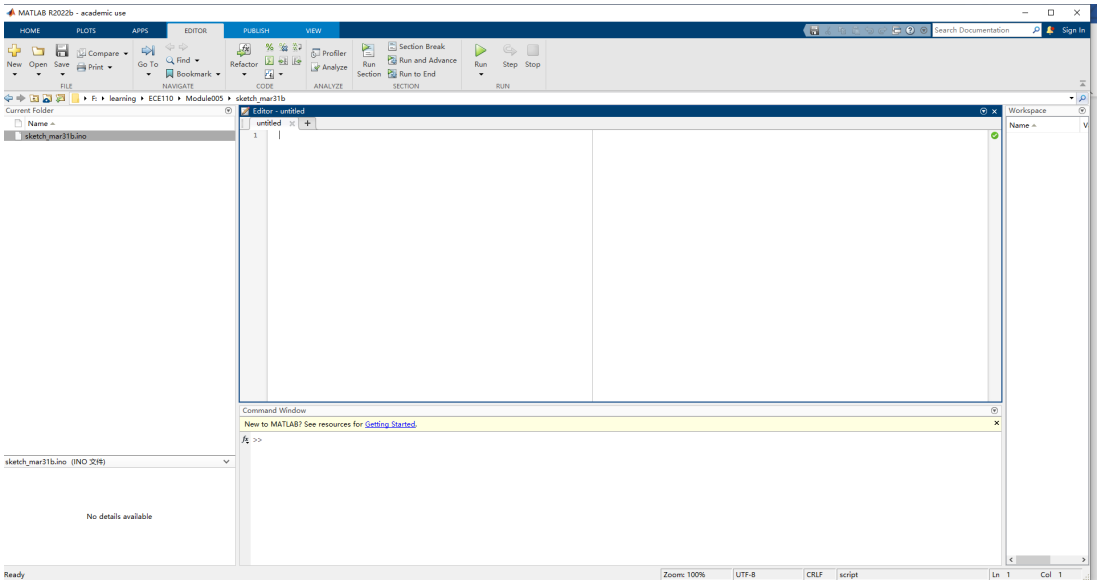


Figure 6: A new instance of MATLAB

Plotting graphs in MATLAB

Throughout the semester you will be asked to perform mathematical operations on your data and to generate your own graphs based on data collected during an experiment. In general, it will be expected that you create these graphs using MATLAB and attach them to the end of your lab report. Given that your graph may be separated from its corresponding portion of the experiment, it becomes very important that you give each graph a useful title and make a note in that section of the experiment so the reader knows where to find it.

Task 1: Below are the basic commands for generating a plot in MATLAB. Go ahead and enter these commands into the Command Window in MATLAB right now. At the command prompt '>' type each line and hit ***Enter***. When the command prompt returns after each line is typed, enter the next line. Notice what changes when you hit Enter in the workspace and history windows.

Note: the semicolon is optional–try entering the first line without it.

```
x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
y = [0, 1,4, 9, 16, 25,36,49, 64, 81, 100];
figure(1);
plot(x, y);
title('Parabola');
xlabel ('X Axis (x units)');
ylabel ('Y Axis (y units)');
```

If you don't already have experience using MATLAB, these commands might be a little cryptic. To shed a little light on the issue, we've broken things down a little further. For further info about any command, you can type "help command" or "doc command" at the command prompt, substituting the appropriate command label.

Further explanation:

- Creating variables

  x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

  This line of the code creates a variable named "x". This particular variable is an array, meaning that it is a collection of variables. Arrays can be multidimensional. The above array is a 1D array, containing 11 values.

  Alternatively, you can create a variable in the same way you would enter data into a spreadsheet. You can do this by using the menu under the Workspace (upper left corner of the MATLAB window) to create new variable. Once you have created and named a variable you can simply double click the variable name in your Workspace and a spreadsheet will open displaying its contents. Once you have this spreadsheet open, you can edit or add to the content associated with that variable name by selecting a cell and entering the desired numbers.

- Creating Plots
  figure (1);

  This command generates a new window in which your plot (figure 1) will appear. This command is not strictly required for creating plots in MATLAB but it is good practice. If a figure is not open before you call the plot function, MATLAB will automatically open one for you–making this command unnecessary. If a figure window is already open, MATLAB will plot into that window and overwrite any graph that may have been there. Use this function to avoid overwriting your plots by accident.

plot (x, y);

This command calls a function and gives it two parameters. The first parameter, x, is used as the horizontal axis and the second parameter, y, is the vertical axis. It is important to note that in this case, the variable names were chosen to illustrate how the plot function works. When you're creating a plot for an experiment, you should give your variables meaningful names (in case you forget what they were for). For example, if our data represented voltage and current measured in Lab 1, you can name your variables something like "voltage_lab1" and "current_lab1."

Taks 2: So we just created a plot with some arbitrary values that we assigned to the arrays x and y. Now let's create two new arrays named voltage and current, using the values in Table 1. Once you've created your arrays, plot the variables.

```
figure(2);
voltage=[THE VOLTAGES VALUES in Table 1];
current=[THE CURRENT VALUES in Table 1];
plot(voltage, current);
```

You can enter your code, line by line at the command prompt. Your code should look like the following, but with your measured values:

At this point you should have a simple plot showing the IV relationship of the single motor from Lab1. Now we need to label our axes and give the plot a title. Doing so is quick and easy from the command line. The commands are shown below. Substitute an appropriate title and appropriate axis labels.

- Title and Labels

```
title('Plottitle');
xlabel('X Axis(x units)');
ylabel('Y Axis(y units)');
```

These function calls should be self-explanatory. Just remember that the text you want as a title or label must have single quotes around it.

Great! At this point you should have a nicely formatted IV plot of your motor from Lab 1. But what if we want to add some additional features or modifications to out plots? Often we may change the color and thickness of the data traces or change the line types to dashes and dots (a good way to make lines distinct when printing out plots in black and white). Any change you want to make can be done from the command prompt, and often that is the preferred practice. But for now, let's

explore some ways to further modify our graph using the MATLAB plot tools.

- Making Your Graph Pretty

  MATLAB provides a very robust graphical interface for changing virtually any parameters of your plots. You can access the ***Property Inspector*** by selecting the icon on the figure window that is shown in the figure below.

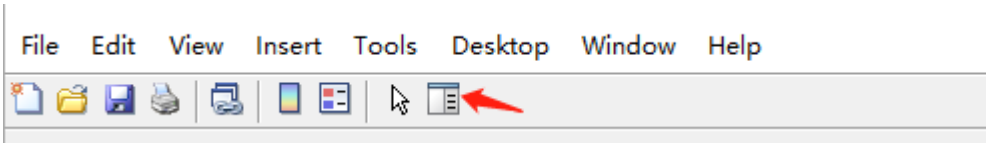


Figure 7: Finding property inspector on the MATLAB taskbar.

Once you have the opened the property inspector, you can change the properties of most aspects of your graph by clicking on the thing you want to change and adjusting the properties displayed in the lower portion of the window. Alternatively, it is possible to adjust any of the plot properties from the command line but it is left to you to use the extensive documentation to find this information if needed.

Task 3: Using the plot tools interface, adjust the setting you're your IV plot so that it looks like *Figure 8*(except with different data). To do this, you'll need to •add grid lines, •change the thickness of the curve, •change the data trace from a solid line to a dashedline, •adjust the font sizes for the title and labels. Don't worry about the exact font sizes and line properties. Also, be sure to **enter your intl ID** into the title.

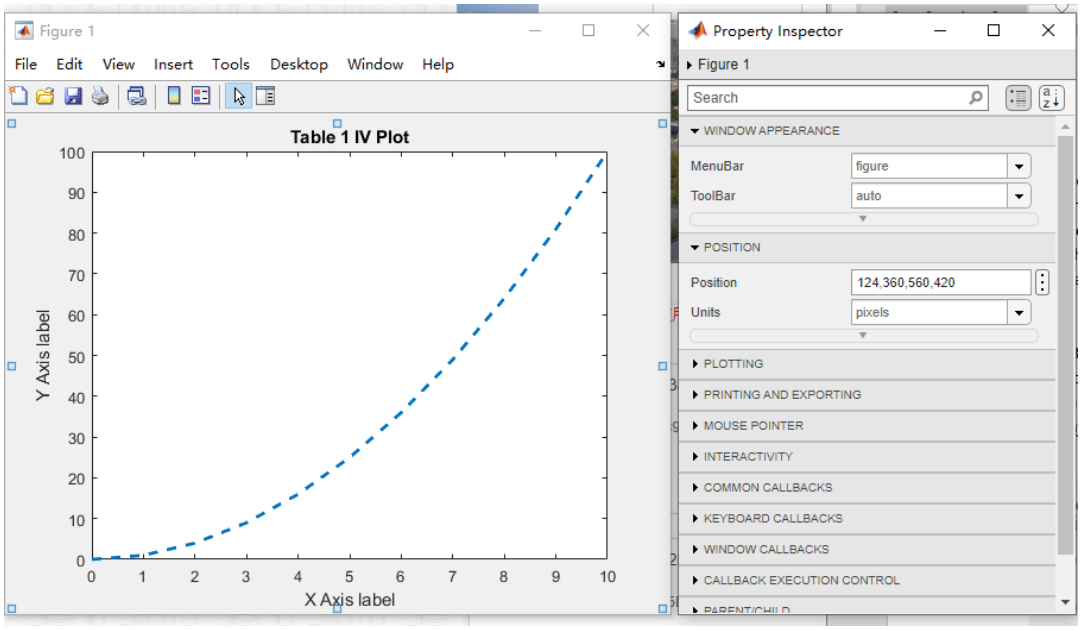


Figure 8: Example plot with attribute checker windows

Now let's try something a bit different.

- Multiple Plots in a Single Graph

Often times, we would like to compare two sets of data by plotting them on the same axes. You can do this using the "hold" command as shown below. To do this, we're going to create another variable, and plot it on the same graph as our previous plot. While we still have our previous plot open, return to the command prompt.

Task 4: Enter the following commands, creating a plot with two different data traces.

```
current2 = 2*current;
hold on;
plot(voltage, current2);
hold off;
```

Note: After entering "hold on", MATLAB will put every curve you plot on the same axes until you enter "hold off".

Task 5: Now we should have another data trace plotted next to our blue dashed line. Now let's give each trace a name, and put a legend on our plot. In the Plot Tools window, click each trace shown in the Plot Browser window on the right. Once the trace is selected, you can enter a name in the Display Name property, shown below the plot. With each plot named, now add a legend by pressing the "Insert Legend" button on the top bar.

With these changes made, your plot should look something like the example show in Figure 9.

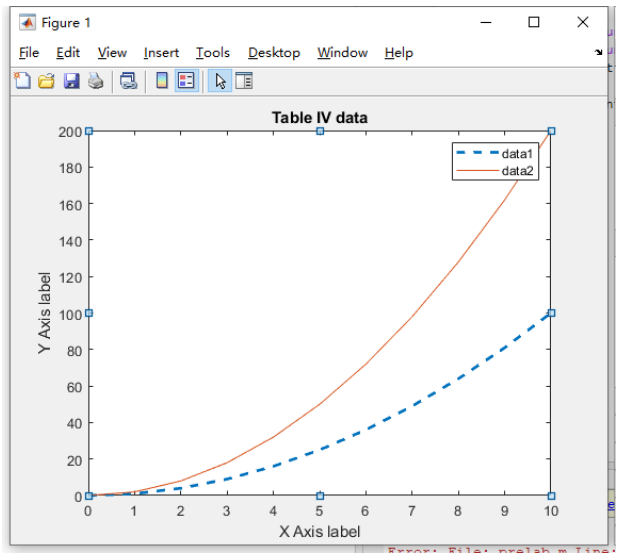


Figure 9: An example plot with two traces

Following all the tasks above, then finish **Question 2.**

Question 5: What does the following line of MATLAB code do?

```
Time = (0:5)*1e-3;
```

Question 6: what happens if you try to plot two variables that do not have the same number of elements in them? (Try it for yourself and see what MATLAB does.)

Introduction to Linear Fit using MATLAB

The Curve Fitting Toolbox is a collection of graphical user interfaces (GUIs) built on the MATLAB technical computing environment. You can perform a linear fit using ***polyfit*** or using a custom equation. Library equations include polynomials, exponentials, rationales and so on. The graphical environment allows you to explore and analyze data and fits visually and numerically.

Polyfit is a function in MATLAB that fits a polynomial to a set of date points. It is simple to use and can save time when working with polynomials in your code. *Polyfit* returns a vector of coefficients representing the fitted polynomial in descending order. The syntax for *polyfi*t is as follows:

```
p=polyfix(x,y,n)
```

where x and y are vectors of the same length representing the x- and y-coordinates of the data points, respectively, n is the degree of the polynomial, and p is a vector of coefficients representing the fitted polynomial in descending order.

Linear fit example:

Fit a simple linear regression model to a set of discrete 2-D data points.

Create a few vectors of sample data points *(x,y)*. Fit a first-degree polynomial to the data.

```
x = 1:50;
y = -0.3*x + 2*randn(1,50);
p = polyfit(x,y,1);
```

Evaluate the fitted polynomial p at the points in x. Plot the resulting linear regression model with the data.

```
f = polyval(p,x);
plot(x,y,'o',x,f,'-')
legend('data','linear fit')
```

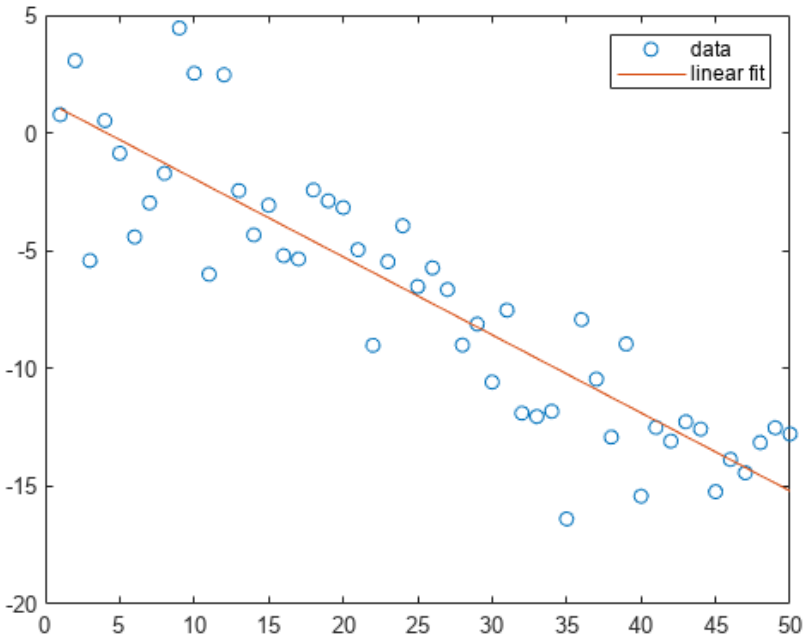


Figure 10 : A fit example of MATLAB

The fitted polynomial is of degree 1 (a straight line).

Power and Energy

Batteries are devices that store electrical energy typically by either exploiting a chemical process. In a circuit schematic, sources of electrical power can be distinguished from power sinks (or loads) by the relationship between the polarity of the voltage across the device and the polarity of the current flowing through it.

$$P = V\,I$$

Or, more specifically, if V+→- is the voltage drop across a device in the direction of the current's polarity arrow:

$$P_{device} = V_{device} \times I_{device} = V_{+\rightarrow -} \times I_{\rightarrow} \text{ which is } \begin{cases} < 0 \; if \; power \; source \\ > 0 \; if \; power \; sink \end{cases}$$

Power and energy are two related concepts often confused by the aspiring engineer. Often energy is first learned in the potential and kinetic forms. It is conserved in that it is neither created nor destroyed, but instead may incur transformations in form. It has units of joules (J). A charge-storing device, like a battery or a capacitor, is often referenced by the amount of energy it stores. A device that utilizes energy for long periods of time, like a light bulb, are often referenced by the amount of energy they utilize each second. Power is the amount of energy expended over time. It has units of Watts (1 Joule/sec). In this way, a 60-Watt incandescent bulb transforms 60 Joules each second into light and heat.

You have been supplied with two rechargeable 3.6-V Lithium batteries with a 2800 milliamp-hour (mAh) rating. The batteries will deliver 1 milliamp at 7.2 volts for 2800 hours while running continuously. Significant variations between batteries will occur due to deviations in manufacturing, the age and current-supplying limitations imposed by the materials comprising the battery. Later, we will find that the two motors used in our car project consume a large majority of the overall circuit power and we can use this fact to estimate the battery life.

Charging the Battery

It is important not to attempt to charge a battery faster than the chemical reaction can comfortably occur. For Lithium batteries, the suggested charging rate is from $0.5 \times C$ to $1.0 \times C$ where C is the energy rating designated on the battery in mAh. Since our battery is rated at 2800 mAh, a 1-ampmaximum charging current should be a good, safe choice.

Question 7: Calculate how long the battery should charge if it were completely dead before charging continuously at 1.0 A. Assume that the battery has a charging efficiency of 66% (66% of the total current goes toward charging increasing the potential energy of the battery while the rest is wasted, perhaps as heat).

Question 8: State the conservation of energy equation ($E_{input} = \Delta E_{state} + E_{waste}$) as it applied to charging the battery. Give numeric values in Joules for the energy input, change in state, and waste.