Keyboard Emulation Data Logger

By:
Scott Leonard
Mattics Phi
Greg Alesso
Norman Tran

Faculty Advisor: Dale Dolan

Senior Project

ELECTRICAL ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

2013

Table of Contents

List of Tables:

List of Figures:

ABSTRACT

The purpose of this project is to design and build a low cost, portable USB data logger for electrical engineering labs. This data logger connects to any computer through a USB port and appears as a keyboard to the host's operating system. The logger performs electrical measurements and records the data by emulating keyboard strokes inside any spreadsheet program or analysis tool without the need for software or ancillary hardware. The logger is capable of measuring voltage and current. The end result of this project is to create a low cost, easy to use solution for data collection that can be replicated by other students.

Introduction

The purpose of this senior project is to design and build a low cost USB keyboard emulating data logger for electrical engineering labs. This device will measure voltage and current through a variety of common ranges using a microcontroller component with an integrated ADC. This microcontroller will then send measurement values to a computer through a USB port where they will be recorded directly into a spreadsheet program. The USB communication component will be implemented using the Human Interface Device (HID) protocol to ensure compatibility with most personal computers without special logging software. The end goal of this effort is to produce an affordable, potentially commercial product that will help other students and experimenters with their electrical engineering projects.

The idea behind the USB data logger is to produce a product that can measure voltage and current signals and enter them directly into a useful program such as Microsoft Excel. Most data manipulation in undergraduate Electrical Engineering labs are done with the help of a spreadsheet program. The product will connect via a USB drive straight into a PC. The device will consist of an input stage made up of op-amps, resistors and switches that regulate the voltage and current inputs. The main idea is not to create a high-powered multi-meter, rather a versatile, useful device that can generate the necessary data without superfluous programs running. This product should be small and should work much like a keyboard or mouse would, being connected simply through a USB port, using USB protocol.

Another important performance parameter of the data logger is the compatibility and mobility it presents. The device is small enough to transport around from place to place. It can be used on laptops, MAC's and PC's without requiring special software or drivers to be installed.

Background

Team Overview:

Dale Dolan	**Faculty Advisor**	**Power Department Faculty**
Mattics Phi	**Software and Programming**	**Electrical Engineering Student**
Scott Leonard	**Project Leader**	**Electrical Engineering Student**
Greg Alesso	**Hardware and Sensing**	**Electrical Engineering Student**
Norman Tran	**Software and Programming**	**Computer Engineering Student**

Data logging devices on the market vary in prices based on the accuracy and the utility of the device. As seen below in Table 1, some existing devices are listed along with their common features. The reason for the wide price range is because some of the devices are designed with more accuracy, functionality, and usability. The higher priced devices are used more in industry rather than commercially and even then, the commercial devices are expensive. This prevents students and hobbyists from purchasing devices that might be of use to them. By keeping the price of this device low and affordable, the target market will be broader to increase the amount of consumers while providing students and hobbyists with devices that are made for students, for students, and created by students. The diagram below in figure 1 shows the major components of the project.

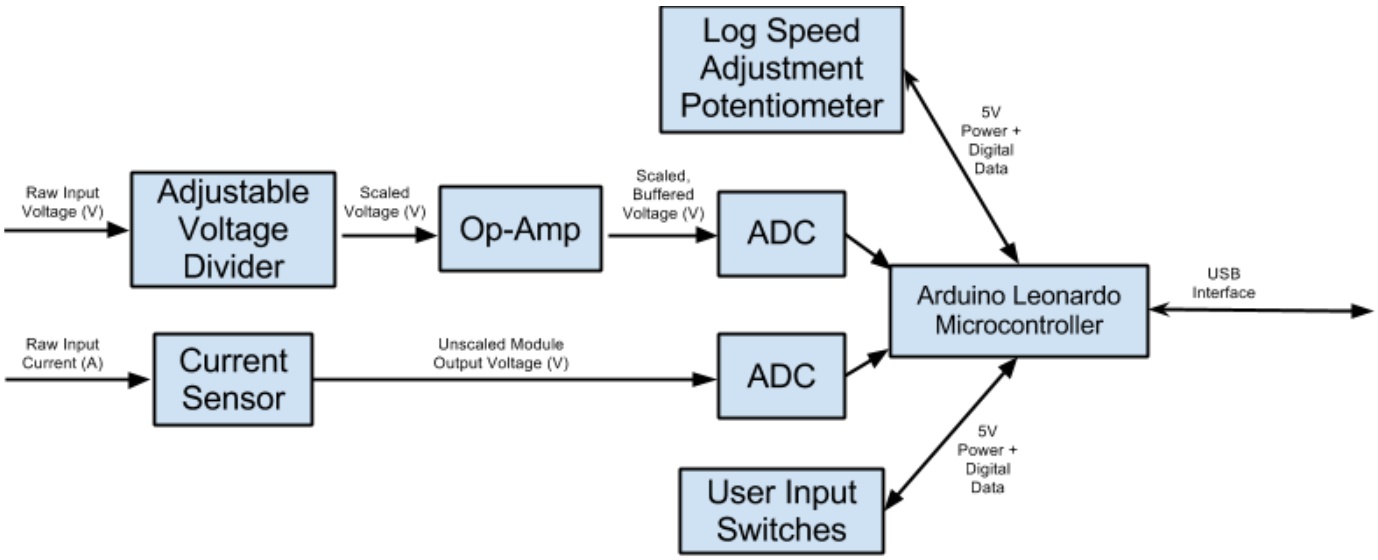


Figure 1: Block Diagram of System Components

Table I: Summary of Existing Data Logger Systems

Product Name	Software Required	Listed Price	Interface	Range	Input Channels	Portable	Other Features
AVR Stick	NO	$9.99	USB	Voltage (0 to 2.56 volts)	1	NO	None
EL-USB-3 Voltage Logger	YES	$75.25	USB	Voltage (0 to 30 volts)	1	YES	None
NI USB-600	YES	$279.00	USB	Voltage (-10 to 10 volts)	8	YES	Lab on a chip: digital IO, Counters, PWM
Fluke 289 Industrial Logging Meter	Yes	$558.24	USB	Voltage (50 mV to 1000 V)	1	YES	Very Wide Range of Features
Lascar Direct Connect Data Recorder	YES	$67.00	USB	Voltage (0 to 30 volts)	1	YES	None
Bipolar Voltage DC Simple Logger	No	$379.00	USB	Voltage (-850 to 850 volts)	1	YES	Long term voltage reading
Micro II USBData Logger	YES	$84.00	USB	voltage (0 to 10 volts)	1	YES	LCD display
SD Voltage Data Logger	YES	$495.00	SD Card	Voltage (0 to 1.5 volts)	1	YES	SD card interface
Supco Current &Voltage Data Logger	Yes	$411.93	USB	Voltage (0 to 500 V)	1	Yes	Current, alarms, three phase
SI Voltage Data Logger	Yes	$249.00	USB	Voltage (0 to 5)	1	Yes	None
SI True RMS AC Voltage/Current Datalog	Yes	$309.00	USB	Voltage (10 to 600 V)	2	Yes	LCD

Requirements and Specifications

General Requirements:

1. **Easy and practical to operate**
2. **Works without external software**
3. **Device will have a low cost**
4. **Device will be durable**
5. **Measures useful values in lab**
6. **Compatible with common spreadsheet programs**
7. **Compatible with a personal computer**

Table II: Detailed System Requirements

General Requirements	Specifications	Justification
1-4, 7	**The device is handheld and portable, weighing no more than 1 pound and no more than 12 cubic inches.**	**Since the device will mostly be used in lab environments, being handheld and portable allows the user to move it to various areas without needed external equipment.**
1-4, 6, 7	**The device is implemented with widely available components.**	**The device will need to fit the budget of a student and can be replicated without special tools.**
1-3, 8	**Device can be implemented as a voltmeter or current meter.**	**The specifications of each of these measurements are described below.**
5	**The voltmeter reads up to 25v with up one decimal place of accuracy.**	**This allows students and hobbyists to measure and log voltage values into the computer with accuracy.**
5	**The current meter reads up to 5 A with DC or AC current, providing up to one decimal places for accuracy.**	**This allows students and hobbyists to measure and log current values into the computer with accuracy.**

1-3, 5-8	**Device is interfaceable with USB 2.0, Excel, and Matlab using included software package.**	**Most lab environments contain computers with Excel and Matlab. Being able to interface with computers with these programs allows the user to log and utilize the data more efficiently.**
1, 2, 5, 7	**Device can be powered from USB.**	**This removes the need for an external power source. Since device will interface with a computer, this makes the device more practical.**
2-4, 7	**Device costs no more than $40.**	**Keeping prices low makes this device a competitor with other devices currently on the market and fits a student's budget.**
4,5	**Device is able to operate under normal operating conditions (0°C to 70°C).**	**Allows the device to be used under various conditions while still being operable.**
4,5	**Device will include over-voltage protection features at inputs**	**By providing over-voltage protection, the user will avoid accidental damage to the device.**

Design: Hardware

Component list:
(2) 400 Pin Prototyping Board
(3) 39k Resistors
(1) Rail to Rail Quad Op-Amp IC
(1) 4 Input DIP Switches
(6) Banana Jack Terminals
(1) 5 Amp Hall Effect Current Sensor
(1) 14 Pin IC Socket
22 AWG Assorted Wire Solid Core & Solder
(3) 10K Ohm Blue Trim Pot (15 Turn)
(1) Log Speed Trim Pot w/Knob
(16) IC Breakoff Male Headers
(1) USB Micro Cable
(1) On/Off Recording Switch
(1) Arduino Leonardo

The hardware design of this project is divided into several different sections. They were designed separately at different stages in the project and integrated together in the final quarter of work.

The first part of this design process was the input stages for the data logger. Considering that our microcontroller can be easily damaged from input voltages over 5 volts, the use of an op amp was crucial to buffer the input stages. This op-amp would be powered directly from the 5v supply available from Arduino, which is ultimately provided by the host computer's USB port. It needed to have rails from 5V all the way down to 0V, which requires a rail-to-rail op-amp. Knowing that there would be multiple inputs, quad op-amps were bought and tested. The quad op-amp package was selected because of its wide availability, value, and through hole package options. Several of these were suitable, so the decision came down to cost. The TI 86A019M was finally settled upon. The input voltage range was chosen to be larger than 5 volts, so the input stage was designed as a resistive divider flowing into a voltage follower. An in depth look of is shown in **Table III**.

Table III: OP-AMP Selection

Name	Ground Output	Rail Output	Digikey Price	Absolute Max Input Voltage
TLV2741	0	5.07	$1.63	vdd+0.2
23741	0	5.09	$2.25	vdd+0.2
MCP6004	0	5.07	$0.48	vdd+0.3
MC33204P	0.04	5.05	$1.61	not stated
TLC2274AIN	0	5.07	$2.48	vdd+0.3
LMC660CN	0	5.06	$2.38	vdd+0.3
LMC6484IN	0.01	5.07	$2.90	vdd+0.3
MCP604	0	5.07	$1.25	vdd+0.3
MCP6024	0.01	5.06	$2.08	(vdd+0.3)
LM324N	1.43	3.799	$2.34	32.0 volts
LS404CN	0.54	4.485	$1.24	32.0 volts
LM2902N	1.80	3.777	$2.65	32.0 volts
TL074CN	1.399	4.472	$3.07	16.0 volts

The voltage range was decided to be 0 - 25V. This was done with a 39K ohm/10K ohm resistive divider(seen in **figure 2**.) Three voltage input stages were chosen for a specific reason. Generally, in electrical engineering labs, the input voltage and the output voltages are measured. The third input stage is to allow the user to measure a voltage difference, while measuring a voltage with an uncommon ground.

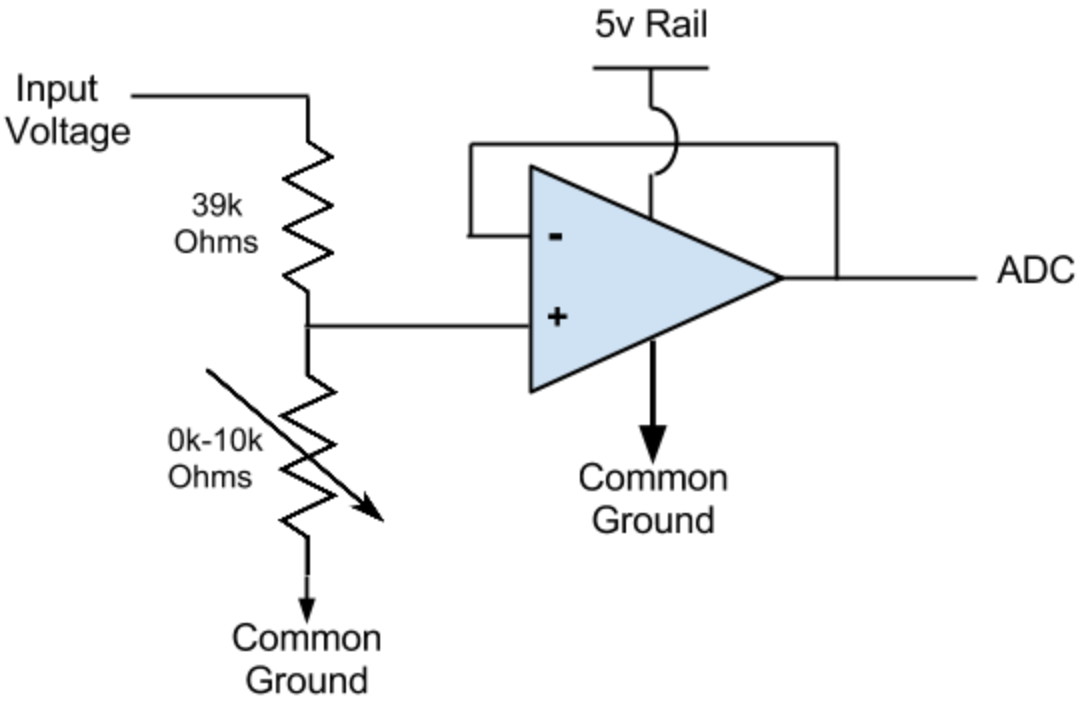


Figure 2: Input voltage stage circuit diagram

The current sensor(**Figure 3**) was chosen based on price and functionality. It can only handle 0-5A, but this was determined to be sufficient for general circuit use. This device was chosen not because it was a high-powered, high-functioning sensor, but because it was the most cost-effective sensor that gave an acceptable current range. The current device (shown below) is connected to the common ground and the 5V output of the Arduino Leonardo. The output signal from the module is connected directly to one of the ADC ports.

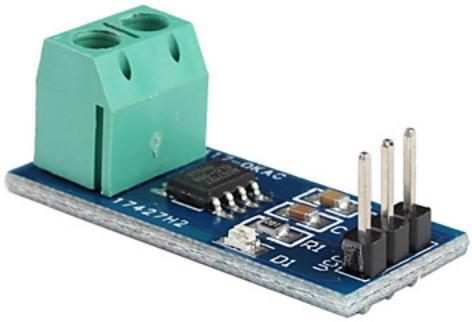


Figure 3: ACS712 LC Electronics 0-5Amp AC/DC Hall Effect Current Sensor Module

The Arduino Leonardo was chosen for its useful capabilities. A sacrifice was made with this decision. The two options were either to use a low-functioning microcontroller pic, or to use the Leonardo. The Leonardo was chosen, despite price, because it came with a built-in keyboard emulator. The Leonardo is interconnected with the input stages. The op-amps get their 5V and 0V rails from the controller's 5V output and the common ground found on the device. This allows the entire project to be powered by USB power from the computer.The device gets the 5V from its USB connection.

The device takes in the 0-5V input from the op-amps and current sensor, converts them into digital values through a built in ADC, and then outputs a manipulated version to give accurate values.

The Arduino Leonardo is also connected to certain user-operated external components. Switches are connected to the device's digital inputs; these switches turn on and off the recording of the voltage and current inputs. The main switch turns on/off keyboard emulation. This allows the user to prepare the Excel or spreadsheet program before turning on logging and having the Arduino take over control of the keyboard. The user also has the option to change the recording speed. This needs to be available due to the varying speeds of different computers. This is designed with a potentiometer connected to an analog input. This varied value will increase or decrease the logging speed, so the user can see real-time change of the measure circuit.

Design: Software

The software for the data logger is written using Arduino sketch. The software is configured to work best with Microsoft Excel, however it functions fine in Notepad, Wordpad, or other text editors. The major functions of the software are broken down into several parts below.

Setup

First, in the setup() function, digital pins 9 through 13 are set as inputs with their internal pullup resistors activated. This sets the “on” switch as LOW and “off” as HIGH. Pins 9 through 11 act as switches to turn on/off data logging for the voltage inputs, pin 12 acts as a switch to turn on/off data logging for the current input, and pin 13 acts as a switch to turn on/off keyboard entry. This switch is intended to prevent unintended keyboard entries because the Arduino takes over the keyboard once it is switched on. The setup() function is only run once.

```
void setup()
{
  Serial.begin(9600);          //  Set up serial
  // Input pullup resistors used instead of external pulldowns
  // HIGH = OFF, LOW = ON
  pinMode(9, INPUT_PULLUP);    // Switch for analog pin 2 (Voltage 1)
  pinMode(10, INPUT_PULLUP);   // Switch for analog pin 3 (Voltage 2)
  pinMode(11, INPUT_PULLUP);   // Switch for analog pin 4 (Voltage 3)
  pinMode(12, INPUT_PULLUP);   // Switch for analog pin 5 (Current)
  pinMode(13, INPUT_PULLUP);   // Switch for analog pin 0 (Potentiometer)
  Keyboard.begin();            // Set up keyboard entry
}
```

Loop

The loop function acts like the main function, but runs continuously. First, digital pin 13 is checked to see if it is on or off. It is on when it is LOW. The flag variable determines whether or not to print the header. It is only printed once at the very top. Then, the adjustDelay() function is called to determine the logging speed. Digital pins 9 to 12 are read to determine if their switches are turned on. If they are, the voltages are read using the logVoltage() function. The value read from the current pin is adjusted and scaled for accuracy before it is printed to the screen. Likewise, the voltage values returned from logVoltage() are scaled by a factor of 5.1 in order to produce accurate results. A *tab*

keystroke is entered to move to the next cell to the right in Excel in between data readings. Lastly, the delay in milliseconds is printed out and an *enter* key is entered to move down to the next line. The logVoltage() and adjustDelay() functions will be explained in depth below.

```
void loop()
{
  if (digitalRead(13) == LOW)  // Pin 13 must be grounded to start writing
  {
    if (flag == 0)  // Print out header at the top
    {
      flag = initializeHeader();
    }
    adjustDelay();  // Set logging speed by adjusting delay

    for (i = 5; i >= 2; i--)
    {
      if (digitalRead(i+7) == LOW)
      {
        // Log voltages from analog pins 2-4, current from analog pin 5
        if (i == 5)  // Current pin
        {
          voltVal = readVoltage(i);  // Read voltage
          curVal = (voltVal-2.56)*5.555;  // Scale value for current
          Keyboard.print(curVal);
        }
        else  // Voltage pin
        {
          voltVal = readVoltage(i)*5.1;  // Read and scale voltage
          Keyboard.print(voltVal);
        }
      }
      if (i == 2)  // Reached end of line
      {
        insertTab();
        // Print logging speed in ms (0 to 200)
        Keyboard.print(logDelay);
        insertEnter();
      }
      else
      {
        insertTab();
      }
    }
  }
}
```

Functions

The readVcc() function is used to improve the accuracy of the ADC in the Arduino. It uses the internal 1.1V reference to calculate the Vcc. This is done because the USB voltage of different computers will vary and will not be exactly 5V. The result is returned in millivolts for more accuracy. This function was taken from http://provideyourown.com/2012/secret-arduino-voltmeter-measure-battery-voltage/

```
long readVcc() {
  // Read 1.1V reference against AVcc
  // set the reference to Vcc and the measurement to the internal 1.1V
reference
  #if defined(__AVR_ATmega32U4__) || defined(__AVR_ATmega1280__) ||
defined(__AVR_ATmega2560__)
    ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  #elif defined (__AVR_ATtiny24__) || defined(__AVR_ATtiny44__) ||
defined(__AVR_ATtiny84__)
    ADMUX = _BV(MUX5) | _BV(MUX0);
  #elif defined (__AVR_ATtiny25__) || defined(__AVR_ATtiny45__) ||
defined(__AVR_ATtiny85__)
    ADMUX = _BV(MUX3) | _BV(MUX2);
  #else
    ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  #endif

  delay(2); // Wait for Vref to settle
  ADCSRA |= _BV(ADSC); // Start conversion
  while (bit_is_set(ADCSRA,ADSC)); // measuring

  uint8_t low  = ADCL; // must read ADCL first - it then locks ADCH
  uint8_t high = ADCH; // unlocks both

  result = (high<<8) | low;

  result = 1125300L / result; // Calculate Vcc (in mV); 1125300 =
1.1*1023*1000
  return result; // Vcc in millivolts
}
```

The logVoltage() function takes in the analog pin number as a parameter and returns the voltage reading as a double. It calls the readVcc() function to determine the Vcc and then divides by 1000 to get the reading in volts. Then, the analog pin is read, returning a 10-bit

ADC value from 0 to 1023. This value is divided by 1023 and multiplied by Vcc to get an accurate voltage measurement.

```
double readVoltage(int anPin)
{
  vcc = readVcc()/1000.0;  // Read Vcc
  digRead = analogRead(anPin);  // Read analog pin
  delay(10);
  voltage = ((digRead/1023.0)*vcc);  // Calculate voltage
  return voltage;
}
```

The adjustDelay() function reads the pin tied to the potentiometer, which is designated as analog pin 0. The range for this value was measured to be from 0 to 892. The delay() function is then called using the potentiometer value divided by 4.46. This produces a delay between 0 ms to 200 ms.

```
void adjustDelay()
{
  potVal = analogRead(potPin);  // Read the potentiometer value
  // Potentiometer reading should be between 0 and 1023
  logDelay = potVal/5.115;  // Gives a delay between 0 and 200 ms
  delay(logDelay);
}
```

TESTING

Test Objective: The purpose of these tests is to determine the accuracy of the device, as well as to show the device's capabilities in recording values of actually test circuits.

Procedure:

Test 1
A. Insert power supply straight into terminals 1, 2, and 3.
B. Connect a separate voltmeter across the power supply terminals.
C. Increase the voltage up to 26 V in increments of 1 V.
D. Run the device and compare the values.

Test 2
A. Connect the current source through an electric load and a power supply.
B. Set the current at 0 A and then increase to 3 A.
C. Let the device record the current values.
D. Compare to the electric load values.

Test 3
A. Set up circuit: connect power supply through an electric load.
B. Connect the current through the current sensor terminals.
C. Put the from the power supply through Voltage input 1.
D. Set the current to an arbitrary value, 1 A.
E. Record the current and Voltage values.
F. Increase the voltage to get an acceptable range of data.
F. Using the excel function graph the power vs. load voltage.

Data and Graphs

Table IV: Voltage accuracy test data

v1	v2	v3	Avg.	Calibration	Difference
0.05	0.05	0.05	0.05	0.1	0.05
1.05	1.05	1.05	1.05	1.099	0.05
1.95	1.95	1.94	1.95	1.979	0.03
3.02	3.02	3.02	3.02	3.056	0.04
4.05	4.05	4.07	4.06	4.087	0.03
5.05	5.05	5.05	5.05	5.078	0.03
5.99	5.97	5.97	5.98	5.99	0.01
7.07	7.07	7.07	7.07	7.084	0.01

8.22	8.2	8.22	8.21	8.222	0.01
9.22	9.22	9.22	9.22	9.222	0
10.22	10.19	10.19	10.2	10.207	0.01
10.99	10.99	10.94	10.97	10.99	0.02
12.09	12.09	12.06	12.08	12.091	0.01
13.06	13.04	13.06	13.05	13.06	0.01
14.11	14.11	14.14	14.12	14.111	-0.01
15.24	15.24	15.24	15.24	15.226	-0.01
16.11	16.11	16.11	16.11	16.091	-0.02
17.06	17.03	17.06	17.05	17.034	-0.02
18.08	18.08	18.08	18.08	18.072	-0.01
19.24	19.24	19.26	19.25	19.224	-0.03
20.26	20.24	20.24	20.25	20.23	-0.02
21.18	21.16	21.18	21.17	21.16	-0.01
22.21	22.18	22.13	22.17	22.174	0
23.23	23.21	23.23	23.22	23.2	-0.02
24.26	24.23	24.23	24.24	24.234	-0.01
25.13	25.1	25.24	25.16	25.1	-0.06
26.2	26.2	26.2	26.2	26.159	-0.04

Table V: Current accuracy test data

Low current recorded(A)	Largest Current recorded(A)	Actual Current (A)	Biggest Difference
-0.13	0.04	0	0.13
0.02	0.21	0.1	0.11
0.31	0.46	0.4	0.09
0.64	0.79	0.7	0.09
0.94	1.07	1	0.07
1.27	1.4	1.3	0.1
1.52	1.76	1.6	0.16
1.82	2.01	1.9	0.11
2.04	2.26	2.1	0.16
2.35	2.56	2.4	0.16
2.65	2.89	2.7	0.19
2.98	3.14	3	0.14

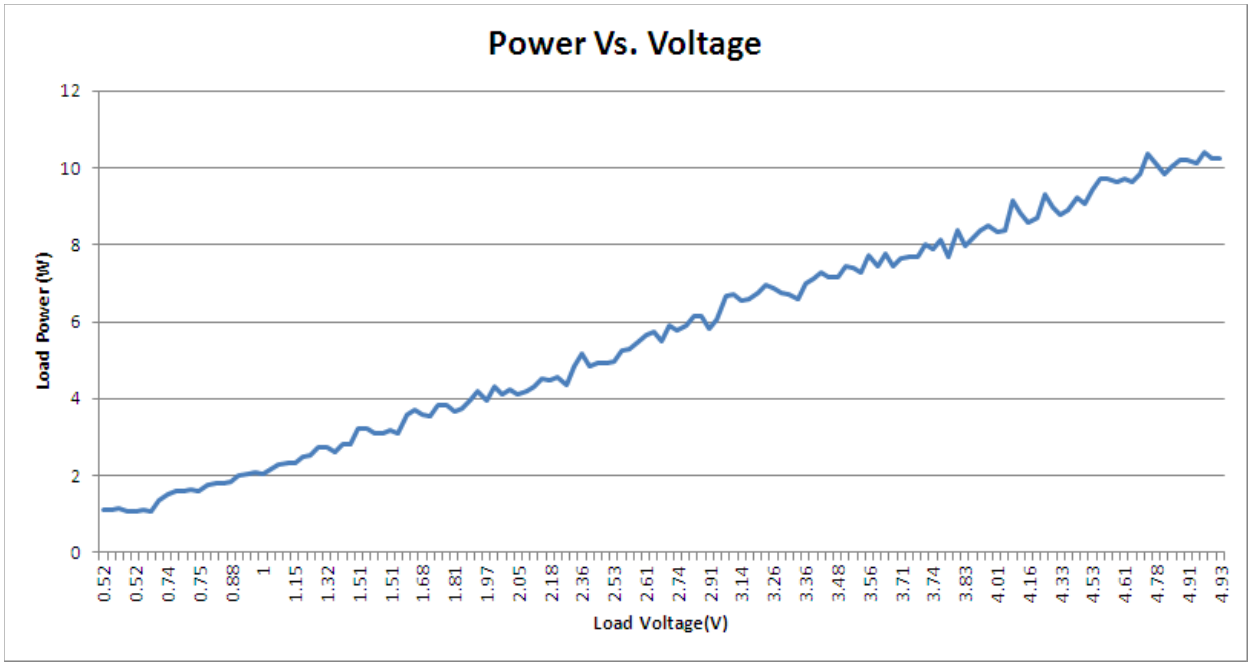


Figure 4: Test 3 Power Calculation Graph Data

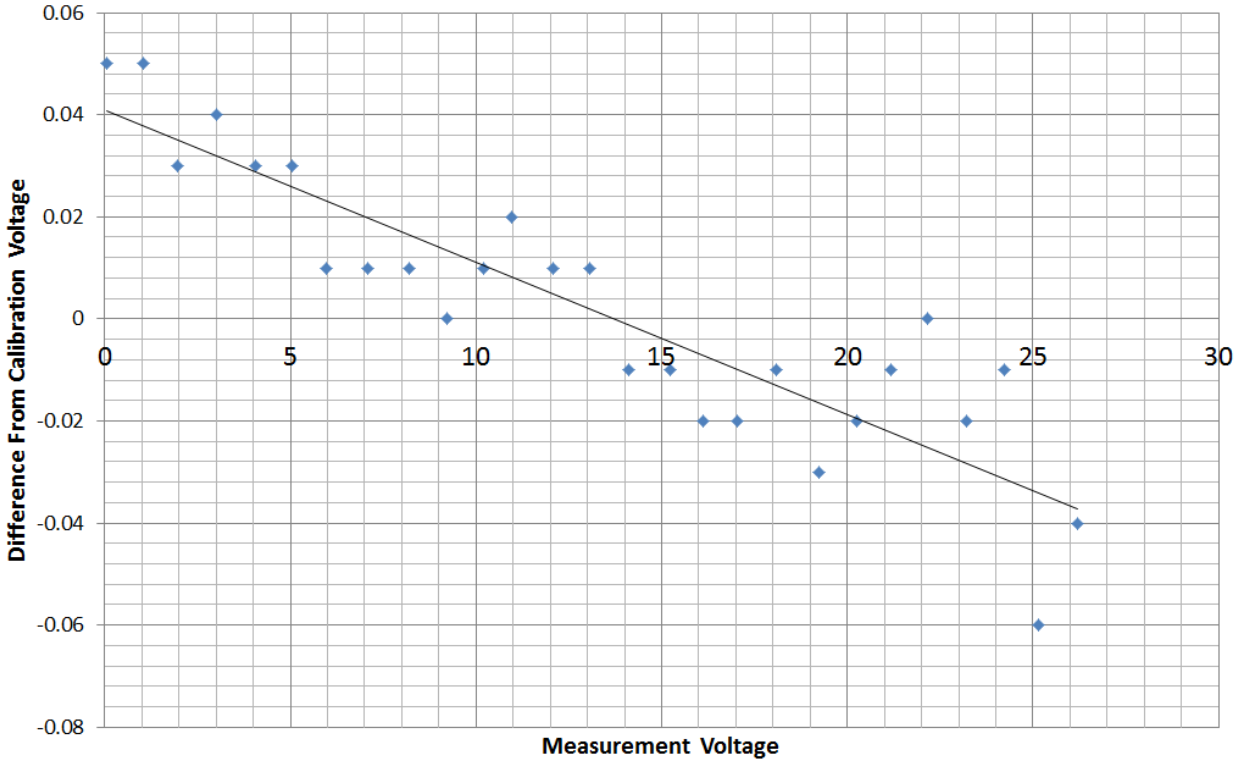


Figure 5: Accuracy vs. Calibration Voltage

Results and Analysis

Test one was performed to give an accuracy value of the device. Accuracy was generally determined using a benchtop multimeter, recorded above as calibration voltage. The three terminals, which were all calibrated individually, produced roughly the same values. The difference between the logged values and the values being recorded manually was never more than .05V(see **Table IV**. This lets us say with confidence that the accuracy of the device is +/- 50 mV. This is an acceptable value and meets the design requirements we set out to match. Figure 5 above shows the difference between the logger's measurements and the calibration voltage.

Test two was designed to test the accuracy of the current sensor and the device. This was done using an electric load. The current was was varied for a range of data 0-3 A.
The device had a hard time getting a constant current, even though the current inputed was constant. The data in **Table V** shows a large range of difference. The largest difference was .19 V from the actual amperage. This lets us say that our device has an accuracy of +/- 200 mA.

Test 3 provides an example function of the data logger. The purpose was to record the current and the voltage across a specific load. The data was then graphed against the increasing voltage, while keep current constant. The result can be seen in **Figure 4**. This test offers one of the main appeals of the project. Because the data was already in Microsoft Excel, it was a simple task multiplying the current and the voltage to obtain power. Selecting and graphing the two axes was also very simple and fast. It can be observed from the graph above that the device does its job adequately.

Development and Construction

This project was developed as two major systems: hardware involved with the data logger shield and software for the Arduino microcontroller. A summary of the major development steps is shown below in figure 5:

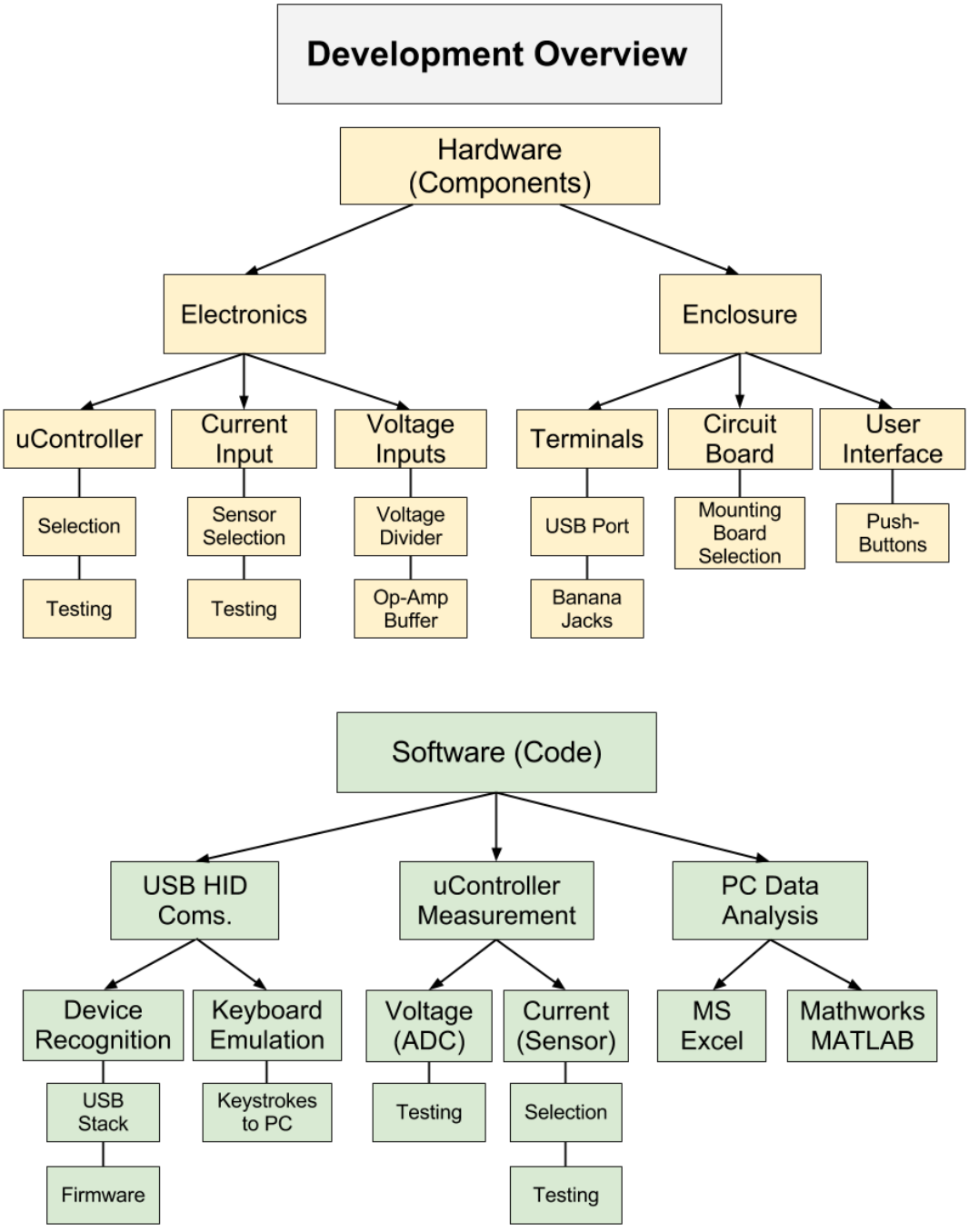


Figure 6 : Development Overview

This data logger was implemented with commonly available components with the hope that it could be assembled by the average hobbyist or student with minimal experience. The complete construction process is detailed in the appendix.

The most difficult steps in the device's construction involve solder connections in close proximity to one another. The construction process for this device could be greatly simplified by etching a printed circuit board with copper traces between the connections. This extra step in construction would be well worth the effort from the gain in reliability and ease of assembly.

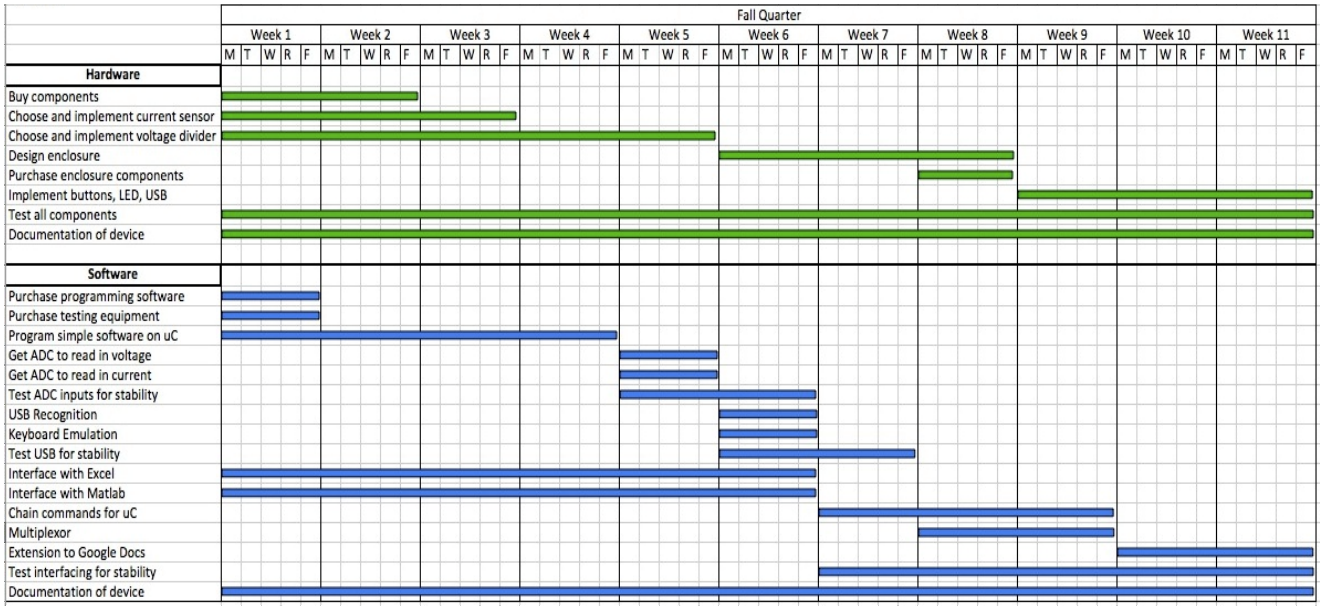


Figure 7 : Fall Quarter Gantt Chart

The goal of Fall Quarter was to decide what core components will be needed for the project, as shown above in figure 6. From there, we created and tested the most basic sub-assembles of the system: the voltage divider, current sensor and the ADC input into the microcontroller. By the end of Winter Quarter, the device was tested for basic functionality which included reading voltage and current values. The device should also be tested for data-logging and communication to the computer through USB in order to operate with the data-logging software.

Throughout the quarters, there will be a lot of design/implement/testing cycles that both the hardware and software teams will go through. This allows the team to focus on other goals at the beginning of Winter Quarter with the knowledge that the device is able to operate with basic functionality. Spring quarter was dedicated to documenting the device and fine tuning its performance.

Conclusion

This device meets the requirements and goals set at the start of the project. The first requirements were ease of use and portability. The device is small enough to transport from place to place, lightweight, and is stable enough not to break or fall apart. The device also requires no special software uploaded by the user. Once the code is written onto the microprocessor, it will function on any computer. The data logger also utilizes just one common micro USB to USB male-to-male cable to connect to a computer and once connected, it is able to log data at the flip of a switch. The device itself is easy to understand. It has clear terminals for voltage and current inputs as well as an easy-to-see switch system to control when the device records and doesn't record.

Another criteria was price (see appendix B.) The total price of the hardware and software needed to build this product was $34.37, this is a slightly higher price than the initial estimate. We first envisioned a price of around $20. This became a enlarged when we changed the design to include an Arduino Leonardo. This microprocessor is a significant upgrade from the PIC originally chosen in terms of easy usability. The Leonardo offered more features to help with the keyboard emulation, and therefore the increase in price was deemed acceptable. A final price of $34.37 still meets our requirements. The market for this type of device is substantially above our component price. When considering the customer for this device, a college student with limited funds, it was important to make this an affordable alternative. This price was met and this was done while still meeting acceptable accuracy numbers.

A small sacrifice was made to keep the cost low. Although the accuracy of the device isn't as high powered as some of the other alternatives, it is able to produce accurate data for what it is meant to measure. By doing several calibration and testing steps, the voltage and current accuracy readings were obtained to be +/- 50mV and +/- 200mA respectively. By choosing a voltage range from 0-26 V and having 3 input stages, it allows the user to do several different things. The wider range was deemed to be the most useful range for an electrical engineering student in an electronics lab. Voltages don't commonly exceed 26V and usually hover around the 8-12 V range. Having multiple input stages allows for measuring input and output voltages simultaneously; this allows users to graph the two columns versus one another. Similarly, the ability to record current as well as voltage means that power graphs can be graphed and analyzed.

Another large requirement that was met was the ability to record something straight into Microsoft Excel or a similar program. The device acts as a keyboard, printing values one keystroke at a time. This is practical in the world of electronics labs. Almost all data manipulation is done within Excel. As mentioned above, the ease of multiplying two columns and then graphing it right away is invaluable. Many data logging products use their own interface and software to print values. But the idea is that those values will just be posted into Excel anyway, so this feature cuts out the middle step.

Overall this device can be defined an affordable, portable and practical voltage and amperage data logger. That is what we set out to design and create. The finished product is above satisfactory when compared to what we envisioned at the start.

Appendices
A. Senior Project Analysis

Summary of Functional Requirements

The goal of this project is to create a data logging device for electrical engineering that is small, portable, and easy to use for students and hobbyists. To meet this goal, this logger will be implemented on a low cost microcontroller platform with USB compatibility. This device will measure voltage and current through a variety of common ranges using a microcontroller component with an integrated ADC. This microcontroller will then send measurement values to a computer through a USB port, where they will be recorded directly into a spreadsheet program. The USB communication component will be implemented using the Human Interface Device (HID) protocol to ensure compatibility with most personal computers without special logging software.

Primary Constraints

Three main performance parameters constrain our design:

1. Input range vs. input accuracy
2. Number of inputs vs. sampling rate
3. Cost vs. functionality

Input range and input accuracy form a tradeoff that must be addressed for each of the inputs on the microcontroller. Our system relies on ADC's to approximate a digital value given an analog input. This means that for any given range of voltages, there is a fixed number of bits that can be approximated from the value. In our case, this measurement is limited to 10 bits, so for any given measurement range, the possible voltages are divided into 2^10 possible values. As the input range increases, these states become further apart, decreasing accuracy.

Another constraint is the trade-off between input number and sampling rate. Our data logging device must sample from multiple inputs, providing a reading for each. As a result, some input values may change as the device is waiting to read them, causing a loss of information. This creates inaccuracies in measurements that need to be addressed, especially when measuring audio and control signals. As the number of input devices increases, the max sampling rate for the inputs is reduced. This problem can be overcome by allowing the user to specify which input sources are to be measured and which are to remain inactive.

Finally, the most important trade-off for our device is the cost vs. its functionality. The end price of the device determines how many people can afford to use it, but the functionality is what sets it apart from other products. The reason for keeping the cost low is to keep this product competitive when compared to the current market; in order to do this, the production cost of the device must be low as well which is why the device will be sold as a kit to save on labor costs.

Size is another major factor that limits the components of the device. In order to keep it small, the microcontroller must be small as well as the enclosure and the components within the microcontroller.

Economic

The target customer base for this device is electrical engineering students and hobbyists of all ages. The main costs come from the microprocessor platform, the Arduino Leonardo. The original estimated cost of all the components was under $20. This was the goal set in the beginning of the project. However, upon further research, the Arduino Leonardo was determined to be the best choice as the platform for the project due to its built-in keyboard emulation and hardware safety features. The final component costs for the project is $34.37.

If the device is successful, there are several cost benefits that will occur over the lifetime of the product if it was commercialized. As more devices are sold and production increases, more components can be ordered in large bulk purchases, which translates to a lower overall cost. As more people use and experiment with the device, sales may increase or decrease over the lifespan of the product depending on how well it is supported and updated as problems arise.

If manufactured on a commercial basis

The amount of devices sold each year is currently unknown. Since the product mainly targets students, a large range of units could be sold.

The manufacturing cost of this device current amount to about $35. This includes purchasing all the parts and their shipping costs. Since most of the components will be bought in bulk, the prices will decrease as more components are purchased.

The sales price cost of this device will be between $50 and $60. This includes the price on purchasing and bagging the components. Shipping prices will be based on location. The main goal of this device is not profit but more of a learning experience and helping other students with their labs. If used correctly

and not recklessly, the device should not require any additional costs to maintain and operate.

Environmental

This device may use potentially environmentally harmful microelectronic components so the byproducts of manufacturing those components will have an impact on the environment.

Since this device uses a personal computer as a power supply, the byproduct of generating that power would be the emissions from the various electric generators including, but not limited to, natural gas, nuclear, coal, petroleum, and other renewable resources. Fortunately, the power drawn from the device is less than 1 watt, making a very small load on the electrical grid.

Assuming that there are defective parts within the device, the user can recycle the defective parts at electronic recycling centers, which helps the environment and keeps unwanted waste out of the ecosystem.

Manufacturability

There should be no special issues with manufacturing this device once a design is validated. It uses parts that can be bought online from anywhere. Since this device will not be sold prebuilt, there will not be any manufacturing issues from the seller as well.

The components used for this device are low in cost and very common. Purchasing in bulk saves on the cost of shipping and price per part. This allows us to keep the price of the device lower to keep it more competitive.

Sustainability

Most, if not, all of the components of the device are recyclable, so in the case where the device is faulty or the components or faulty, the device itself or the components can be recycled at an electronics recycling center or re-purposed for other projects.

Ethical

The device will have to perform to the claims that are written by us regarding the ability to measure up to certain voltages and currents. In addition, the accuracy of the device must be listed correctly for each measurement range. The product will be safe to use under the specified environments without any reasonable risk of harming the user. A warning sheet will be included with the product to specify any hazards or dangerous operating conditions.

Health and Safety

An important safety issue that needs to be addressed is a fail-safe feature which protects the user from misuse in the case of measuring high voltages or currents. These dangers can be avoided by using the protection circuitry and limiting exposed high-voltage contacts. The device is intended for use by people over the age of 10 and is not intended as a toy. Appropriate markings will be added to the device's packaging to reflect this.

Social and Political

The direct stakeholders are the students and hobbyists that purchase this device, our group, and the manufacturers that make the components of this device. Indirect stakeholders include the companies the hobbyists work for that make the products and the teachers who the students learn from.

Development

This project required domain knowledge from many different areas of electrical engineering. Developing this program on time with the required functions called for a team with a wide range of skill sets. This program improved our knowledge of data collection tools and the software associated with USB keyboard interfaces.

B. Cost Tables

Original Cost Estimate:

Cost per Item:		
Item Name	**Quantity**	**Total Price**
400 Pin Breadboard	1	$3.25
Misc. Resistors		$0.50
Quad Op-Amp IC	2	$0.65
PIC 18F 2550 Mircocontroler	1	$3.48
Breadboard Jumpers	1	$2.53
5 Amp Hall Effect Current Sensor	1	$4.29
Plastic Angle for Case (3" Section)	1	$0.50
Banana Terminal Red/Black	8 Pair	$3.20
Mirs. Fasteners and Wire		$1.50
Mirco Push Button Switch	1	$0.50
Temperature Sensor IC Package	1	$0.99
Mini USB Socket Terminal	1	$0.19
Total:		**$21.58**

Final Spending Table:

Detailed Spending Table					
Item Name	**Unit Price**	**Quantity**	**Total Price**	**Purchase Link**	**Aviable Locally?**
400 Pin Prototyping Board	0.29	2	$0.58	http://goo.gl/gTBrH	Yes, Radio Shack
39k Resistors	0.05	3	$0.15	http://goo.gl/Lnf5A	Yes, Radio Shack
Rail to Rail Quad Op-Amp IC	2.25	1	$2.25	http://goo.gl/jcoSg	No, Use Digikey
4 Input DIP Switches	0.56	1	$0.56	http://goo.gl/9YKcG	No, Use Digikey
Banana Jack Terminals	0.25	5	$1.25	http://goo.gl/NXsLS	Yes, Radio Shack
5 Amp Hall Effect Current Sensor	3.55	1	$3.55	http://goo.gl/4xqhz	No, Use eBay
14 Pin IC Socket	0.15	1	$0.15	http://goo.gl/U4E84	Yes, Radio Shack
22 AWG Assorted Wire Solid Core & Solder	0.50	1	$0.50	http://goo.gl/Yzk9A	Yes, Radio Shack
10K Ohm Blue Trim Pot (15 Turn)	0.20	3	$0.60	http://goo.gl/aizsS	Yes, Radio Shack
Log Speed Trim Pot w/Knob	1.99	1	$1.99	http://goo.gl/0Uhqw	Yes, Radio Shack
16 IC Breakoff Male Headers	0.02	16	$0.32	http://goo.gl/pgqjl	No, Use Digikey
USB Mirco Cable	1.56	1	$1.56	http://goo.gl/OLRwP	Yes, Radio Shack
On/Off Recording Switch	0.92	1	$0.92	http://goo.gl/ptX4T	Yes, Radio Shack
Arudino Leonardo	19.99	1	$19.99	http://goo.gl/OfJrn	No, Use Amazon
Total Purchase Price:			**$34.37**		

C. Arduino Sketch Code

```
/*
  USB Data Logger
  Students: Mattics Phi, Scott Leonard, Greg Alesso, Norman Tran
  Advisor: Dale Dolan
  California Polytechnic State University
  Senior Project
  Spring 2013
*/

int i;
static int flag = 0;  // Flag to print out header
long result;  // Vcc in millivolts
unsigned int digRead;  // Digital value of voltage read from input
double voltage;  // Voltage read from analog input
double vcc;  // Vcc in volts
double voltVal;  // Voltage value to print
double curVal;  // Current value to print
double potVal;  // Digital potentiometer value
int potPin = 0;  // Analog pin for potentiometer
double logDelay;  // Logging speed delay in ms

void setup()
{
  Serial.begin(9600);          //  Set up serial
  // Input pullup resistors used instead of external pulldowns
  // HIGH = OFF, LOW = ON
  pinMode(9, INPUT_PULLUP);    // Switch for analog pin 2 (Voltage 1)
  pinMode(10, INPUT_PULLUP);   // Switch for analog pin 3 (Voltage 2)
  pinMode(11, INPUT_PULLUP);   // Switch for analog pin 4 (Voltage 3)
  pinMode(12, INPUT_PULLUP);   // Switch for analog pin 5 (Current)
  pinMode(13, INPUT_PULLUP);   // Switch for analog pin 0 (Potentiometer)
  Keyboard.begin();            // Set up keyboard entry
}

void loop()
{
  if (digitalRead(13) == LOW)  // Pin 13 must be grounded to start writing
  {
    if (flag == 0)  // Print out header at the top
    {
      flag = initializeHeader();
```

```
    }
    adjustDelay();  // Set logging speed by adjusting delay

    for (i = 5; i >= 2; i--)
    {
      if (digitalRead(i+7) == LOW)
      {
        // Log voltages from analog pins 2-4, current from analog pin 5
        if (i == 5)  // Current pin
        {
          voltVal = readVoltage(i);  // Read voltage
          curVal = (voltVal-2.56)*5.555;  // Scale value for current
          Keyboard.print(curVal);
        }
        else  // Voltage pin
        {
          voltVal = readVoltage(i)*5.1;  // Read and scale voltage
          Keyboard.print(voltVal);
        }
      }
      if (i == 2)  // Reached end of line
      {
        insertTab();
        // Print logging speed in ms (0 to 200)
        Keyboard.print(logDelay);
        insertEnter();
      }
      else
      {
        insertTab();
      }
    }
  }
}

int initializeHeader()
{
  Keyboard.print("Current");
  insertTab();
  Keyboard.print("Voltage 1");
  insertTab();
  Keyboard.print("Voltage 2");
  insertTab();
  Keyboard.print("Voltage 3");
  insertTab();
  Keyboard.print("Logging Delay (ms)");
  insertEnter();
```

```
  return 1;
}

void insertTab()
{
  Keyboard.write(KEY_TAB);
}

void insertEnter()
{
  Keyboard.write(KEY_RETURN);
}

long readVcc() {
  // Read 1.1V reference against AVcc
  // set the reference to Vcc and the measurement to the internal 1.1V
reference
  #if defined(__AVR_ATmega32U4__) || defined(__AVR_ATmega1280__) ||
defined(__AVR_ATmega2560__)
    ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  #elif defined (__AVR_ATtiny24__) || defined(__AVR_ATtiny44__) ||
defined(__AVR_ATtiny84__)
    ADMUX = _BV(MUX5) | _BV(MUX0);
  #elif defined (__AVR_ATtiny25__) || defined(__AVR_ATtiny45__) ||
defined(__AVR_ATtiny85__)
    ADMUX = _BV(MUX3) | _BV(MUX2);
  #else
    ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  #endif

  delay(2); // Wait for Vref to settle
  ADCSRA |= _BV(ADSC); // Start conversion
  while (bit_is_set(ADCSRA,ADSC)); // measuring

  uint8_t low  = ADCL; // must read ADCL first - it then locks ADCH
  uint8_t high = ADCH; // unlocks both

  result = (high<<8) | low;

  result = 1125300L / result; // Calculate Vcc (in mV); 1125300 =
1.1*1023*1000
  return result; // Vcc in millivolts
}

double readVoltage(int anPin)
{
  vcc = readVcc()/1000.0;  // Read Vcc
```

```
  digRead = analogRead(anPin);  // Read analog pin
  delay(10);
  voltage = ((digRead/1023.0)*vcc);  // Calculate voltage
  return voltage;
}

void adjustDelay()
{
  potVal = analogRead(potPin);  // Read the potentiometer value
  // Potentiometer reading should be between 0 and 1023
  logDelay = potVal/5.115;  // Gives a delay between 0 and 200 ms
  delay(logDelay);
}
```

D. Literature Search

[1] National Instruments, "Low-Cost, Bus-Powered Multifunction DAQ for USB" 218 Datasheet [Revised Sept. 2012]

The source has authority to provide reliable information because it is from a well-known company named National Instruments.

[2] Wim Claes et al., *Design of Wireless Autonomous Datalogger IC's (The Springer International Series in Engineering and Computer Science*, 1st ed. New York: Springer-Verlag, 2005.

This source has authority because the authors have written other books on low powered dataloggers, transistor design, and many other books about biomedical systems and theory.

[3] Engelberg, S. et al., "Instrumentation notes – A USB-Enabled, FLASH-Disk-Based Data Logger," *Instrumentation & Measurement Magazine, IEEE*, vol. 10, iss. 2, p. 63 – 66, April 2007.

Shlomo Engelberg has also written 8 books on electrical and computer engineering including digital signal processing, microcontroller design, and control theory.

[4] Gabriel Ibarra, et al., "Data Storage Device Compatible with Multiple Interconnect Standards," U.S. Patent 8 239 581, May 10, 2010.

Gabriel Ibarra and his team at Seagate Technology LLC filed another patent that dealt with data storage devices methods for power-on initialization.

[5] Gabriel Ibarra, et al., "Data Storage Device and Methods for Power-on Initialization," U.S. Patent 8 055 942, November 8, 2011.

Gabriel Ibarra and his team at Seagate Technology LLC filed another another patent that dealth with data storage device compatible with multiple interconnect standards.

[6] Extech Instruments, "Humidity and Temperature USB Datalogger" RTH10 Datasheet [Revised 2009]

The source has authority to provide reliable information because it is from a technological company named Extech.

[7] Gemini Data Loggers, "Tinytag Ultra 2 Temperature Logger" TGU-4017 Datasheet [Revised April 2011]

Gemini Data Loggers is a company that focuses solely on datalogging devices – because of this, we decided that if we were to look at data loggers, it might as well come from Gemini.

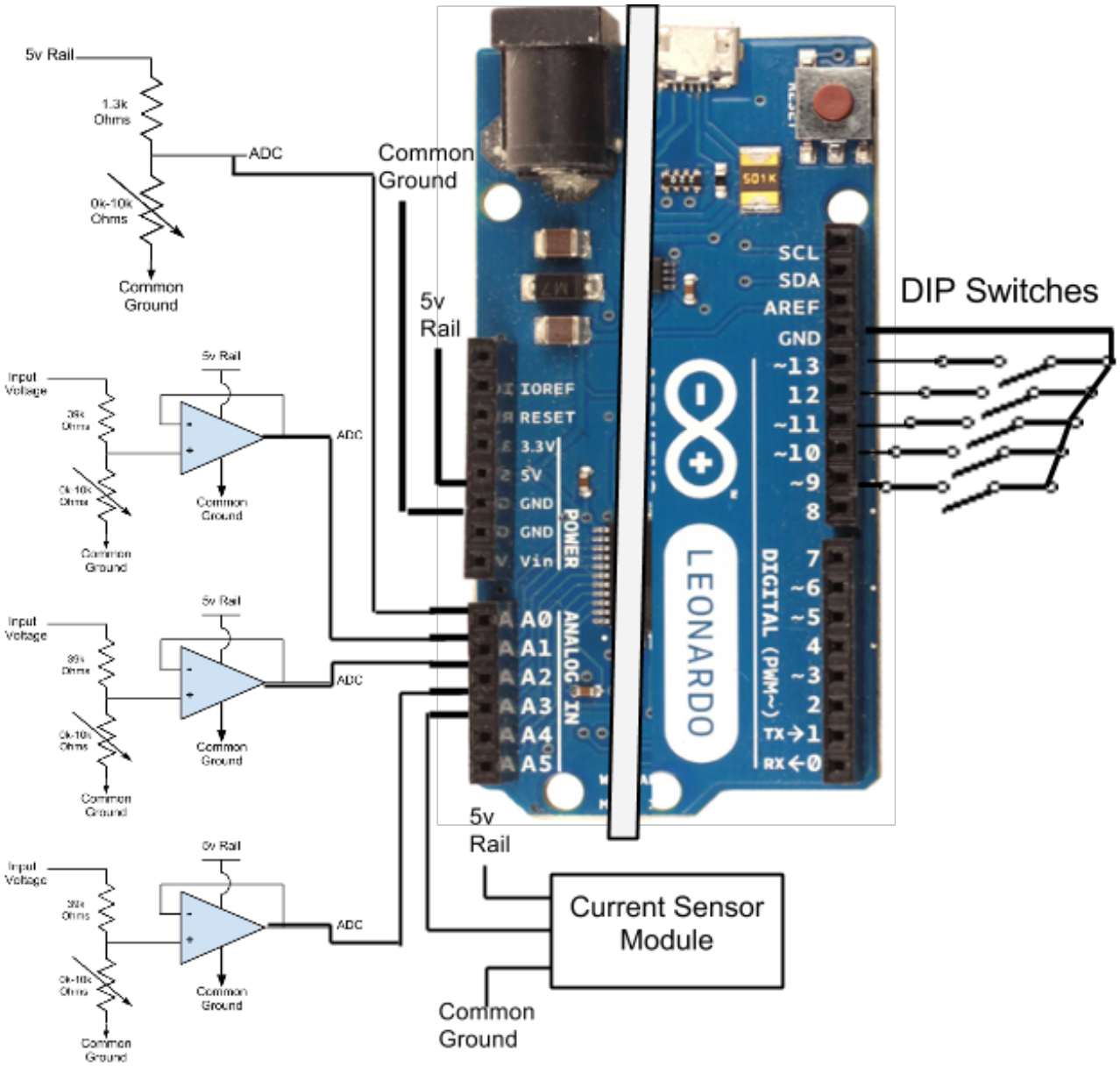
[8] Agilent Technologies, "InfiniiVision 7000A Series Oscilloscopes" Datasheet [Revised Feb. 2010]

Agilent Technologies is a well-known company that creates measurement tools and equipment. They sell equipment worldwide and are well known within our labs as well.

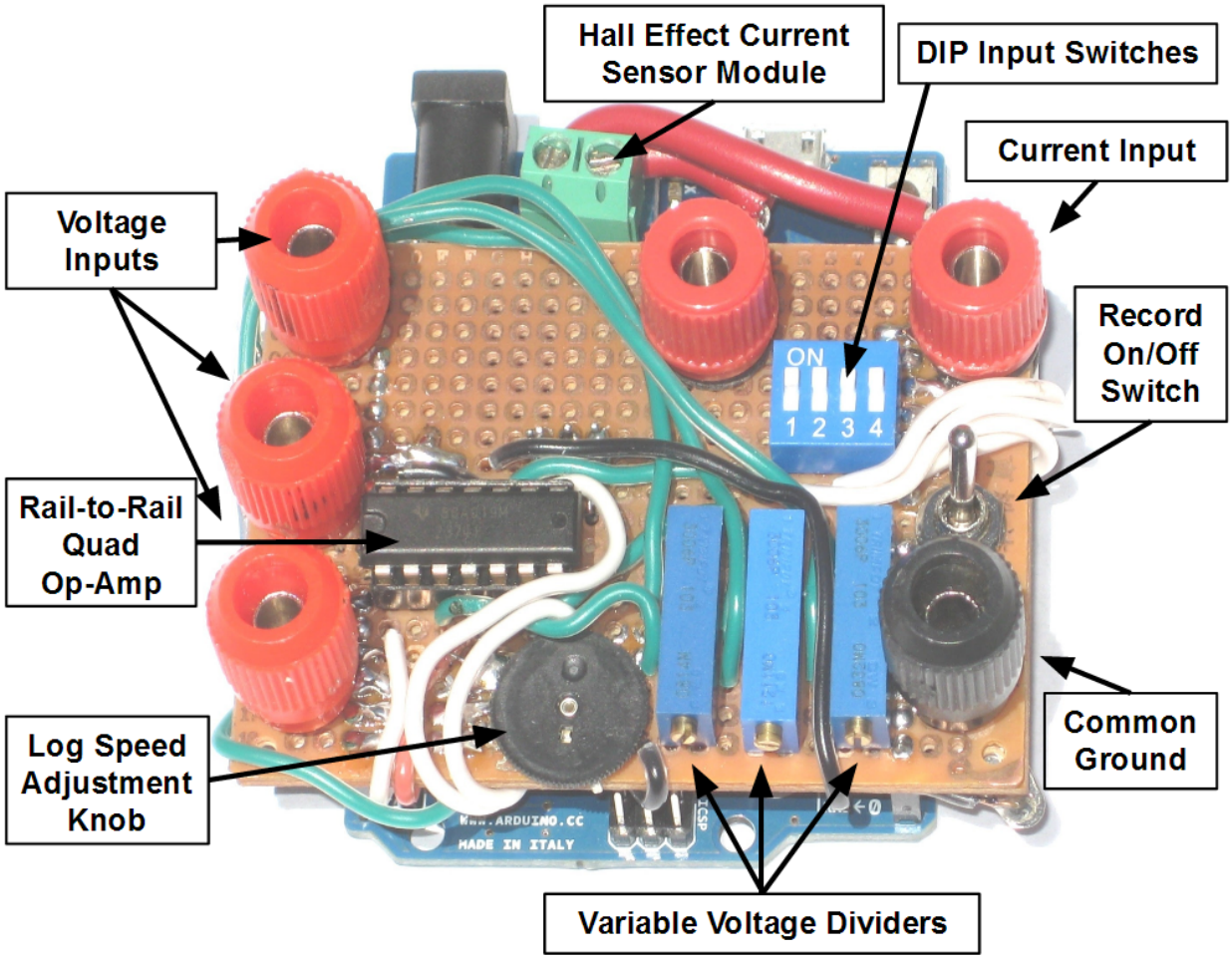
[9] Thurlby Thandar Instruments, "Low-cost 20 MHz Function Generator" TG120 Datasheet

Thurlby Thander is a European company but nevertheless, the brand is well known on that side of the world.

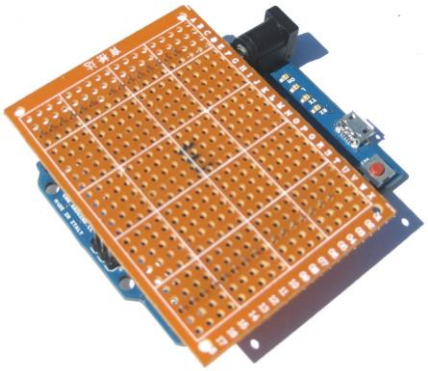
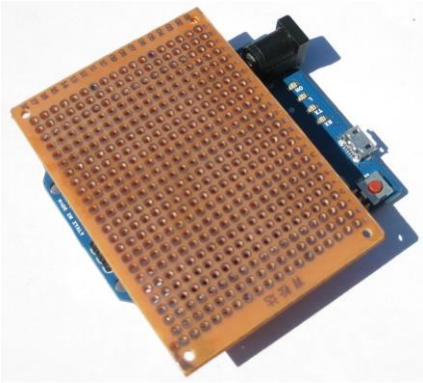
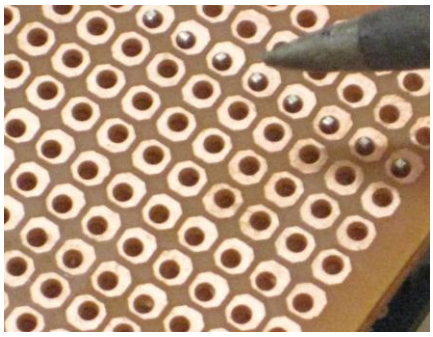
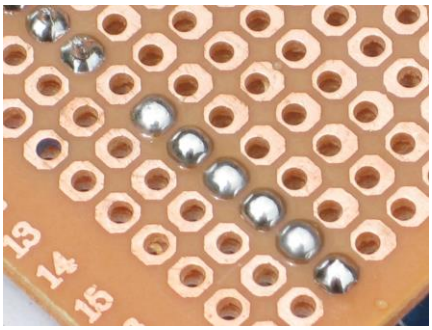
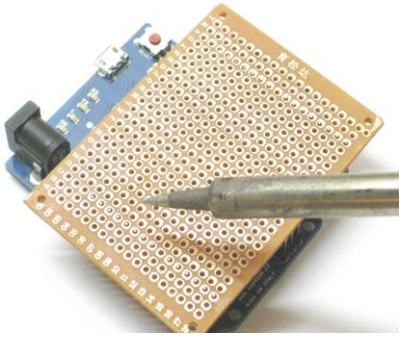
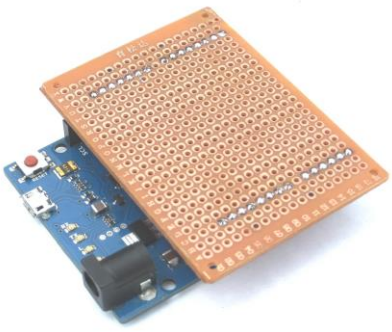
E. Master Wiring Diagram

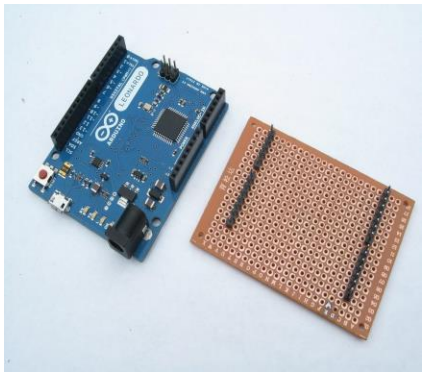
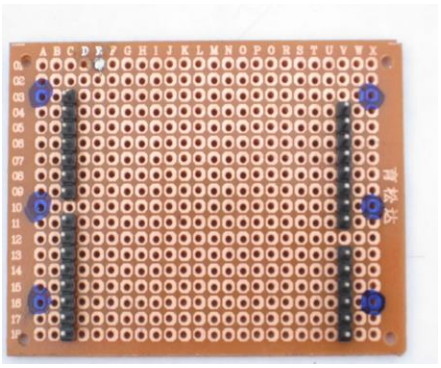
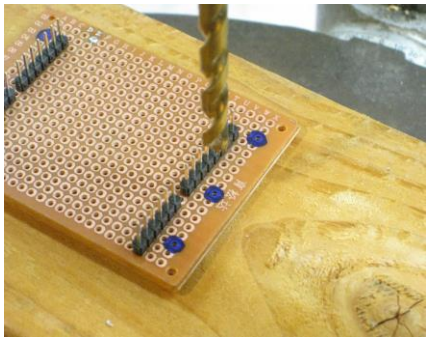
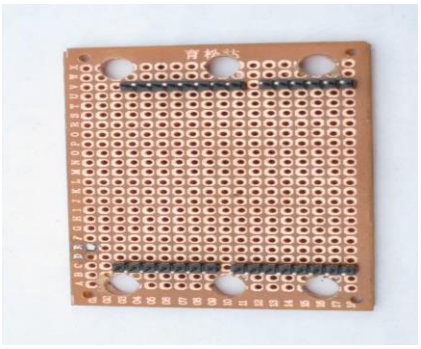
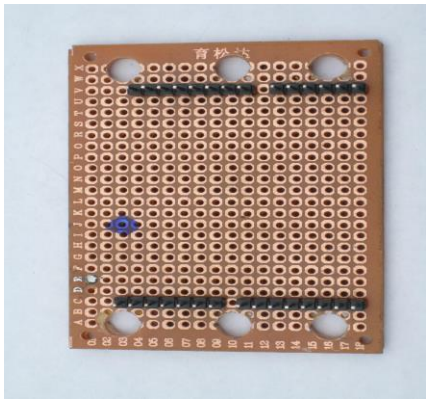
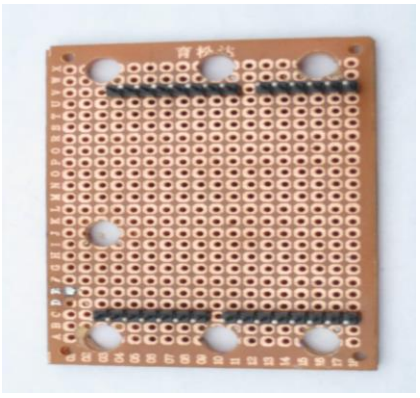
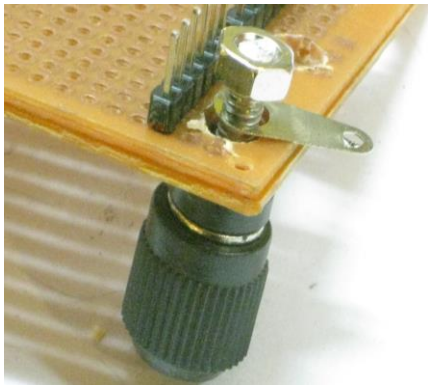
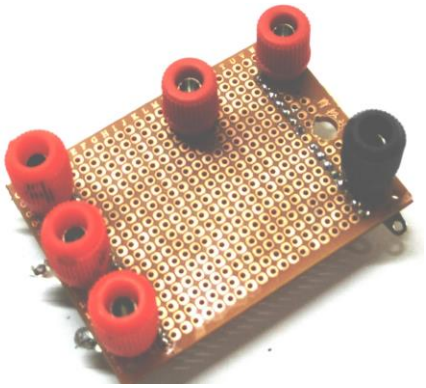


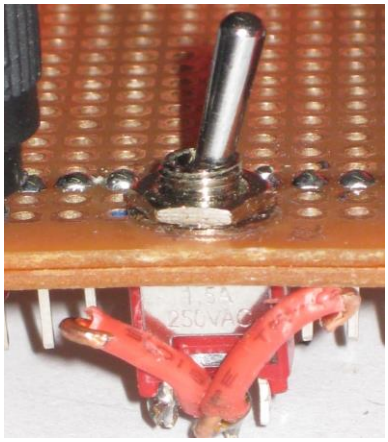
F. Hardware Layout

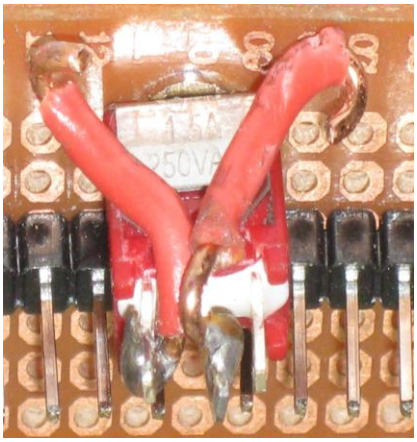
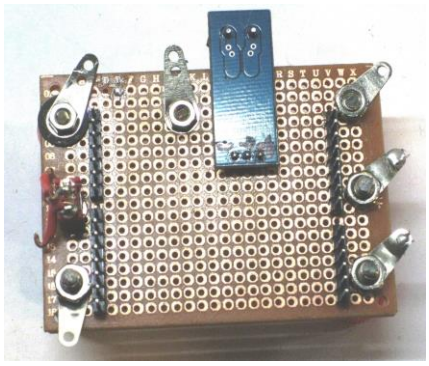


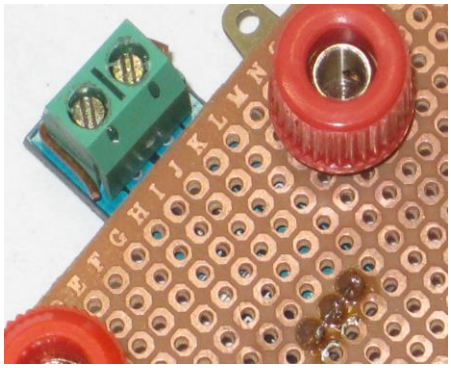
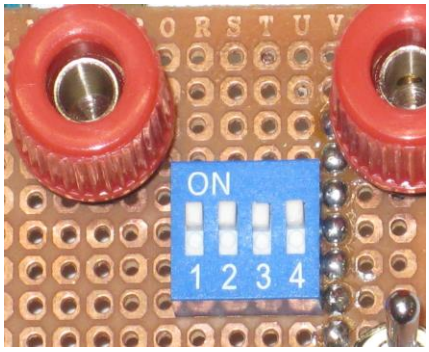
Appendix G: Construction Guide

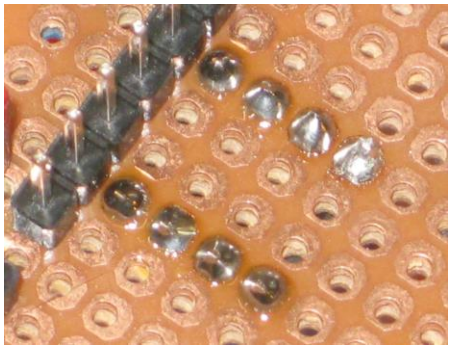
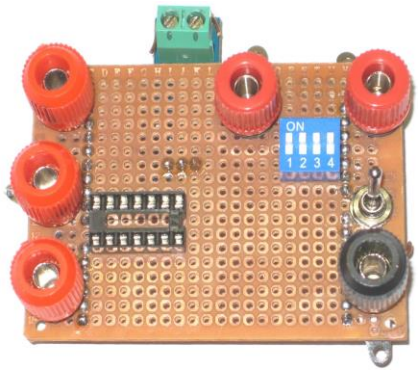
	**1. Insert Headers** Break off four groups of male headers to form the connection between the board and the shield. All pins receive a header except for SCL, SDA, and AREF.	
	**2. Align Proto-Board** The shield is made from two proto-boards with copper pads facing out from both sides. Stack two boards over the heads so that they are centered over the Arduino.	
	**3. Solder Headers** Solder the headers to the proto-boards while they are aligned over the Arduino.	
	**4. Check Board** Using a multimeter, check each of the solder points for unwanted shorts to its neighboring pads. Repeat this process for each new component added.	

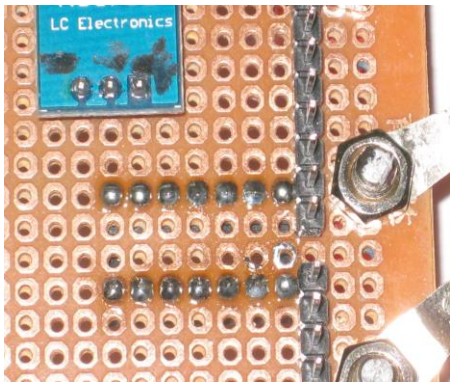
	**5. Mark for Drilling** Six holes need to be added for the voltage inputs and recording on/off switch. Use the outermost pre-drilled holes as a center for three equally spaced marks on each side of the the board	
	**6. Drill Terminal Holes** Using a 1/4” drill bit, drill out holes for the bannana terminals.	
	**7. Optional Terminal** If you are including the current sensor, drill an extra hole for an additional terminal.	
	**8. Mount Terminals** Mount the banana terminals to the board, making sure that each post is insulated from the board by a plastic spacer.	

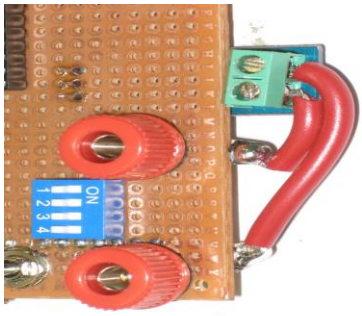
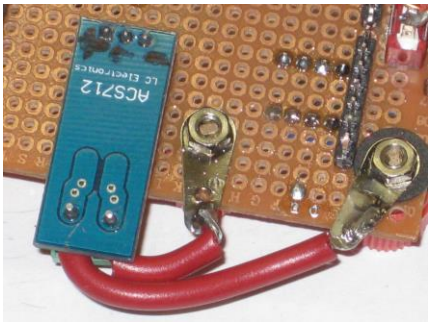
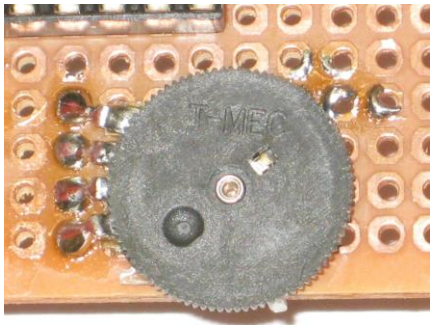
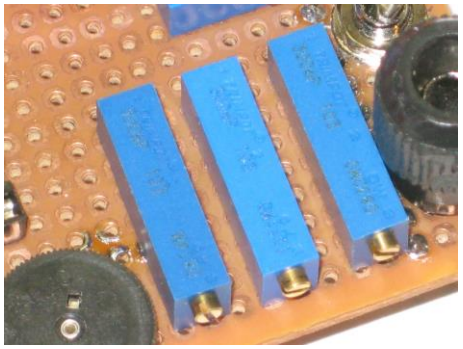
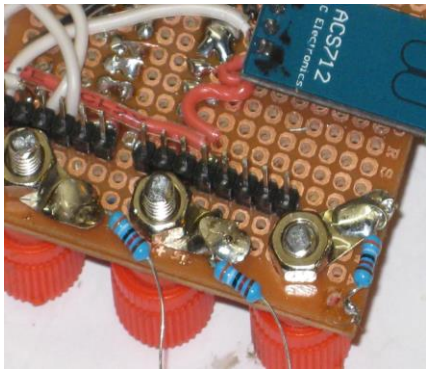
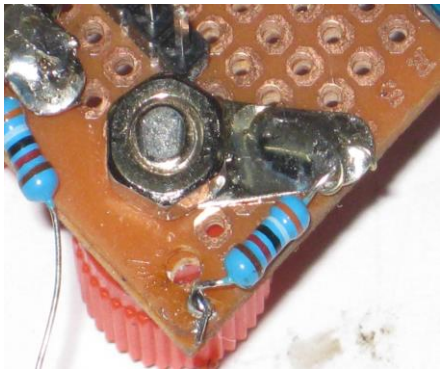
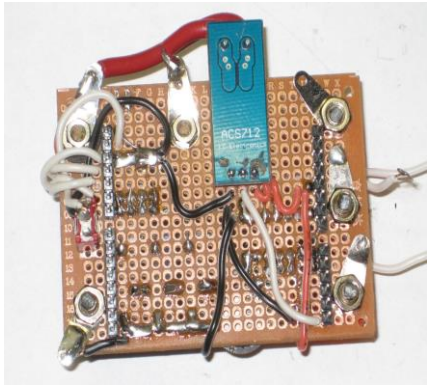
<table>
<tr>
<td>  </td>
<td>9. Mount On/Off Switch

In the remaining hole, mount the on/off recording switch. The switch pictured here is pre-wired.</td>
<td>  </td>
</tr>
<tr>
<td>  </td>
<td>10. Mount Current Sensor Module

Mount the current sensor on the bottom of the board as shown. Test fit the shield to the board with the sensor to avoid collisions before soldering.</td>
<td>  </td>
</tr>
<tr>
<td>  </td>
<td>11. Mount DIP Switches

Mount and solder the DIP switches to the board. Pins 12 through 9 will be connected to ground through the switch.</td>
<td>  </td>
</tr>
<tr>
<td>  </td>
<td>12. Mount IC Socket

Mount and solder the IC socket to the board. To form the voltage followers, solder the negative input and output of each op-amp together for each of the four stages.</td>
<td>  </td>
</tr>
</table>

	**13. Connect Current Sensor to Terminals** Connect the current sensor to the current terminals.	
	**14. Mount Potentiometers** Mount and solder the log speed adjustment potentiometer, along with the voltage divider pots.	
	**15. Add Resistors to Voltage Inputs** Solder resistors directly to the voltage input terminals. This fixed resistor forms the top half of each voltage divider.	
	**16. Connect Components** Using segments of 22 AWG wire, connect each of the components as shown in the complete wiring diagram.