ECE 341: Junior Design

Power Supply Testing Report

Kyle Barton
Lab Partner: Samuel Barton

Fall 2020
Instructor: Matt Shuman
Grading TA: Shane Witsell

1 Introduction

Throughout the ECE 341 Junior Design I class, a wide array of topics have been explored through the JD Power Supply board. Beginning with understanding and analyzing the schematic, then soldering the surface mount components, using the board, and now to extensive testing, the JD Power Supply has been an effective way to bring learned concepts to hands on work.

This report details in-depth testing of the behavior and limitations of the JD Power Supply system. Input and output power tests, thermal behavior tests, and other analysis is done for both the 3.3V and the 5V circuits on the board.

2 Background Information

Four weeks of labs have been performed with activities surrounding the JD Power Supply as well as its use in separate projects. The first lab activity focused on understanding the schematics of both the 3.3V and the 5V circuits and gathering background information on the board. The datasheets of the components were explored to calculate maximum power rating and the processes for some simple measurements were discovered. To show sufficient understanding, block diagrams were created for the system.

The second lab activity went more in depth exploring some of the important circuit components, mainly the LT1117 voltage regulator in the 3.3V circuit. This activity familiarized LTSpice simulation which was used to simulate and test different blocks in the circuit. Overall a better understanding of the functionality of the 3.3V linear power supply circuit was gained.

From here the JD Power Supply system was assembled on the PCB. This involved using the schematic to put the correct components in the correct orientation and then solder them. This was a good insight into surface mount soldering as the power supply used components as small as 0603.

The next lab activity focused on the characteristics of the 3.3V circuit under a load. The test load used was a transistor controlled by a potentiometer. This lab activity also involved taking in and analyzing data with MATLAB, which proved to be a very useful and important concept. This lab looked into how the output current of the power supply behaved and the fact that it was not constant.

The last lab activity was an in depth exercise that focused on a similar programmable load using the Arduino Nano microcontroller. Although this activity was primarily focused on the Arduino, an important part of it was analyzing the thermal behavior of the power supply. This idea was carried over and a similar thermal analysis was performed for this report.

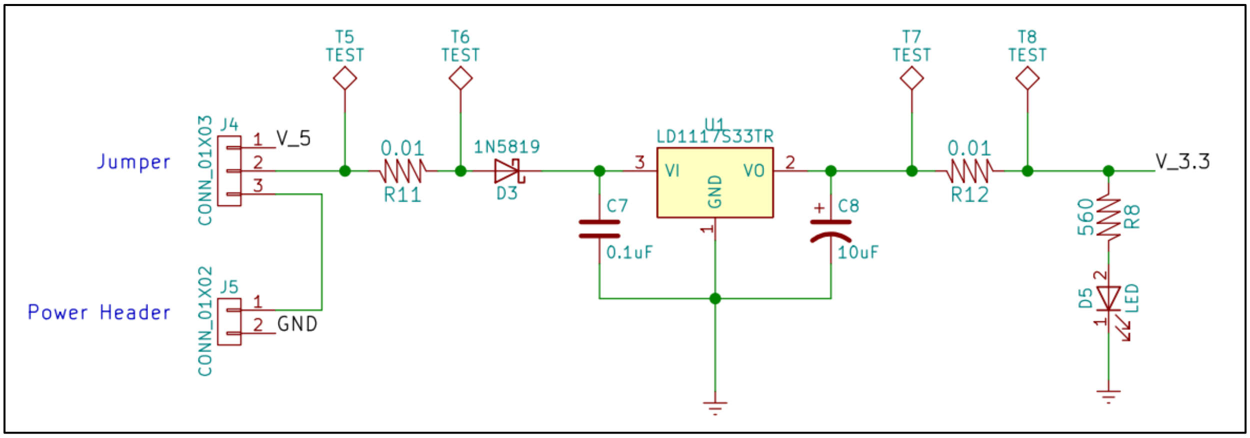


Figure 1: 3.3V JD Power Supply Schematic

Figure 1 above shows the schematic for the 3.3V power supply circuit. Note that this is a linear power supply and the input comes from the 5V supply on the same PCB. Figure 2 below shows the schematic for the 5V power supply circuit. This is a switching power supply that takes a 12V wall adapter to power it.

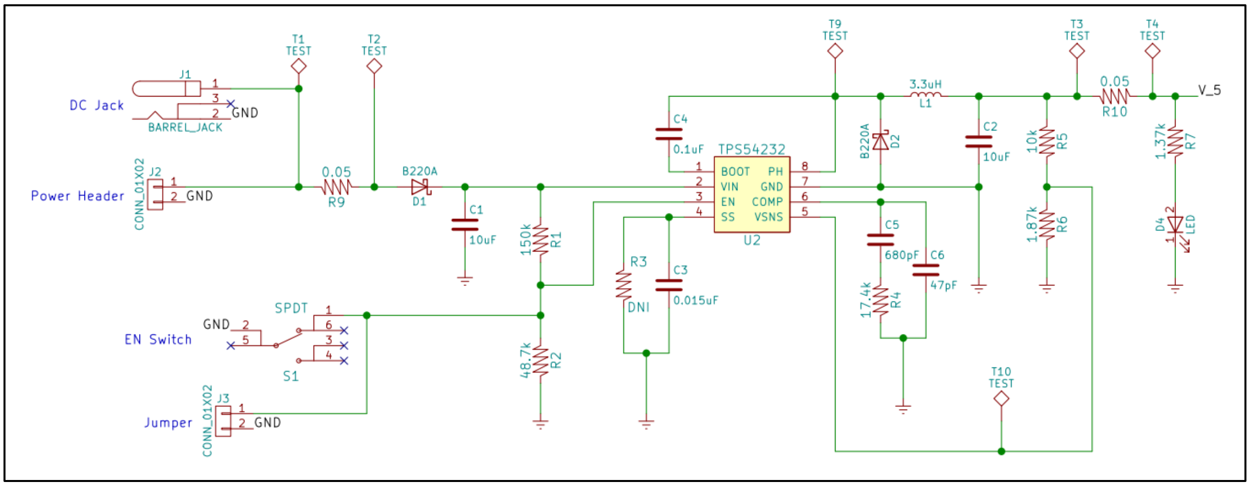


Figure 2: 5V JD Power Supply Schematic

Figure 3 below is the JD Power Supply full bill of materials. Note that the 'designator' corresponds to the labels in figures 1 and 2.

Designator	Package	Quantity	Designation
D5,D4	LED_0603_HandSoldering	2	LED
L1	L_Sumida_4.9x4.9mm_diag	1	3.3uH
U2	SOIC-8-N	1	TPS54232
C8	CP_Elec_D4.0mm	1	10uF
J2,J3,J5,J6,J8	Pin_Header_Straight_1x02_Pitch2.54mm	5	CONN_01X02
J4	Pin_Header_Straight_1x03_Pitch2.54mm	1	CONN_01X03
S1	SPDT_MHSS1104	1	SPDT
T1,T2,T3,T4,T5,T6,T7,T8,T9,T10,T11	TESTPIN	11	TEST
U1	SOT-223	1	LD1117S33TR
D2,D1	D_SMA	2	B220A
D3	D_SOD-123	1	1N5819
C1,C2	C_1206_HandSoldering	2	10uF
J7	JST_PH_B3B-PH-K_03x2.00mm_Straight	1	CONN_01X03
J1	BARREL_JACK	1	BARREL_JACK
C3	C_0805_and_0603_HandSoldering	1	0.015uF
C4,C7	C_0805_and_0603_HandSoldering	2	0.1uF
C5	C_0805_and_0603_HandSoldering	1	680pF
C6	C_0805_and_0603_HandSoldering	1	47pF
R1	C_0805_and_0603_HandSoldering	1	150k
R2	C_0805_and_0603_HandSoldering	1	48.7k
R3	C_0805_and_0603_HandSoldering	1	Rss
R4	C_0805_and_0603_HandSoldering	1	17.4k
R5	C_0805_and_0603_HandSoldering	1	10k
R6	C_0805_and_0603_HandSoldering	1	1.87k
R7	C_0805_and_0603_HandSoldering	1	1.37k
R8	C_0805_and_0603_HandSoldering	1	560
R9,R10	C_0805_and_0603_HandSoldering	2	0.05
R11,R12	C_0805_and_0603_HandSoldering	2	0.01

Figure 3: JD Power Supply Bill of Materials

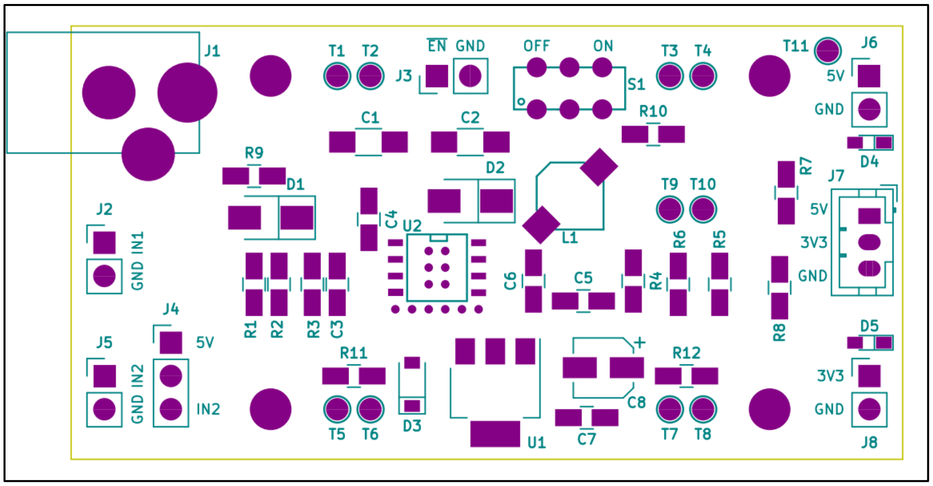


Figure 4: JD Power Supply PCB Layout

3 3.3V Linear Power Supply

A linear power supply uses transformers (or in our case a voltage regulator) to provide a DC signal. Linear power supplies are less power efficient as heat is dissipated in the voltage regulation process, but they provide a much cleaner DC signal which can be very useful. The JD Power Supply board has a 3.3V linear power supply circuit that takes in the 5V signal from the switching power supply as an input.

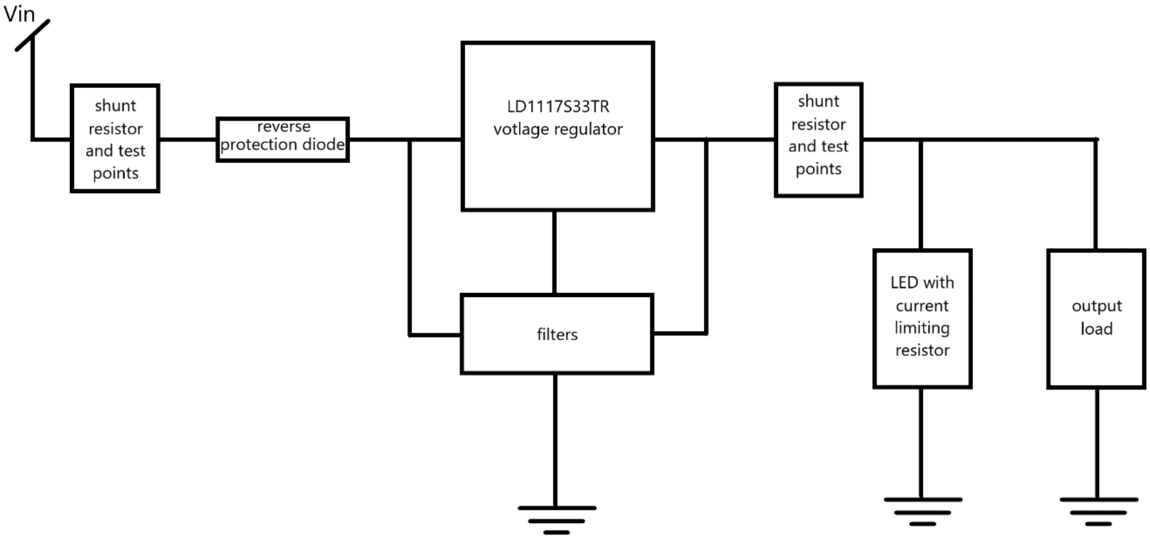


Figure 5: 3.3V Linear Power Supply Block Diagram

As seen in figure 5 above, the 3.3V linear power supply is a relatively simple circuit with the main component being the LD1117 voltage regulator. This component takes in a noisy 5V signal and provides a steady and clean 3.3V signal to the output. Some additional components are added such as shunt resistors for testing the circuit, filters, one way protection diodes, and a power indicating LED.

Shunt Resistor Measurements

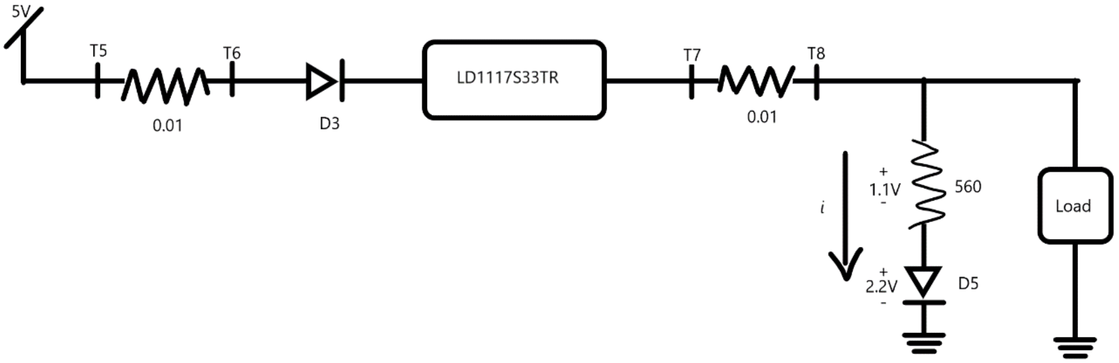


Figure 6: 3.3V Simplified Shunt Resistor Measurement Circuit

Using the simplified 3.3V linear power supply circuit above, the voltages at the test points T5-T8 were calculated. Looking at the datasheet for the D5 diode it was found that it has a foreward voltage of 2.2V, and knowing that the output of the LD1117 voltage regulator would always be 3.3V it could then be determined that the voltage drop over the 560Ohm current limiting resistor would always be 1.1V. From this it was determined that *i* in figure 6 above would always be 1.96mA. Thus the total output current for the LD1117 would be the load current and this constant 1.96mA added together.

From the datasheet of the LD1117 voltage regulator, it was found that the recommended operating current was 800mA. To calculate the shunt resistor measurements for 5 test loads from 0mA to the recommended 800mA, the drop over each component was calculated in figure 7 below.

Load current (mA)	Total current (mA)	T5 (V)	Drop over R11 (V)	T6 (V)	Drop over D3 (V)	T7 (V)	Drop over R12 (V)	T8 (V)
0	1.96	5	0.0000196	4.9999804	0.1175	3.3	0.0000196	3.2999804
160	161.96	5	0.0016196	4.9983804	0.225	3.3	0.0016196	3.2983804
320	321.96	5	0.0032196	4.9967804	0.345	3.3	0.0032196	3.2967804
480	481.96	5	0.0048196	4.9951804	0.375	3.3	0.0048196	3.2951804
640	641.96	5	0.0064196	4.9935804	0.425	3.3	0.0064196	3.2935804
800	801.96	5	0.0080196	4.9919804	0.45	3.3	0.0080196	3.2919804

Figure 7: Shunt Resistor Calculations at Test Load Currents

Load current (mA)	T5 (V)	T6 (V)	T7 (V)	T8 (V)
0	5	4.9999804	3.3	3.2999804
160	5	4.9983804	3.3	3.2983804
320	5	4.9967804	3.3	3.2967804
480	5	4.9951804	3.3	3.2951804
640	5	4.9935804	3.3	3.2935804
800	5	4.9919804	3.3	3.2919804

Figure 8: Calculated Shunt Resistor Voltage Values

As seen in figure 7, the variations in the values of the shunt resistors are very small due mostly to the very small voltage drop over the 0.01Ohm resistors. For this reason the Arduino Nano's ADC itself could not pick up the small differences because the ADC only has a resolution of 1024 from 0V to 5V. Thus the ADC values for T5 and T6 would be the same and the ADC values for T7 and T8 would also be the same. This can be seen in figures 9 and 10 below which calculate the predicted ADC values if the pins were plugged directly into the Arduino Nano.

Load current (mA)	T5 (V)	T5 ADC Value	T6 (V)	T6 ADC Value
0	5	1024	4.9999804	1024
160	5	1024	4.9983804	1024
320	5	1024	4.9967804	1024
480	5	1024	4.9951804	1024
640	5	1024	4.9935804	1024
800	5	1024	4.9919804	1024

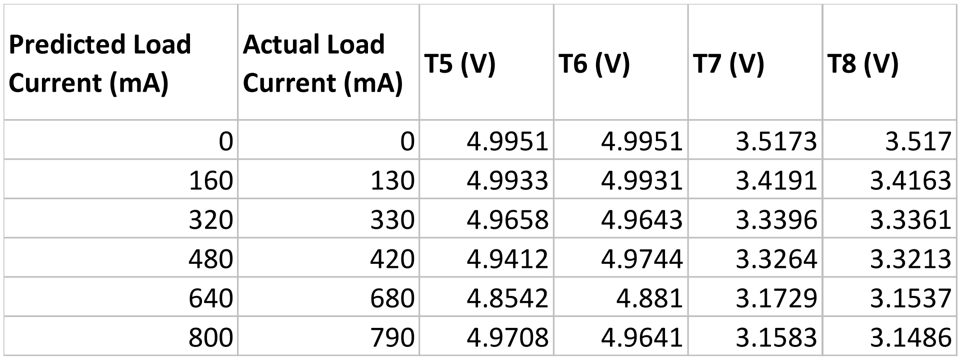
Figure 9: T5 and T6 Predicted ADC Values

Load current (mA)	T7 (V)	T7 ADC Value	T8 (V)	T8 ADC Value
0	3.3	676	3.2999804	676
160	3.3	676	3.2983804	676
320	3.3	676	3.2967804	676
480	3.3	676	3.2951804	676
640	3.3	676	3.2935804	676
800	3.3	676	3.2919804	676

Figure 10: T7 and T8 Predicted ADC Values

Thus it is clear that extra measurement tools are necessary to accurately test and detect the changes. This is where the INA169 Current Sensors come into play. With a gain of 100, the current sensor multiplies the drop over the shunt resistor by 100 to make it within the ADC resolution and therefore more precise. For example a drop of 0.02V would show up as 2V to the ADC with the current sensor and a drop of 0.01V would show up as 1V to the ADC with the current sensor. Both of these values are far apart within the 5V ADC range so they will have drastically different ADC values. If not current sensor was used, 0.002V and 0.001V would turn out to be the same ADC value.

Input and Output Power



Predicted Load Current (mA)	Actual Load Current (mA)	T5 (V)	T6 (V)	T7 (V)	T8 (V)
0	0	4.9951	4.9951	3.5173	3.517
160	130	4.9933	4.9931	3.4191	3.4163
320	330	4.9658	4.9643	3.3396	3.3361
480	420	4.9412	4.9744	3.3264	3.3213
640	680	4.8542	4.881	3.1729	3.1537
800	790	4.9708	4.9641	3.1583	3.1486

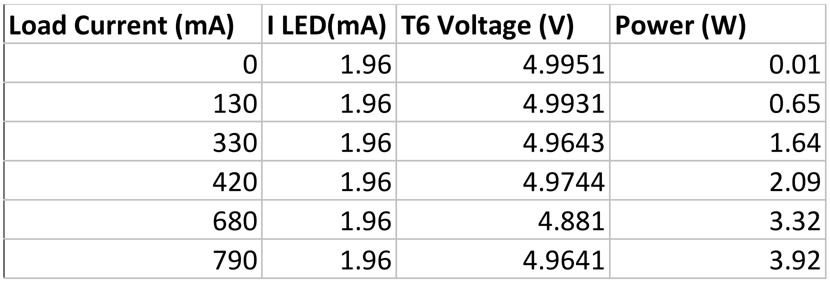
Figure 11: Measured Test Point Voltage Values

A 100k Ohm potentiometer was attached to the output of the 3.3V linear power supply as a simulated load and the voltages at each test point was calculated with the Arduino. The precision of the 100k pot was limited hence the ‘actual load current’ vs the ‘predicted load current.’ The load current was set by monitoring the potentiometer with the DMM and each test pin T5-T8 was connected to a corresponding Arduino analog input pin. The Arduino took in voltage data from the pins for 10 seconds at each test current and calculated the average to obtain the data in figure 11 above.

One thing to note in figure 11 is that due to the process of collecting the voltage data, the switching power supply input caused the voltage at T5 to sometimes be less than the voltage at T6. This error is small enough to be negligible and to avoid this in future calculations, a single data point from both T5 and T6 should be collected at the exact same time and those values used instead of the average. The available equipment limited this particular measurement to the method of using the average.

Using P = IV, the load current plus the correct through the resistor and LED, and the voltages at test point T6, the input power of the LD1117 voltage regulator could be calculated. These calculations are seen in figure 12 below.

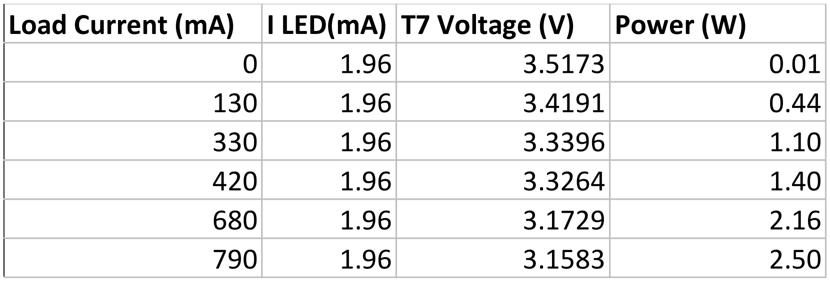
The power calculations are precise to two decimal places as the DMM used measured current to only two decimal places.



Load Current (mA)	I LED(mA)	T6 Voltage (V)	Power (W)
0	1.96	4.9951	0.01
130	1.96	4.9931	0.65
330	1.96	4.9643	1.64
420	1.96	4.9744	2.09
680	1.96	4.881	3.32
790	1.96	4.9641	3.92

Figure 12: 3.3V Input Power Calculations

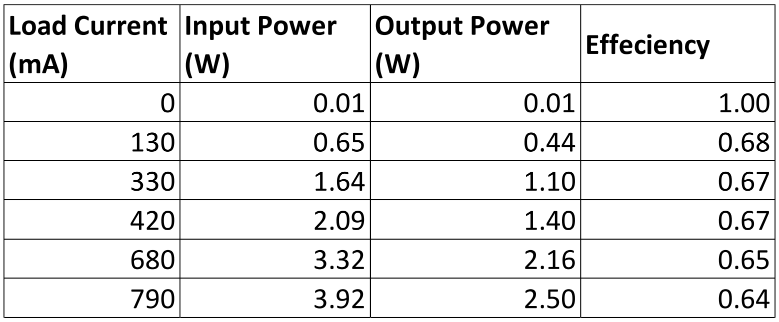
The same process was used for the LD1117 output power using the test point T7. This is seen in figure 13 below.



Load Current (mA)	I LED(mA)	T7 Voltage (V)	Power (W)
0	1.96	3.5173	0.01
130	1.96	3.4191	0.44
330	1.96	3.3396	1.10
420	1.96	3.3264	1.40
680	1.96	3.1729	2.16
790	1.96	3.1583	2.50

Figure 13: 3.3V Output Power Calculation

The efficiency of the 3.3V power supply can be calculated by output power / input power. Table 14 below shows the calculated efficiency at each tested load current.



Load Current (mA)	Input Power (W)	Output Power (W)	Effeciency
0	0.01	0.01	1.00
130	0.65	0.44	0.68
330	1.64	1.10	0.67
420	2.09	1.40	0.67
680	3.32	2.16	0.65
790	3.92	2.50	0.64

Figure 14: 3.3V Power Supply Efficiency

Negating the first calculation at 0mA of load current, the efficiency of the power supply seems to be around 65%. This number appears to decrease with higher load current which makes logical sense as more current is going through the LD1117 voltage regulator so it is dissipating more heat, therefor being less efficient. Overall an efficiency of 65% is fairly inefficient which is a known drawback to linear power supplies.

If a higher voltage than 5V was used to supply the linear 3.3V power supply, the efficiency would further decrease. The efficiency can be roughly calculated by the (output voltage) / (input voltage) so using a 9V supply would cut the efficiency down to 36%. This is because the voltage regulator takes the 9V signal and cuts it down to a steady 3.3V signal.

Thermal Dissipation

For the thermal dissipation calculations for both the 3.3V and the 5V power supplies, the following methods were used. Figure 15 below shows the voltage divider thermistor circuit used. The thermistor resistance value changes with temperature, therefor changing the voltage divider. With a constant source voltage the change in output voltage can be used to calculate the resistance value of the thermistor which can then be related to temperature via specs on the datasheet.

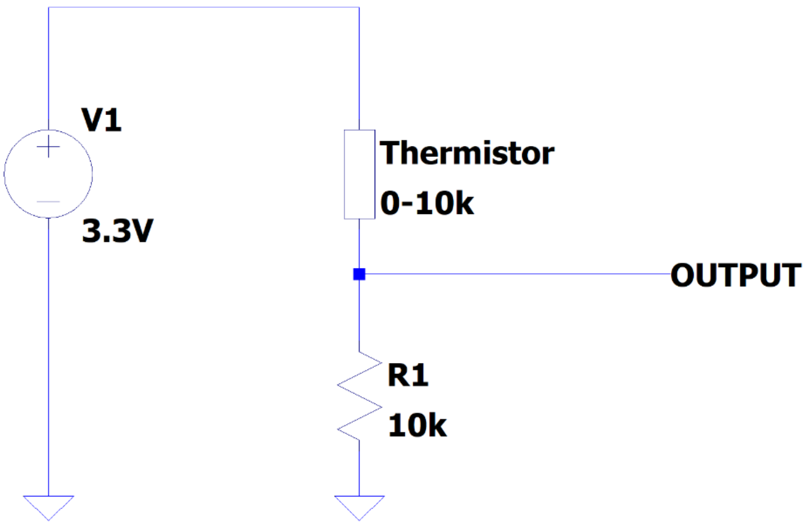


Figure 15: Thermistor Temperature Sensing Circuit

The voltage divider equation to calculate the OUTPUT:

$$OUTPUT = (3.3V)\frac{10k}{Rt + 10k}$$

Which can be solved to obtain the following equation for the thermistor resistance Rt:

$$Rt = \frac{(10k)(3.3V)}{OUTPUT} - (10k)$$

Looking at the thermistor datasheet, the following equation was obtained for the relationship between the resistance and the temperature by creating a linear equation between two datapoints:

$$Rt = 16663.7 - 266.548(T)$$

Where T is the temperature. Using equations in MATLAB, the temperature can be calculated as the OUTPUT voltage is recorded.

Note that this is a very rough estimate of the temperature and the accuracy is very limited. The equation between resistance and temperature for the thermistor is not linear in true practice and if the current through the voltage divider changes (does not in the 3.3V section) then the output is even less accurate. For these reasons, the thermal data is more a representation of the thermal trends of the power supply rather than the accurate thermal temperatures.

The built thermometer was verified at room temperature (no change in heat). The output yielded a relatively steady 24-25^{o}C which was approximately the room temperature at the time of the test. The temperature vs time graph can be seen in Figure 16 below.

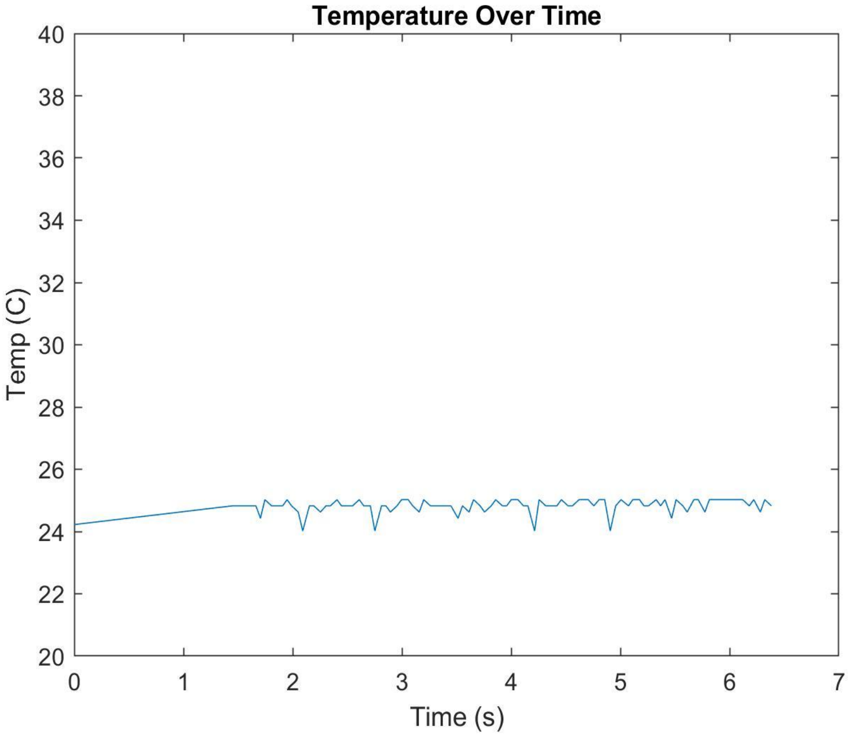


Figure 16: Temperature vs Time for Thermistor Circuit at Room Temp

After verifying the hardware and temperature sensing circuit, the thermistor was bent over and placed on the LD1117 voltage regulator chip for testing. The same setup as above was used, but this time the test was ran for 50 seconds and the 3.3V power supply was at maximum load current (800mA). The temperature vs time graph for this test is seen below in figure 17.

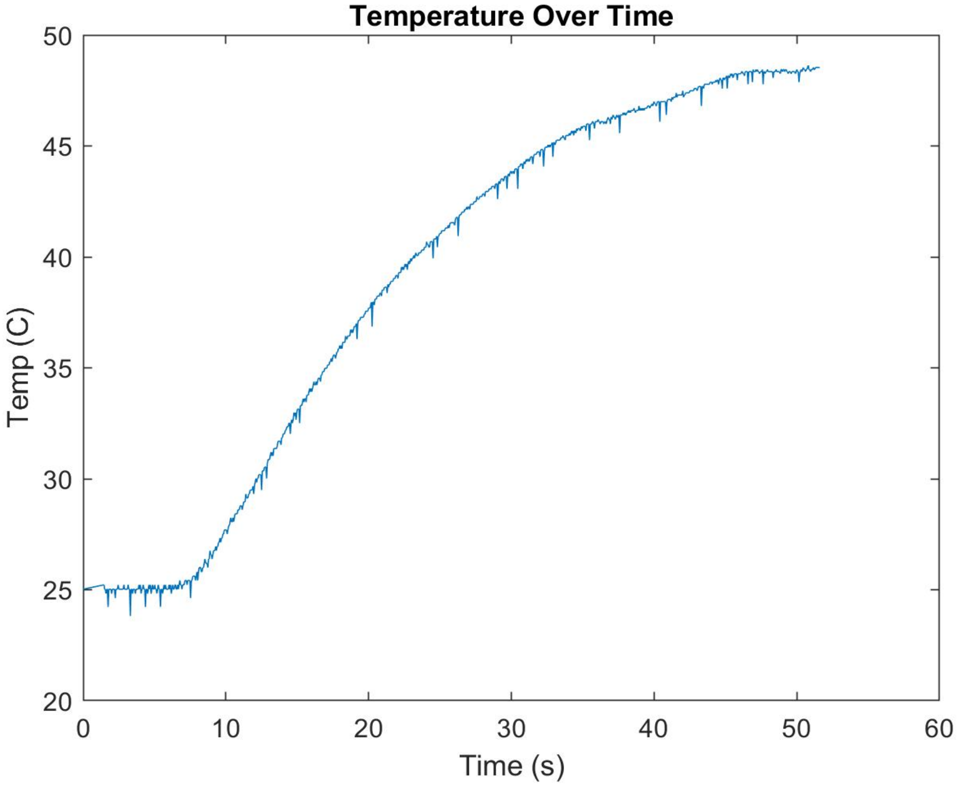


Figure 17: Temperature vs Time 3.3V Power Supply at Maximum Load

As figure 17 above shows, the temperature of the LD1117 voltage regulator chip on the 3.3V linear power supply steadily increased to a maximum 48.2^{o}C. This means that in 50 seconds the temperature of the chip increased by 23.2^{o}C. It appears that right around this 50 second mark, the temperature is beginning to reach a new steady state as the fastest increase in temperature occurred around 15 seconds with maximum load.

The easiest and most effective method to improve the thermal performance of this power supply would be with improved airflow. This could mean a small fan constantly blowing on the board and introducing fresh air which would improve the rate at which heat can be dissipated from the voltage regulator. Another relatively simple and common solution to cool IC's is done through heatsinks. For a chip this small it would likely mean a small metal contact patch with extending blades connected to the chip with thermal paste. The more surface area the metal heat sink has, the better it dissipates heat. Combining these two ideas would still be fairly simple and would prove a great improvement to the thermal performance of the voltage regulator chip.

4 5V Switching Power Supply

The 5V power supply circuit on the JD Power Supply board is a switching power supply that takes in 12V from a wall adapter as an input. Switching power supplies use switching transistors or feedback voltage regulators to transform power. These switching supplies are more efficient due to only being in the on state or the off state most of the time and not in between which is where the power is dissipated. The major tradeoff however is a much noisier signal when compared to a linear power supply.

Input and Output Power

The Arduino Nano analog input pins can only read up to 5V and because the input for the 5V switching power supply is 12V and the output is 5V, the current sensors were used to calculate the power. Connecting the two INA169 current sensor boards between the input test pins and the output test pins (one current sensor between T1 and T2, the other between T3 and T4), the voltage drop over the 0.05Ohm shunt resistors could be measured. The current sensors were set up with a gain of 100 which put the voltages nicely withing the 0V-5V range of the Arduino ADC. These voltage values could then be used along with the resistance of the shunt resistors to calculate both the input and the output currents. The input and output voltages were then measured with a DMM to get the information to calculate input and output power.

Because the switching power supply outputted more power than the linear power supply, the simple potentiometer as a load would not be adequate. For this reason, the power supply was connected to a load which was a transistor with a potentiometer connected to the base. This allowed for a heavier load and rough but usable control of the load current.

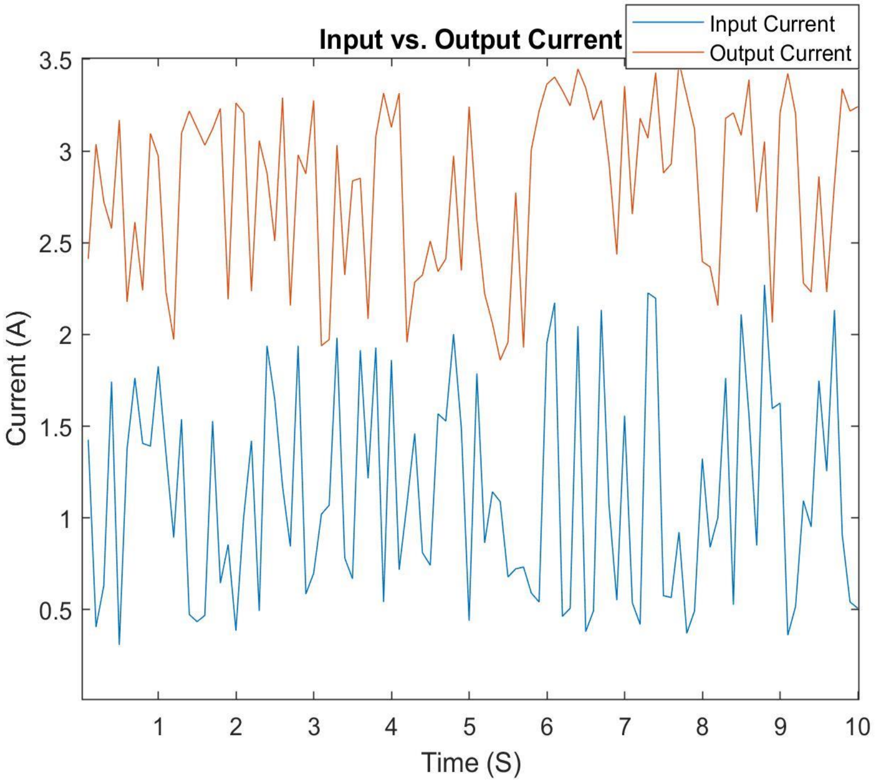


Figure 18: 5V Current Sensor Measurements vs Time at Full Load

Figure 18 above shows the data from the current sensors collected from the 5V switching power supply under a heavy load (600-800mA). This data is very noisy which is confirmation of the switching power supplies characteristics.

The average of this data was taken for two different load currents and the voltages were measured with a DMM. Using the following equation:

$$Efficiency = \frac{Power\ Out}{Power\ In}$$

… the efficiency of the power supply was calculated for each load current.

Load Current (mA)	Input Current (A)	Input Voltage (V)	Input Power (W)
300	0.0311	12.02	0.374
600-800	1.1269	12.02	13.545

Figure 19: 5V Power Supply Input Power Calculation

Load Current (mA)	Output Current (A)	Output Voltage (V)	Output Power (W)
300	0.0639	4.71	0.301
600-800	2.8089	4.71	13.23

Figure 20: 5V Power Supply Output Power Calculation

Load Current (mA)	Efficiency
300	80.50%
600-800	97.70%

Figure 21: 5V Power Supply Efficiency

From this data it is clear that the efficiency of the switching voltage regulator in the 5V power supply is much more efficient than the linear power supply. The data also shows that at a higher load, the voltage regulator (and therefor power supply) is more efficient. This matches up with the TPS54232 data sheet which has an efficiency vs output current graph that shows the same behavior.

Theoretically, if the input voltage was increased (up to 28V max according to the datasheet) the efficiency of the switching voltage regulator in the power supply would stay relatively constant. This is because the same switching characteristic is happening which is ideally either on or off and nothing in between. Though in practice the efficiency would decrease because the switching is not instant and along with this, if more voltage was inputted, more power would be dissipated in the components around the switching voltage regulator, making the entire power supply less efficient at higher voltage.

With this being said, the change in efficiency is much less drastic than in the linear power supply.

Thermal Dissipation

The method for testing the thermal behavior on the 5V switching power supply was nearly identical as to that in the 3.3V. For the 5V supply the thermistor was not placed over the voltage regulator as in the 5V switching power supply, the voltage regulator is much more efficient and therefor dissipating much less heat. For this reason the thermistor and temperature data was instead taken over the TIP29c transistor load attached to the output of the power supply.

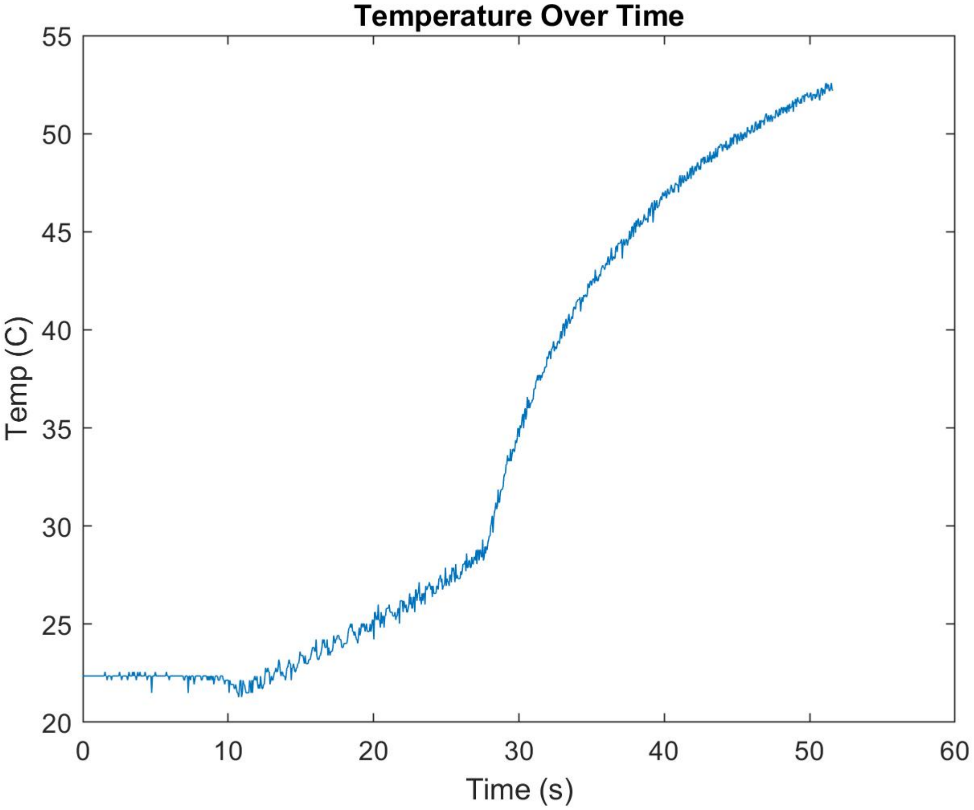


Figure 22: TIP29c 5V Temperature vs Time Graph

For this test the temperature data was taken into MATLAB via the Arduino for 50 seconds while increasing the load current through the transistor to max with the potentiometer. It appears as if the temperature begins to reach a steady state at around 53 or 54^{o}C. The TIP29c reaches a maximum recorded temperature of 53^{o}C near the end of the 50 seconds. This means that in 50 seconds with a load increasing to max, the temperature rose 30^{o}C.

5 Final Testing

For the final testing both the 3.3V and the 5V circuits on the power supply were tested using the Arduino and analyzed in MATLAB for an overall performance vs time test.

For both circuits the two current sensors were hooked up to the input and output shunt resistor test pins to measure input and output current. The potentiometer was connected alone as the load for the 3.3V linear supply and together with the transistor as the load for the 5V switching supply. The rotation vs resistance for the potentiometer was calculated and the rotation was graphed along with the other characteristics. Along with this, the thermistor circuit was again used to show the thermal behavior over the same period of time. Graphing all these factors, the potentiometer rotation, input and output voltage, input and output current, and the temperature all on the same time scale gave insight into how they all behave together.

There were although some shortcomings when running these final tests for the 5V switching power supply. To start, the potentiometer rotation was difficult to obtain as the pure load resistance could not be calculated with the transistor and changing current. For this reason it was given a relationship with the load current.

Along with this, a more troubling problem came up when the 5V switching power supply load with the potentiometer and transistor did not function as intended. In the afterthought it was predicted that connecting the potentiometer directly to the base of the transistor (without any additional fixed resistance) resulted in a high current being amplified through the transistor when the potentiometer resistance got close to 0Ohm. Something about this setup resulted in too much voltage or current going to the Arduino and essentially frying the board. For this reason another test could not be performed. This can be seen in executive summary figure on the 5V column when the potentiometer rotation nears 0 (no resistance) the resulting input current spikes, and then suddenly cuts down to nearly zero. At this spike is when the Arduino began to smoke.

6 Executive Summary

Through this report both the 3.3V and the 5V circuits on the JD Power Supply board have been explored. From learning how to take measurements on the power supply, to actually taking the measurements, to then understanding the data, insight on the characteristics has been gained. The 3.3V linear power supply is less efficient (and therefore dissipates more heat), but more stable and less noisy. The switching 5V supply is more power and thermally efficient but creates a noisy output.

Figure 23 below is a culminating test for each power supply visualizing different characteristics over time.

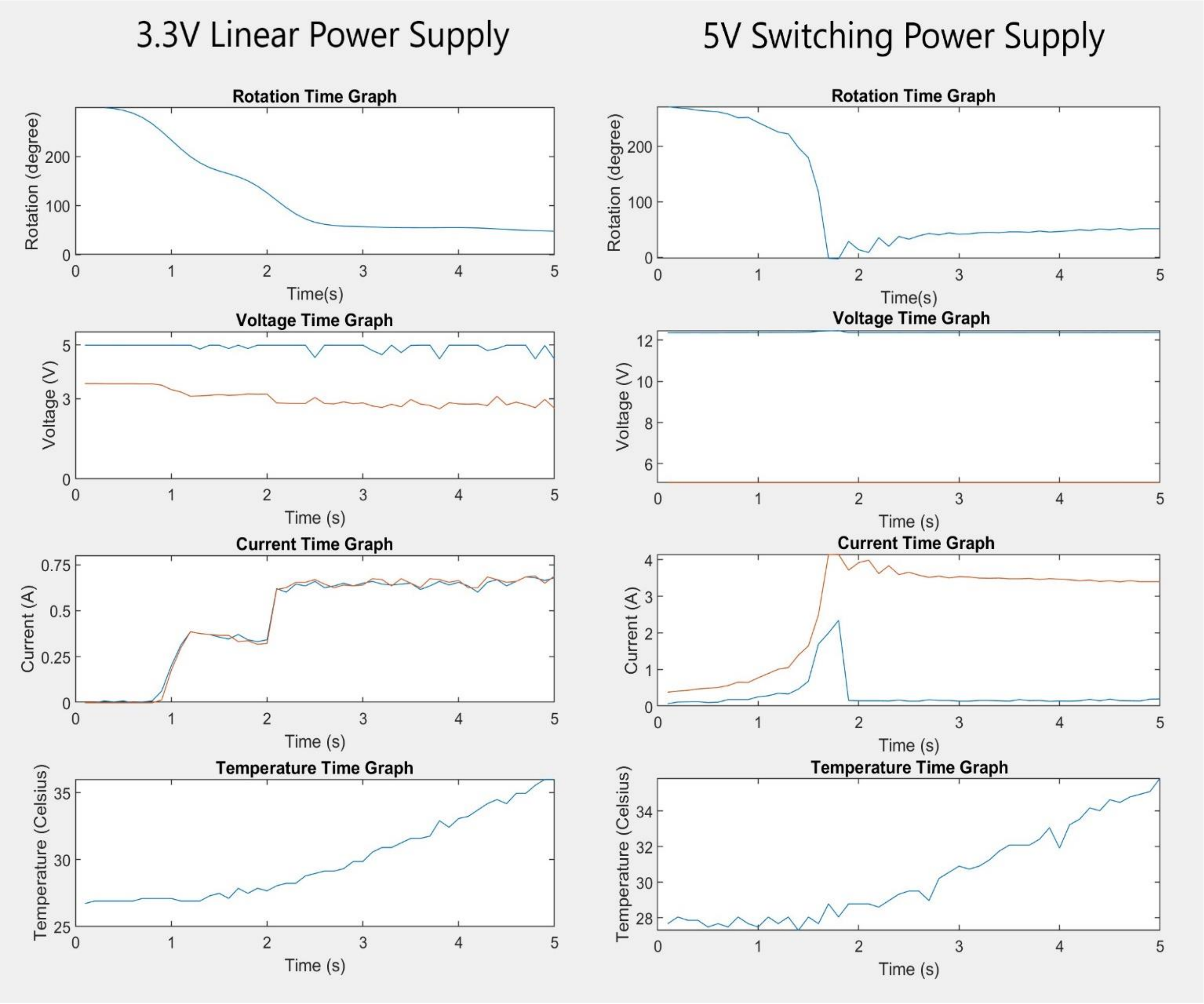


Figure 23: JD Power Supply Summary Graphs

7 Appendix

1 – Voltage Analysis MATLAB Script

```
clear all
device = serialport("COM4", 9600)
flush(device)

readline(device)

n = 100;
for i = 1:n
       Time(i) = 0.1*i
       X(i) = readline(device)
       V = strsplit(X(i))
       V1(i) = str2double(V(:,1)) / (1024) * 5
       V2(i) = str2double(V(:,2)) / (1024) * 5
       V3(i) = str2double(V(:,3)) / (1024) * 5
       V4(i) = str2double(V(:,4)) / (1024) * 5
```

```
       V1avg = mean(V1)
       V2avg = mean(V2)
       V3avg = mean(V3)
       V4avg = mean(V4)
      %voltage(i) = str2double(X)/ (1024) * 5
end

clear device

%figure
%plot(Time, Current)
%title('Current Over Time')
%xlabel('Time (s)')
%ylabel('Current (A)')
```

2 – Thermistor Temperature MATLAB Script

```
clear all
device = serialport("COM4", 9600)
flush(device)

n = 100;
R2 = 10000;
Vs = 3.7;

tic;

for i = 1:n
    Time(i) = toc;

    X(i) = readline(device);

    Vo(i) = str2double(X(i)) / (1024) * 5;

    Rt(i) = ((R2*Vs) / Vo(i)) - R2;

    temp(i) = (Rt(i) - 16662.5) / (-266.5);
end

clear device

figure
plot(Time, temp)
title('Temperature Over Time')
xlabel('Time (s)')
```

```
ylabel('Temp (C)')
```

3 – 3.3V Arduino Executive Summary Script

```
const int in_cur = A0;
const int out_cur = A1;
const int in_v = A2;
const int out_v = A3;
const int temp_v = A4;

int raw0 = 0; //input current variable
int raw1 = 0; //output current variable
int raw2 = 0; //input voltage variable
int raw3 = 0; //output voltage variable
int raw4 = 0; //temp voltage variable

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

}

void loop() {
  // put your main code here, to run repeatedly:

  raw0 = analogRead(in_cur);
  raw1 = analogRead(out_cur);
  raw2 = analogRead(in_v);
  raw3 = analogRead(out_v);
  raw4 = analogRead(temp_v);

  Serial.print(raw0);
  Serial.print(" ");

  Serial.print(raw1);
  Serial.print(" ");

  Serial.print(raw2);
  Serial.print(" ");

  Serial.print(raw3);
  Serial.print(" ");

  Serial.println(raw4);
```

```
  delay(200);
}
```

4 – 3.3V MATLAB Executive Summary Script

```
clear all
device = serialport("COM4", 9600)
flush(device)

n = 50;
R2 = 10000;
V3v3 = 3.626;

for i = 1:n
    Time(i) = 0.1*i
    X(i) = readline(device)
    V = strsplit(X(i))

    In_C(i) = str2double(V(:,1)) / (1024) * 5
    Out_C(i) = str2double(V(:,2)) / (1024) * 5

    In_V(i) = str2double(V(:,3)) / (1024) * 5
    Out_V(i) = str2double(V(:,4)) / (1024) * 5

    Temp_V(i) = str2double(V(:,5)) / (1024) * 5
    Rt(i) = ((R2*V3v3) / Temp_V(i)) - R2
    temp(i) = (Rt(i) - 16662.5) / (-266.5)

    %Rpot = Out_V(i) / (Out_C(i))
    %per_pot = Rpot / 10000
    %deg_rot(i) = per_pot * 300
    deg_rot(i) = 300 - 375*(Out_C(i))

end

deg_rot = smoothdata(deg_rot, 'gaussian');

clear device

subplot(4,1,1);
title('Rotation Time Graph')
plot(Time, deg_rot)
xlabel('Time(s)')
ylabel('Rotation (degree)')
```

```
subplot(4,1,2);
title('Voltage Time Graph')
plot(Time, In_V)
hold on
plot(Time, Out_V)
hold off
xlabel ('Time (s)')
ylabel('Voltage (V)')

subplot(4,1,3);
title('Current Time Graph')
plot(Time, In_C)
hold on
plot (Time, Out_C)
hold off
xlabel('Time (s)')
ylabel('Current (A)')

subplot(4,1,4)
title('Temperature Time Graph')
plot(Time, temp)
xlabel('Time (s)')
ylabel('Temperature (Celsius)')
```

5 – 5V Arduino Executive Summary Code

```
const int in_cur = A0;
const int out_cur = A1;
const int out_v = A2;
const int temp_v = A3;

int raw0 = 0; //input current variable
int raw1 = 0; //output current variable
int raw2 = 0; //output voltage variable
int raw3 = 0; //temp voltage variable

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:

  raw0 = analogRead(in_cur);
  raw1 = analogRead(out_cur);
```

```
  raw2 = analogRead(out_v);
  raw3 = analogRead(temp_v);

  Serial.print(raw0);
  Serial.print(" ");

  Serial.print(raw1);
  Serial.print(" ");

  Serial.print(raw2);
  Serial.print(" ");

  Serial.println(raw3);

  delay(200);
}
```

6 – 5V MATLAB Executive Summary Script

```
clear all
device = serialport("COM4", 9600)
flush(device)

n = 50;
R2 = 10000;
V3v3 = 3.626;
Vt2 = 12.35 %measured with DMM%
Vt4 = 5.08 %measured with the DMM%

for i = 1:n
    Time(i) = 0.1*i
    X(i) = readline(device)
    V = strsplit(X(i))

    In_C(i) = str2double(V(:,1)) / (1024) * 5
    Out_C(i) = str2double(V(:,2)) / (1024) * 5

    In_V(i) = Vt2 + In_C(i)*(0.05)
    Out_V(i) = Vt4

    Temp_V(i) = str2double(V(:,4)) / (1024) * 5
    Rt(i) = ((R2*V3v3) / Temp_V(i)) - R2
    temp(i) = (Rt(i) - 16662.5) / (-266.5)

    deg_rot(i) = 300 - 73.1*(Out_C(i))
```

```
end

clear device

subplot(4,1,1);
title('Rotation Time Graph')
plot(Time, deg_rot)
xlabel('Time(s)')
ylabel('Rotation (degree)')

subplot(4,1,2);
title('Voltage Time Graph')
plot(Time, In_V)
hold on
plot(Time, Out_V)
hold off
xlabel ('Time (s)')
ylabel('Voltage (V)')

subplot(4,1,3);
title('Current Time Graph')
plot(Time, In_C)
hold on
plot (Time, Out_C)
hold off
xlabel('Time (s)')
ylabel('Current (A)')

subplot(4,1,4)
title('Temperature Time Graph')
plot(Time, temp)
xlabel('Time (s)')
ylabel('Temperature (Celsius)')
```