Board 4 Report
ECEN 3730 - PCB Design and Manufacturing

Completed by:
Shane McCammon

Introduction

Board 4, the instrument droid, is a four-layer PCB designed to test and characterize voltage sources and Voltage Regulator Modules (VRMs) as an Arduino shield that can be attached to any Arduino Uno R3. It determines the Thevenin voltage and resistance of these sources by applying a controlled electronic load, drawing current, and calculating key parameters such as voltage drop and internal resistance. The board is also intended to have smart LED indicators and a buzzer to improve usability. This report details the development process, including schematics, PCB layout, board assembly, and performance testing.

What does it mean to work?

Board 4 is to be considered "working" if it meets all the requirements outlined in the Plan of Record (POR) found below. This includes successfully running Arduino code, measuring and displaying the Thevenin voltage and resistance of a voltage source as expected. The power supply, test points, and smart features must also function correctly to support testing and debugging. To summarize, the board must operate reliably and perform all tasks.

Board 4 Plan of Record (POR)

1. A power plug to use an external 5V AC to DC charger to power board.
2. An additional power plug and terminal to measure external voltage sources
3. Indicator LEDs for 5V power
4. Properly assigned header connections for connection to Arduino Uno board
5. Plan test points for these signals:
    a. 5V input
    b. The VRM+ and VRM- for VRM
    c. Op-Amp output
    d. DAC output
6. Placing decoupling capacitors closer to all the IC's to reduce the switching noise.
7. 4-layer board with top and bottom as signal layers and middle layers as ground.
8. Copper poured return plane (ground).
9. Include smart indicator features such as LEDs and Buzzer
10. Use best scope measurement practices when you test your board

Schematics and Design

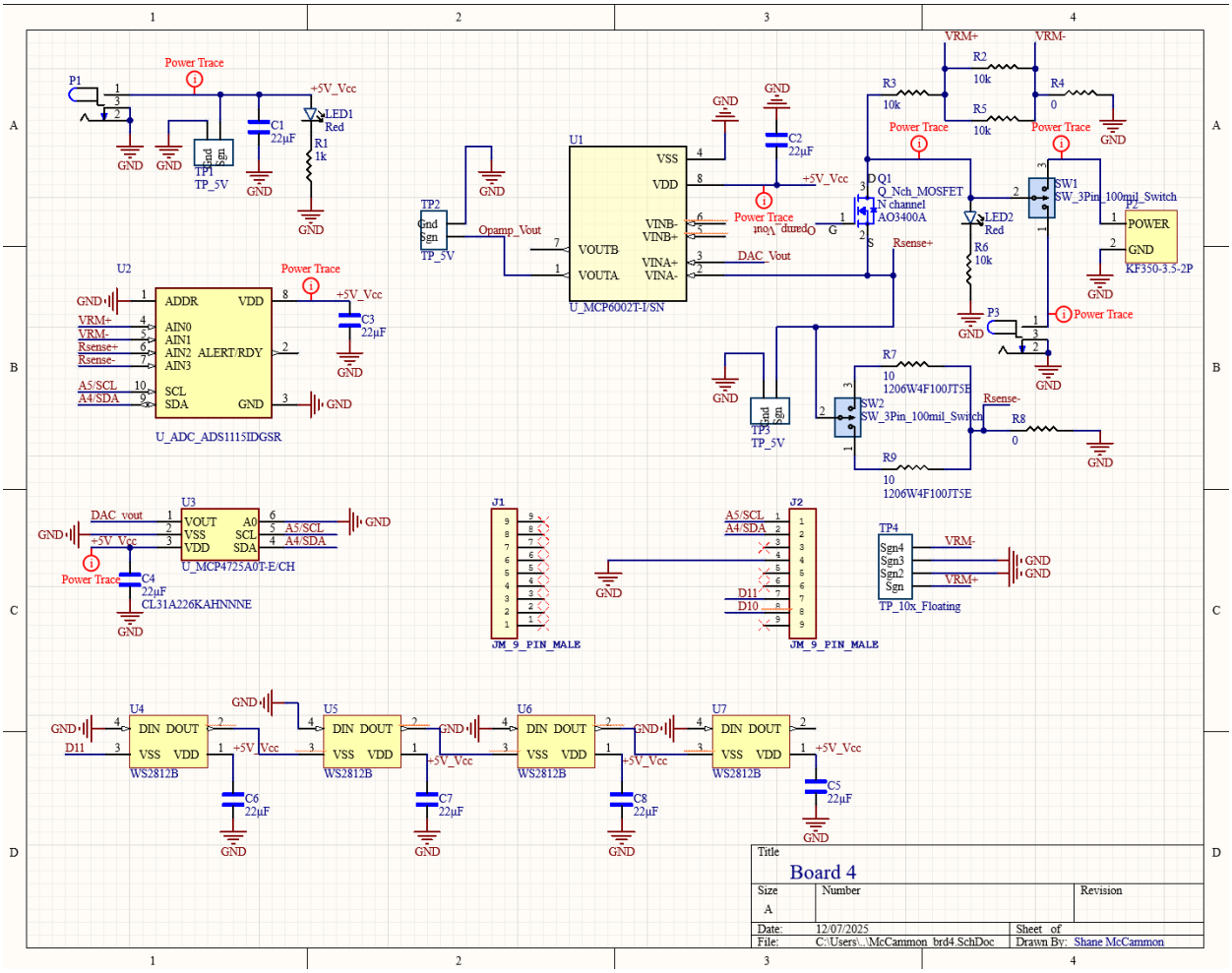

Figure 1: Altium schematic of Board 4

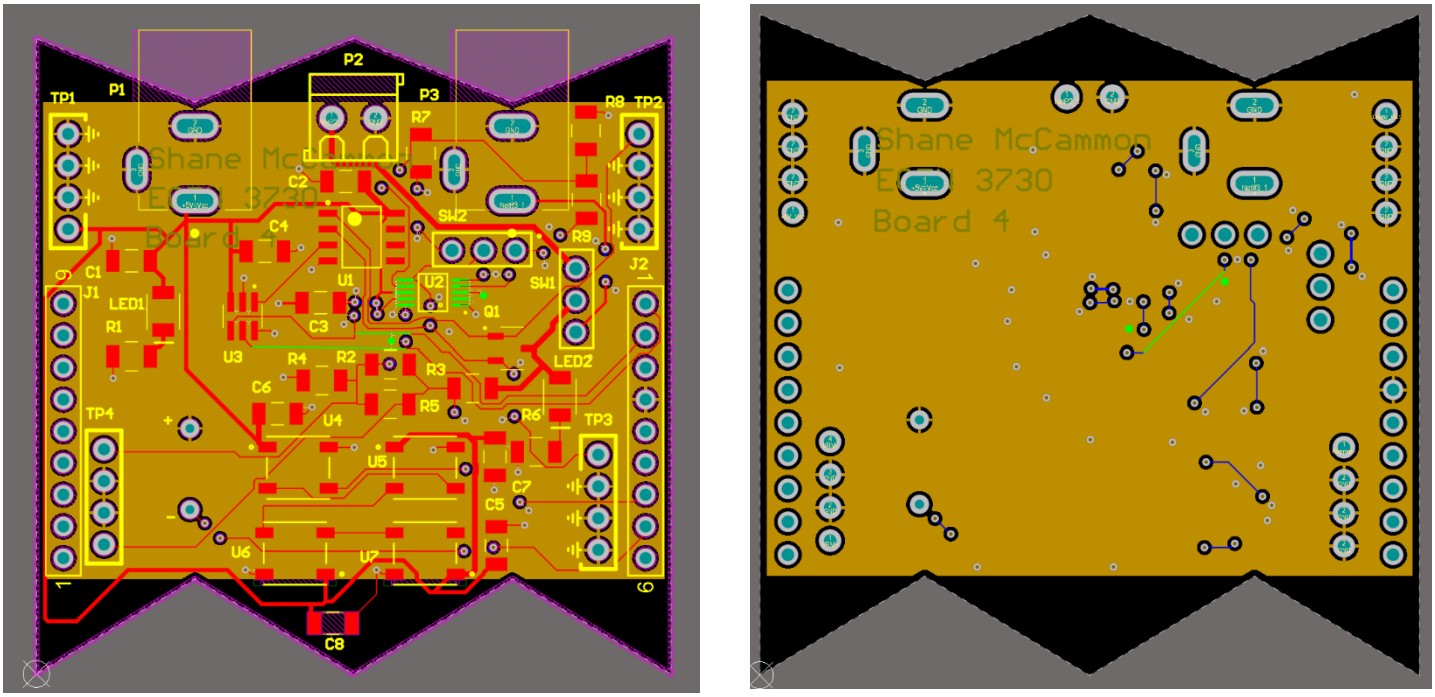

Figure 2: Altium board layout

Figure 3: Printed and Assembled Board 4

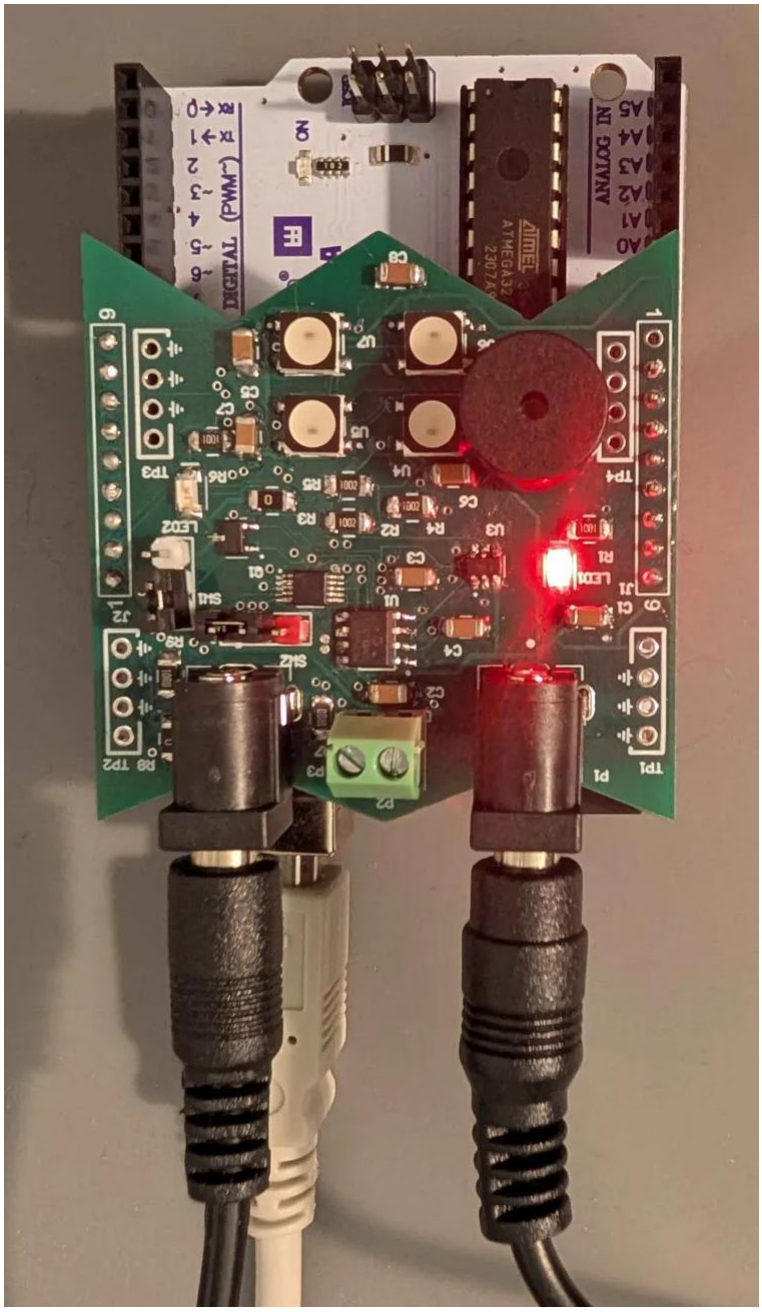

Figure 4: Functioning Board 4

Output Waveforms & Measurements

Below is the Arduino code used to measure $R_{Thevenin}$ of three different VRMs: 5V power supply, 9V power supply, and a function generator:

```
// vrm characterizer board
#include <Wire.h>
#include <Adafruit_MCP4725.h>
#include <Adafruit_ADS1X15.h>

Adafruit_ADS1115 ads;
Adafruit_MCP4725 dac;

float R_sense = 10.0; //current sensor
long itime_on_msec =100; //on time for taking measurements
long itime_off_msec =itime_on_msec * 10; // time to cool off
int iCounter_off = 0;// counter for number of samples off
int iCounter_on = 0; //counter for number of samples on
float v_divider = 5000 / 15000.0; //voltage divider on the VRM
float DAC_ADU_per_v = 4095.0 / 5.0; //conversion from volts to ADU
int V_DAC_ADU; // the value in ADU to output on the DAC
int I_DAC_ADU; // the current we want to output
float I_A = 0.0; //the current we want to output,in amps
long itime_stop_usec; // this is the stop time for each loop
float ADC_V_per_ADU = 0.125 * 1e-3; // the voltage of one bit on the
gainof 1 scale
float V_VRM_on_v; // the value of the VRM voltage
float V_VRM_off_v; // the value of the VRM voltage
float I_sense_on_A; // the current through the sense resistor
float I_sense_off_A; // the current through the sense resistor
float I_max_A = 0.25; // max current to set for
int npts = 20; //number of points to measure
float I_step_A = I_max_A / npts; //step current change
float I_load_A; // the measured current load
float V_VRM_thevenin_v; float V_VRM_loaded_v;
float R_thevenin; int i;
```

```
//The next part is the set up for the ADC and the DAC:
void setup()
{
  Serial.begin(115200);
  dac.begin(0x60); // address is either 0x60, 0x61, 0x62,0x63, 0x64 or
0x65
  dac.setVoltage(0, false); //sets the output current to 0 initially
  // ads.setGain(GAIN_TWOTHIRDS); // 2/3x gain +/- 6.144V 1 bit = 3mV
0.1875mV (default)
  ads.setGain(GAIN_ONE); // 1x gain +/- 4.096V 1 bit = 2mV 0.125mV
  // ads.setGain(GAIN_TWO); // 2x gain +/- 2.048V 1 bit = 1mV 0.0625mV
  // ads.setGain(GAIN_FOUR); // 4x gain +/- 1.024V 1 bit = 0.5mV 0.03125mV
  // ads.setGain(GAIN_EIGHT); // 8x gain +/- 0.512V 1 bit = 0.25mV
0.015625mV
  // ads.setGain(GAIN_SIXTEEN); // 16x gain +/- 0.256V 1 bit = 0.125mV
0.0078125mV
  ads.begin(0x48); // note- you can put the address of the ADS111 here if
needed
  ads.setDataRate(RATE_ADS1115_860SPS);// sets the ADS1115 for higher
speed
}

void loop()
{
  for (i = 1; i <= npts; i++)
  {
    I_A = i * I_step_A; dac.setVoltage(0, false); //sets the output
current
    func_meas_off();
    func_meas_on();
    dac.setVoltage(0, false); //sets the output current
    I_load_A = I_sense_on_A - I_sense_off_A; //load current
    V_VRM_thevenin_v = V_VRM_off_v;
    V_VRM_loaded_v = V_VRM_on_v;
    R_thevenin = (V_VRM_thevenin_v - V_VRM_loaded_v) / I_load_A;
```

```
    //if (V_VRM_loaded_v < 0.75 * V_VRM_thevenin_v) i = npts; //stops the
ramping
    Serial.print(i);
    Serial.print(", ");
    Serial.print(I_load_A * 1e3, 3);
    Serial.print(", ");
    Serial.print(V_VRM_thevenin_v, 4);
    Serial.print(", ");
    Serial.print(V_VRM_loaded_v, 4);
    Serial.print(", ");
    Serial.println(R_thevenin, 4);
  }
  Serial.println("done");
  delay(30000);
}

void func_meas_off()
{
  dac.setVoltage(0, false);
  //sets the output current
  iCounter_off = 0; //starting the current counter
  V_VRM_off_v = 0.0; //initialize the VRM voltage averager
  I_sense_off_A = 0.0; // initialize the current averager
  itime_stop_usec= micros() + itime_off_msec * 1000; // stop time
  while (micros() <= itime_stop_usec)
  {
    V_VRM_off_v = ads.readADC_Differential_0_1() * ADC_V_per_ADU /
v_divider + V_VRM_off_v;
    I_sense_off_A = ads.readADC_Differential_2_3() * ADC_V_per_ADU /
R_sense + I_sense_off_A;
    iCounter_off++;
  }
  V_VRM_off_v = V_VRM_off_v / iCounter_off;
  I_sense_off_A = I_sense_off_A / iCounter_off;
  // Serial.print(iCounter_off);Serial.print(", ");
  // Serial.print(I_sense_off_A * 1e3, 4); Serial.print(", ");
  // Serial.println(V_VRM_off_v, 4);
```

```
}

void func_meas_on()
{
  //now turn on the current
  I_DAC_ADU = I_A * R_sense * DAC_ADU_per_v;
  dac.setVoltage(I_DAC_ADU, false); //sets the output current
  iCounter_on = 0;
  V_VRM_on_v = 0.0; //initialize the VRM voltage averager
  I_sense_on_A = 0.00; // initialize the current averager
  itime_stop_usec = micros() + itime_on_msec * 1000; // stop time
  while (micros() <= itime_stop_usec)
  {
    V_VRM_on_v = ads.readADC_Differential_0_1() * ADC_V_per_ADU /
v_divider + V_VRM_on_v;
    I_sense_on_A = ads.readADC_Differential_2_3() * ADC_V_per_ADU /
R_sense +I_sense_on_A;
    iCounter_on++;
  }
  dac.setVoltage(0, false); //sets the output current to zero
  V_VRM_on_v = V_VRM_on_v / iCounter_on;
  I_sense_on_A = I_sense_on_A / iCounter_on;
  // Serial.print(iCounter_on);Serial.print(", ");
  // Serial.print(I_sense_on_A * 1e3, 4);Serial.print(", ");
  // Serial.println(V_VRM_on_v, 4);
}
```

5V AC/DC Power Adapter Thevenin Resistance Measurements:

Index	Current (mA)	V_th (V)	V_loaded (V)	R_th (Ω)
1	11.827	5.2115	5.2096	0.1632
2	23.723	5.2111	5.2071	0.1674
3	35.823	5.2108	5.1964	0.4033
4	47.807	5.2109	5.2002	0.2227
5	60.024	5.2115	5.1839	0.4599
6	72.092	5.2113	5.1730	0.5303
7	84.186	5.2114	5.1921	0.2293
8	96.448	5.2111	5.1836	0.2852
9	108.464	5.2109	5.1863	0.2271
10	120.670	5.2113	5.1842	0.2244
11	132.864	5.2114	5.1804	0.2333
12	144.834	5.2114	5.1776	0.2335
13	156.929	5.2115	5.1763	0.2248
14	169.058	5.2110	5.1724	0.2279
15	181.267	5.2110	5.1706	0.2231
16	193.482	5.2117	5.1676	0.2284
17	205.512	5.2115	5.1648	0.2270
18	217.438	5.2113	5.1620	0.2266
19	229.697	5.2110	5.1594	0.2245
20	241.885	5.2114	5.1568	0.2256

\

Table 1: Measurements collected using Arduino serial monitor with 5V supply

9V AC/DC Power Adapter Thevenin Resistance Measurements:

Index	Current (mA)	V_th (V)	V_loaded (V)	R_th (Ω)
1	11.808	9.248	9.2463	0.1438
2	23.733	9.2485	9.2791	-1.2908
3	35.829	9.2485	9.2784	-0.8332
4	47.839	9.2489	9.2779	-0.6045
5	60.031	9.2498	9.2761	-0.4375
6	72.104	9.2494	9.2757	-0.3641
7	84.147	9.2497	9.2678	-0.2152
8	96.412	9.2501	9.2623	-0.1260
9	108.420	9.2496	9.2576	-0.0741
10	120.640	9.2495	9.2522	-0.0221
11	132.870	9.2491	9.2473	0.0140
12	144.857	9.2492	9.2447	0.0313
13	156.959	9.2491	9.2419	0.0455
14	169.092	9.2494	9.2364	0.0766
15	181.325	9.2493	9.2308	0.1018
16	193.517	9.2490	9.2258	0.1201
17	205.577	9.2491	9.2233	0.1256
18	217.532	9.2494	9.2217	0.1275
19	229.785	9.2495	9.2202	0.1276
20	241.933	9.2491	9.2177	0.1297

Table 2: Measurements collected using Arduino serial monitor with 9V supply

Function Generator Thevenin Resistance Measurements:

Index	Current (mA)	V_th (V)	V_loaded (V)	R_th (Ω)
1	11.914	4.9665	4.3412	52.4887
2	23.904	4.9667	3.6834	53.6882
3	36.065	4.9653	3.0369	53.4702
4	48.123	4.9678	2.4921	51.4463
5	60.401	4.9680	1.8972	50.8399
6	72.517	4.9683	1.2601	51.1357
7	81.962	4.9681	0.8355	50.4212
8	81.472	4.9687	0.8306	50.7912
9	82.716	4.9691	0.8432	49.8805
10	82.717	4.9694	0.8431	49.8844
11	82.720	4.9692	0.8432	49.8787
12	82.802	4.9692	0.8440	49.8197
13	82.903	4.9692	0.8450	49.7466
14	82.924	4.9691	0.8453	49.7304
15	82.917	4.9693	0.8452	49.7374
16	82.918	4.9693	0.8453	49.7360
17	82.905	4.9693	0.8450	49.7466
18	82.902	4.9692	0.8451	49.7469
19	82.916	4.9691	0.8453	49.7360
20	82.911	4.9691	0.8452	49.7388

Table 3: Measurements collected using Arduino serial monitor with function generator

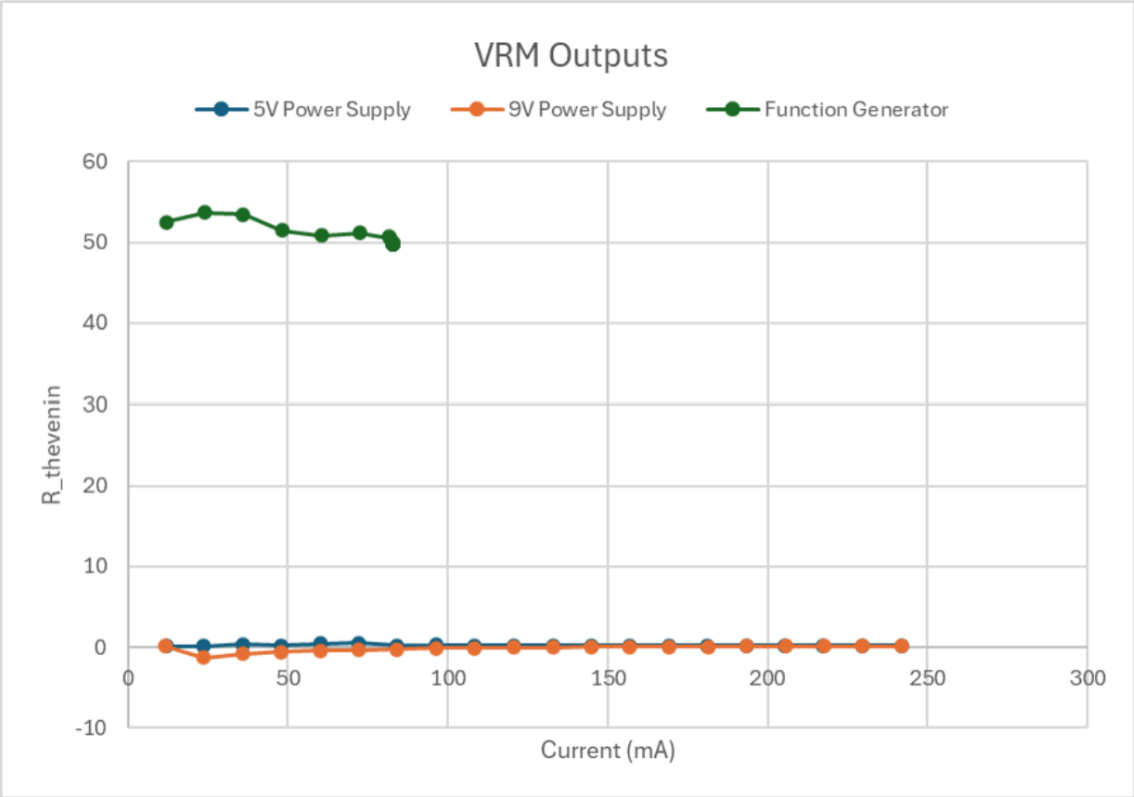

Figure 5: Graph of current vs. Thevenin resistance of all three VRMs

For validation, I characterized three different VRM sources: the 5 V supply, the 9 V supply, and the function generator output. By comparing how these known and unknown sources behaved on the VRM tester, I could check whether it produced consistent, sensible results. The measured outputs closely matched what we saw earlier with the solderless breadboard prototype, indicating that the new PCB was assembled correctly and operates as intended. This agreement also increases our confidence in the accuracy of the voltage measurements.

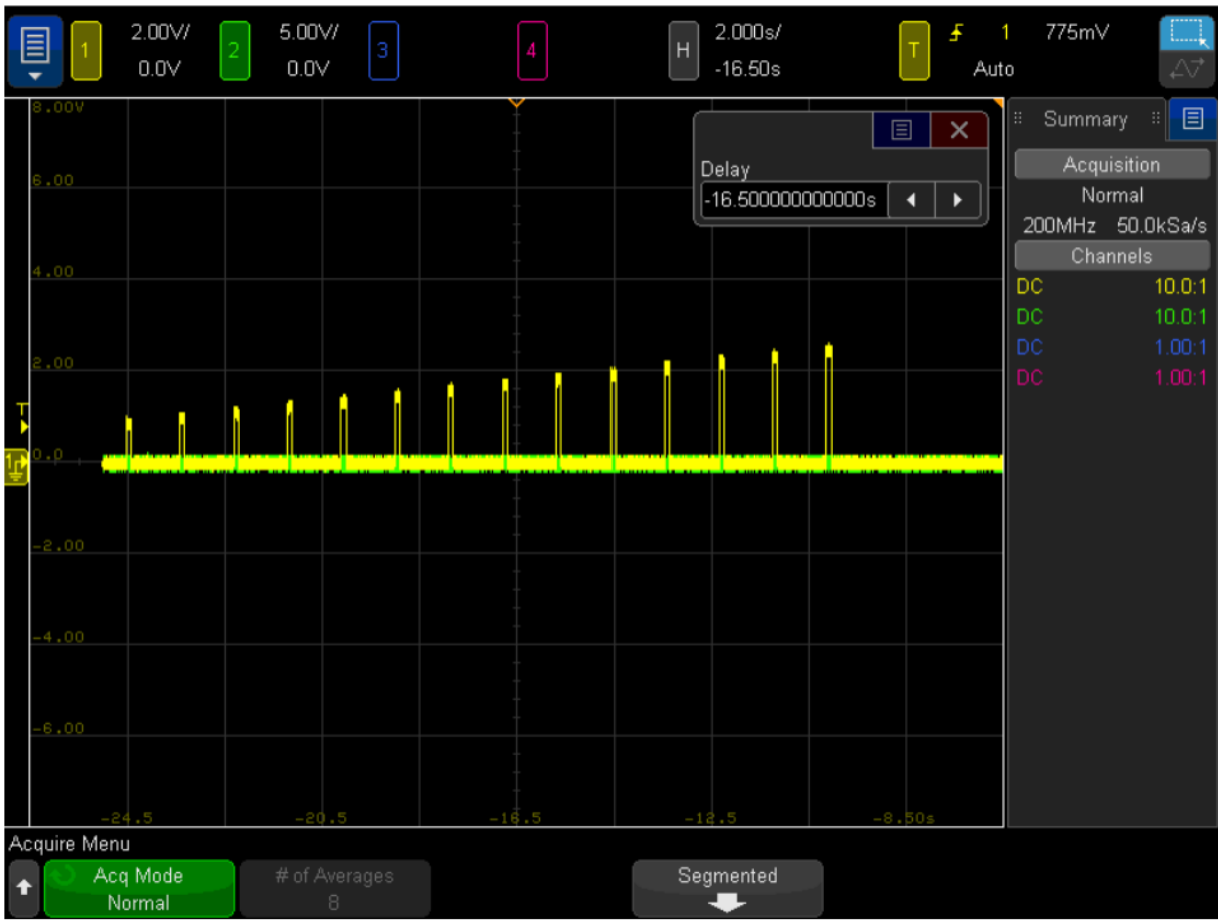

Figure 6: DAC Output

Issues

I submitted my board for manufacturing independently and completely mis-wired my smart LEDs and buzzer. I was initially using an incorrect footprint for the Smart LEDs and after acquiring the correct one I did not notice the pinouts were slightly changed. Fixing this would be a simple solution in Altium but due to delivery lead time from China, I would not get a fixed board in time. The LEDs simply need rotated and the buzzer needs a transistor between power and the Arduino signal. I manually created the footprint for the buzzer, which fit perfectly, but I forgot to add the transistor to the circuit. The Arduino does not output enough current to reliably run the buzzer without burning the output pins and the buzzer does not operate as an indicator without being attached to the MOSFET. Outside of these issues, the measurement circuit performed well.

Conclusion

Board 4 provided some valuable insights into circuit design, debugging, and characterization. The board measured Thevenin voltage and resistance of power supplies successfully which demonstrated it's ability to characterize VRMs accurately.

Key Learnings:

1. In this application a MOSFET instead of a BJT was preferred because it eliminated base current interference and provided more accurate current measurement.
2. Triple check design and have another person check it prior to manufacturing as there can be significant cost and time consequences.
3. Decoupling capacitor placement is crucial for minimizing noise and ensuring stability.
4. When troubleshooting, be open to iterative design improvements such as manually designing footprints while keeping overall functionality in mind.