Board 4 Report
Instrument Droid

- Vaishnavi Patekar
vaishnavi.patekar@colorado.edu

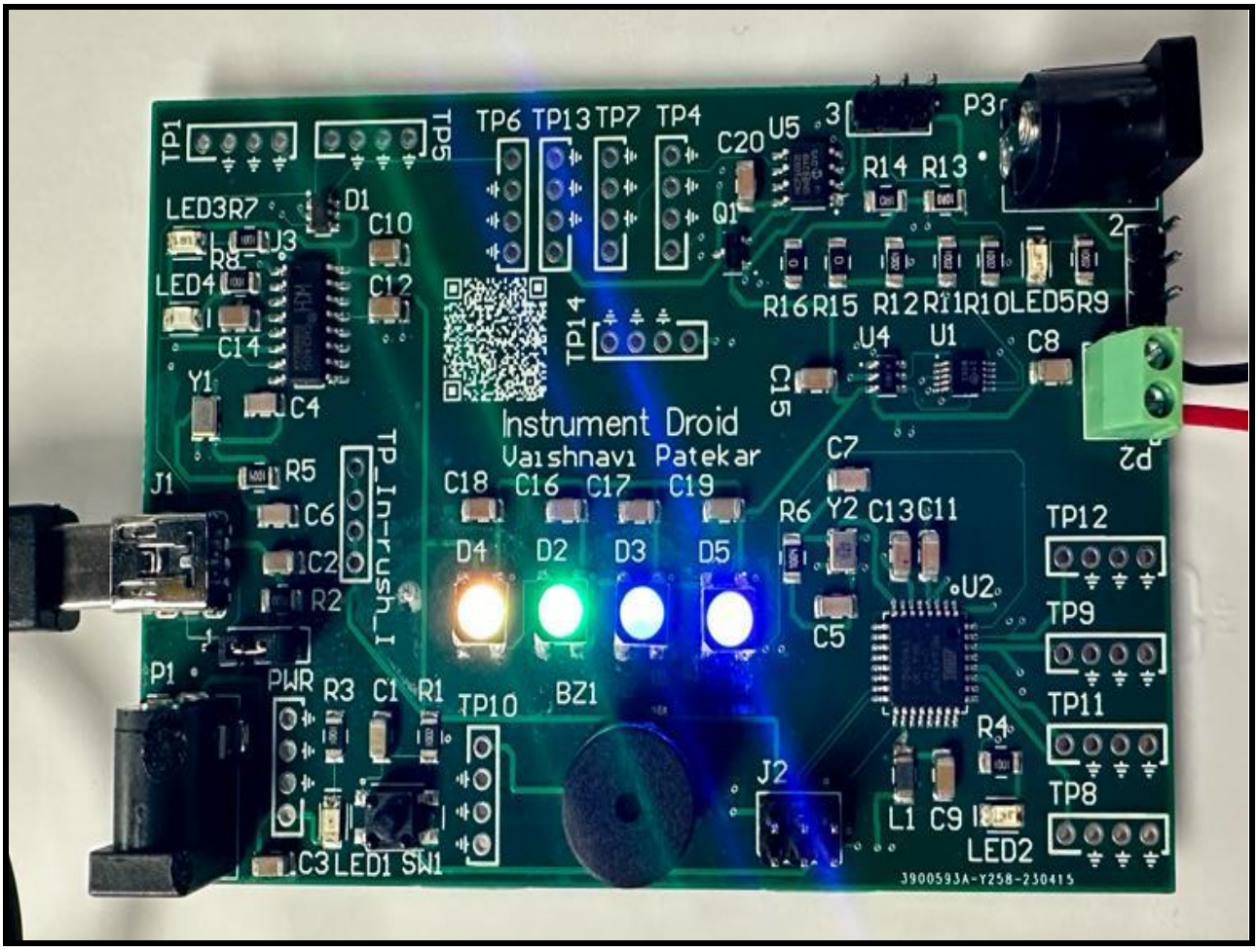


Table of Contents

Overview

Instrument droid is a 4-later board with a microcontroller and data acquisition system. It is used to measure the output impedance of any voltage source, such as a power supply or even just a digital output pin.

Using this board, the Thevenin output resistance of any supply can be measured with respect to the change load current. With the plot of the Thevenin resistance vs the current load, instrument droid can be used to characterize multiple voltage sources such as

- 5 V wall wart
- 9 V wall wart
- 12 V wall wart
- 6V power supply
- AA battery
- Coin cell battery
- LiPo battery
- Digital output pin of an Arduino
- Output of a 601 opamp
- Output of TLV4110 opamp
- Output of a 555 timer
- Output of hex inverter
- Output of 5 V rail of an Arduino
- output of 3.3 V rail of an Arduino
- Output of 3.3 V LDO
- Output of DAC

Specifications

This board would include practically all of the functionality of the Golden Arduino GPIO and power pins since this board is no longer a general-purpose board and will be utilized for a specialized purpose.

- Atmega 328 microcontroller
- CH340g USB to UART interface chip
- 16 MHz resonator for generating a clock. A 12 MHz resonator for the CH340g.
- Appropriate decoupling capacitors
- Connector for the SPI and boot-loading pins
- TVS chip to protect the data pins from ESD
- Power from the power plug or the USB connector, but not both at the same time
- Reset switch with a debounce capacitor
- 3.3 V LDO, not used by a component on the board, but available on one of the header pins.
- Header sockets that match the location of the standard Arduino board so that you can plug a shield into your Uno board
- Board Size: 3.9 x 2.5
- Test points on USB, I2C, and UART lines
- 4 Smart LEDs indicating the measurements
- Buzzer indicating the completion of the measurement

Circuit Diagram

Figure 1 below shows the circuit diagram of the Instrument Droid.

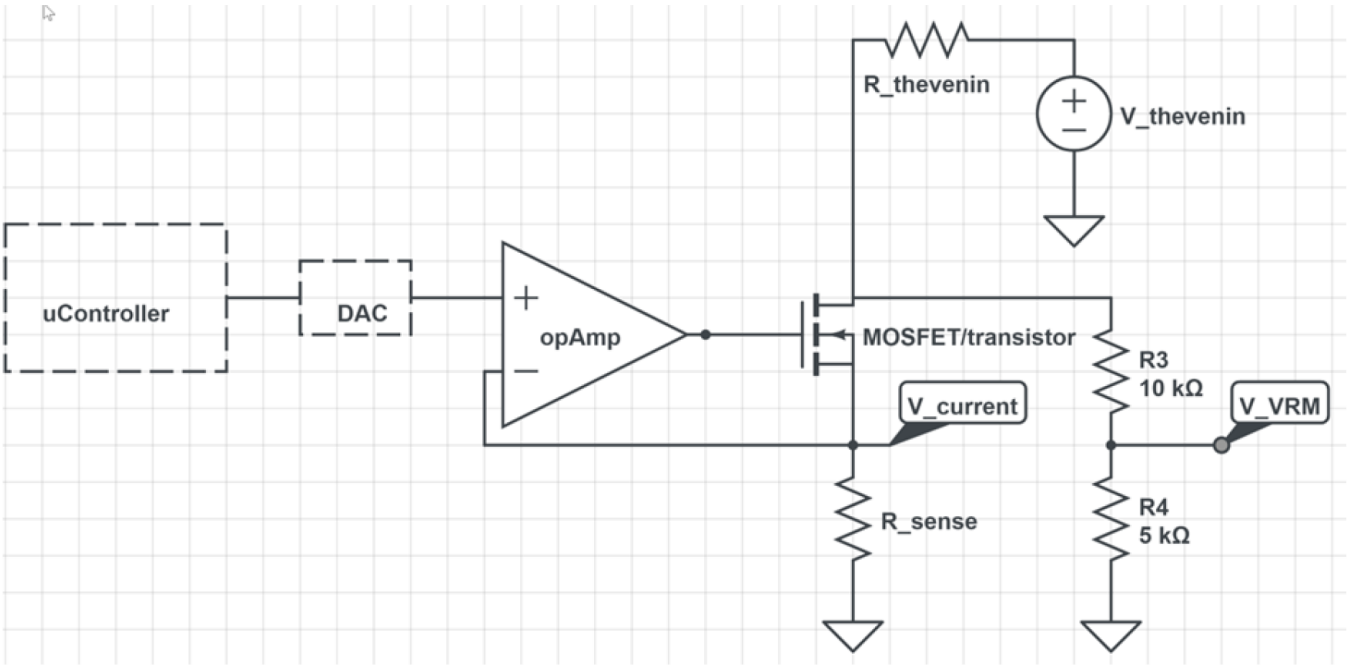


Figure 1: Instrument Droid Circuit Diagram

The function of the above circuit is to measure the Thevenin voltage and the Thevenin resistance of any connected power source. The microcontroller will continuously send out analog signals via the DAC module interfaced via I2C, increasing the step size with each repetition. All of the readings used as input to calculate Thevenin's Resistance are obtained using an ADC module that is also interfaced through I2C. The differential amplifier receives feedback from the FET source as it receives the DAC signal. The transistor can be turned ON or OFF with the aid of this feedback. The measured voltage would be VThevenin while the transistor is turned on and VLoaded when it is off. The transistor's Drain terminal is where the VRM's input is provided.

Schematic

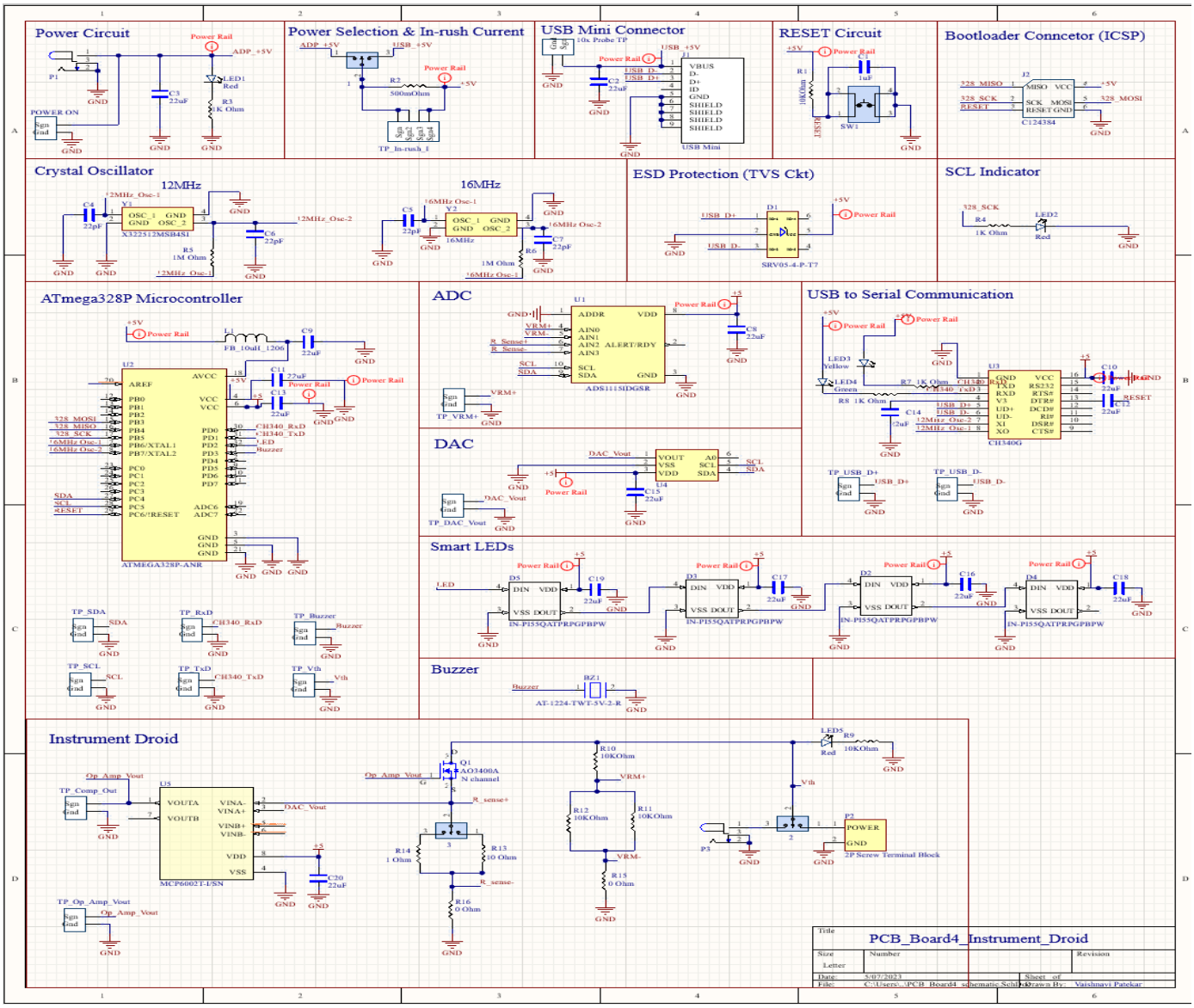


Figure 2: Schematic for Instument Droid Board

Layout

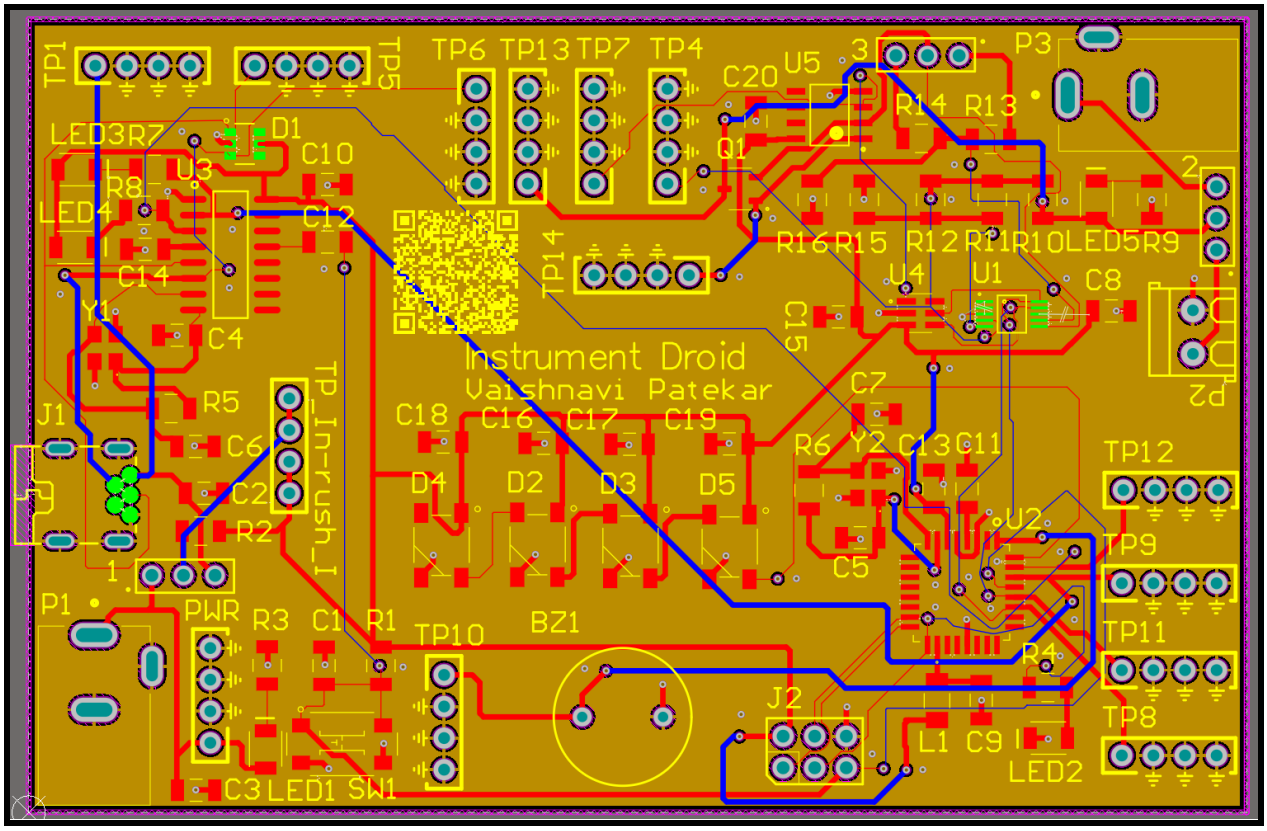


Figure 3: Layout for Golden Arduino Board

3D View Of the Board Layout

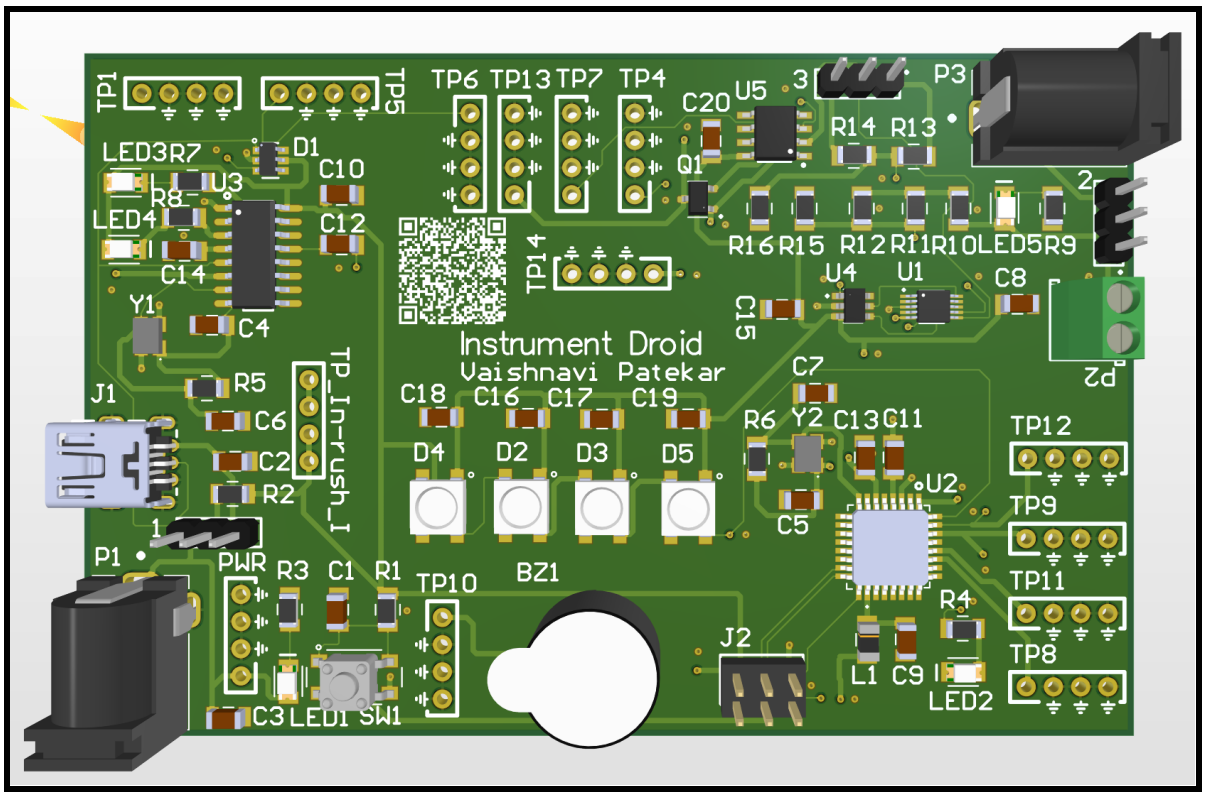


Figure 4: 3D view of the Layout for the Golden Arduino Board

Bare PCB Board from JLC

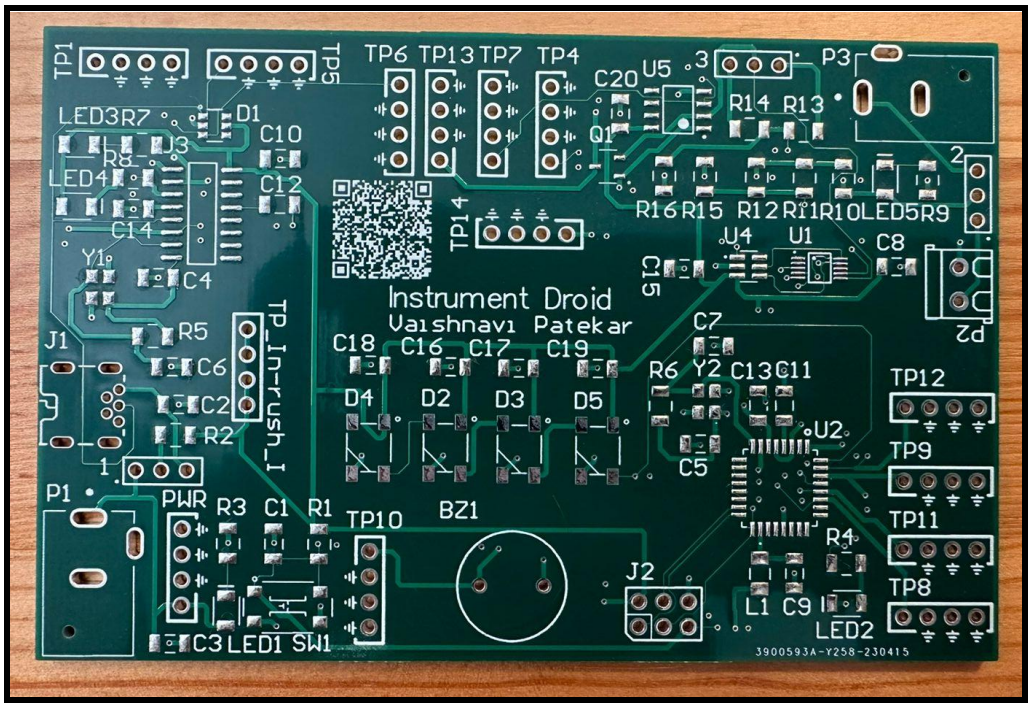


Figure 5: Bare PCB Board from JLC

Assembled PCB Board

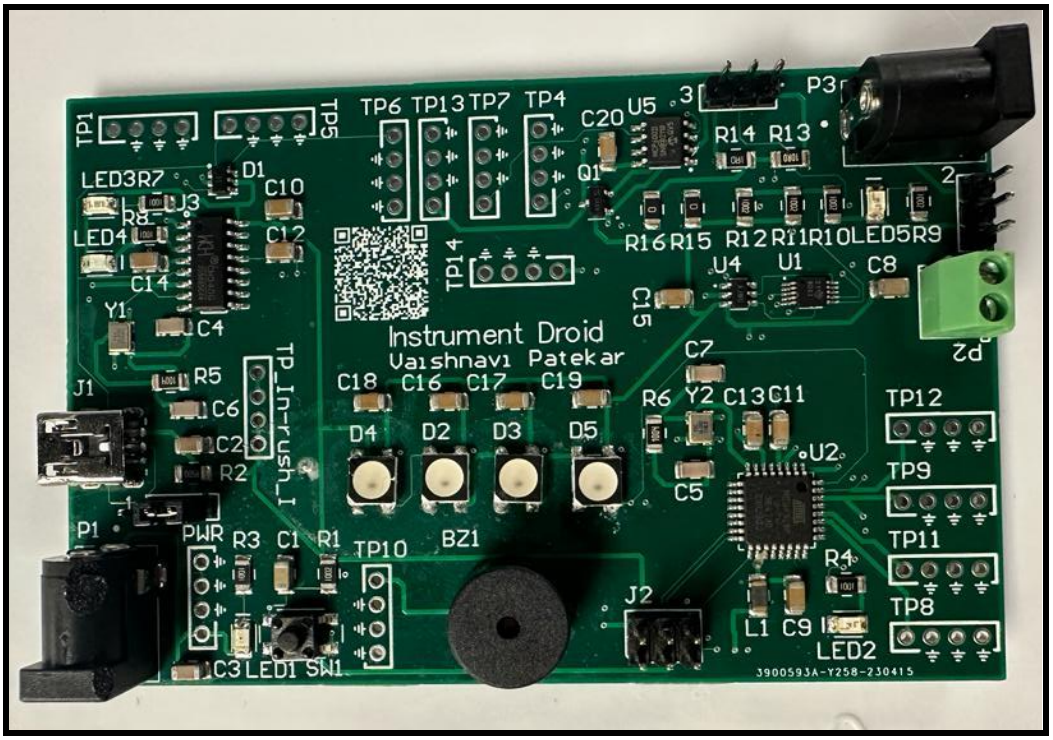


Figure 6: Assembled Board from JLCPCB

Key features of the Instrument Droid Board Design

- It is a 4-layer PCB board where top and bottom layers are used for routing and the middle two layers are used as the return planes.
- Power traces are of 20 mils thickness.
- The decoupling capacitor's location is in near proximity to the Vcc pins helps in reducing switching noise.

List of the non-commodity parts

Sr. No.	Part Name & Data-sheet Link	Reference Number
1.	Atmega328	ATMEGA328P-ANR
2.	USB to Serial	CH340G
3.	Crystal Oscillators 16Mhz	X322516MLB4SI
4.	Crystal Oscillators 12Mhz	X322512MSB4SI
5.	TVS diode	SRV05-4
6.	16-bit ADC	ADS1115
7.	12-bit DAC	MCP4725
8.	Operational Amplifier	MCP6002
9.	RGBW Digital Smart LED	-
10.	Buzzer	-

Board Bring-Up

The bring-up process involved the following steps:

1. Connection test on bare-board
2. Place the 1206 components and solder them precisely
3. Connection test after the component soldering
4. Boot-loading the board using a commercial Arduino

What does it mean to work?

The Instrument Droid board, in addition to having similar functionality to the Golden Arduino, would support the addition and operation of the following to be able to say "It worked":

Sr. No.	Function
	Power Source (5v & 3.3V)
2.	Oscillator Circuit: 12Mhz & 16Mhz
3.	Bootloading through ICSP
4.	USB Detection
5.	COde uploading via USB
6.	Smart LED Circuit
7.	Buzzer Circuit
8.	Onboard Droid Circuit to Measure Thevenin Resistance

Boot-loading the board

Arduino board is used to bootload the instrument droid circuit and the following steps were followed:

1. Connect the ICSP pins on the Instrument droid same as the Arduino board
2. Upload the ISP code to the Arduino
3. Monitor the heartbeat signal to ensure that the target was successfully boot-loaded
4. After boot loading, verify the working of D+ & D- and serial pins by dumping any simple code (such as Blinky) to the board

Arduino Code for Board under Test

```
// vrm characterizer board
#include <Wire.h>
#include <Adafruit_MCP4725.h>
#include <Adafruit_ADS1X15.h>
#include <Adafruit_NeoPixel.h>
#define PIN 2 // On Trinket or Gemma, suggest changing this to 1
#define NUMPIXELS 20 // Popular NeoPixel ring size
#define DELAYVAL 100 // Time (in milliseconds) to pause between pixels
Adafruit_ADS1115 ads;
Adafruit_MCP4725 dac;
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
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 = 5000.0 / 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 gain of 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;
const int buzzer = 3; //buzzer to arduino pin 2
void tone_gen()
{
for (int i = 0; i< 10; i++)
{
tone(buzzer, 5000); // Send 1KHz sound signal...
delay(1); // ...for 1 sec
noTone(buzzer); // Stop sound...
// delay(50); // ...for 1sec
}
}
```

```
void led_seq()
{
pixels.clear(); // Set all pixel colors to 'off'
// The first NeoPixel in a strand is #0, second is 1, all the way up
// to the count of pixels minus one.
for(int i=0; i<NUMPIXELS; i++) { // For each pixel...
// pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
// Here we're using a moderately bright green color:
pixels.setPixelColor(i, pixels.Color(50,10,50));
pixels.show(); // Send the updated pixel colors to the hardware.
delay(DELAYVAL); // Pause before next pass through loop
pixels.clear();
}
}
void func_meas_on()
{
tone_gen();
led_seq();
//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);
}
void func_meas_off()
{
noTone(buzzer); // Stop sound...
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 setup()
{
pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
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(); // 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);
ECEN 5730 Practical PCB Design and Manufacture
Board – 4 Devang Boradhara
15
University of Colorado, Boulder
Serial.print(", ");
Serial.print(V_VRM_loaded_v, 4);
Serial.print(", ");
Serial.println(R_thevenin, 4);
}
Serial.println("done");
delay(30000);
}
```

Test & Analysis of Assembled Board

- **Testing Blinky code on Instrument Droid**

  After the successful boot-loading of the Instrument Droid, blinky code was dumped on the board to test.

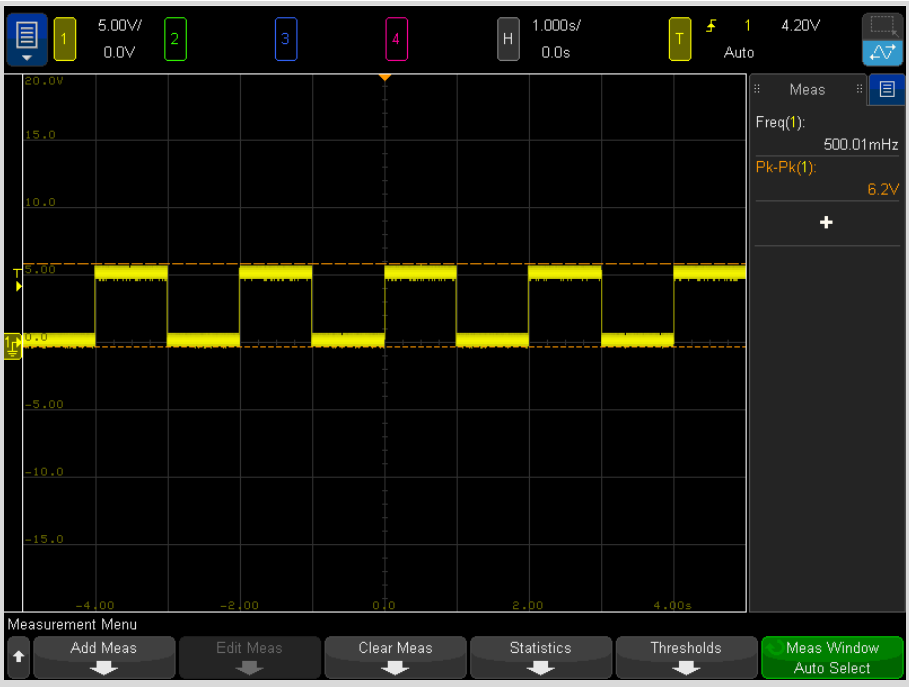


Figure 7: Switching Output of Blilnky code

From the trace above, we can observe that the switching waveform is as per the expectation of Blinky code.

5V on Adapter

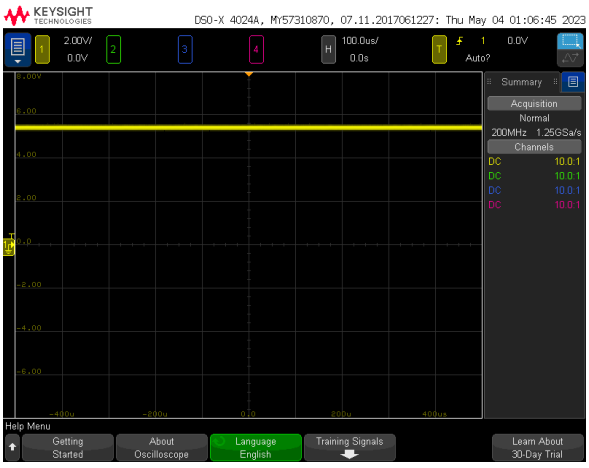


Figure 7: Output Voltage on 5V Adapter

5V on USB

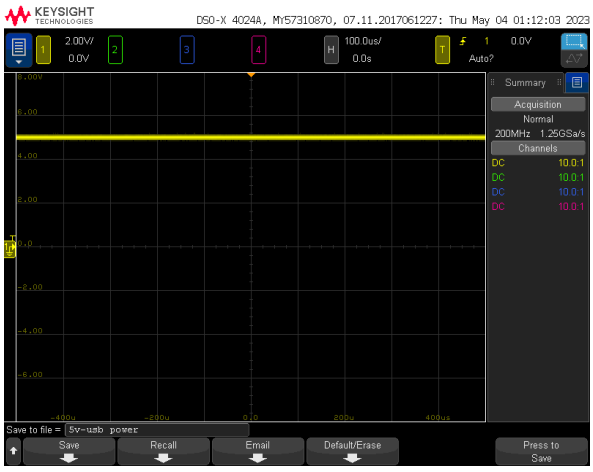


Figure 8: Output Voltage on USB

Smart LED Test

Smart LEDs were tested using the Arduino code available on the NeoPixel Library

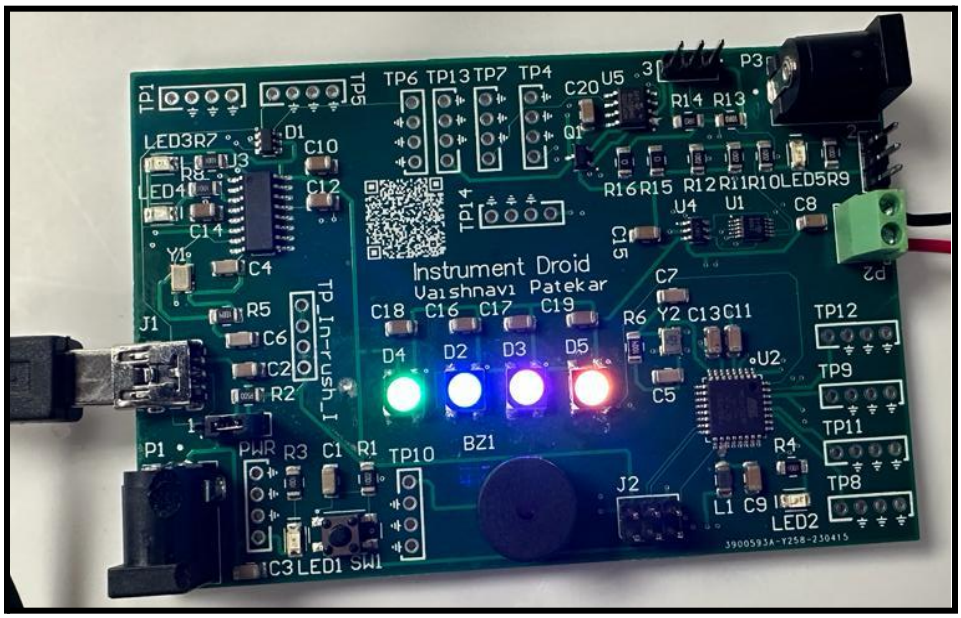


Figure 9: Smart LED test

Buzzer Test

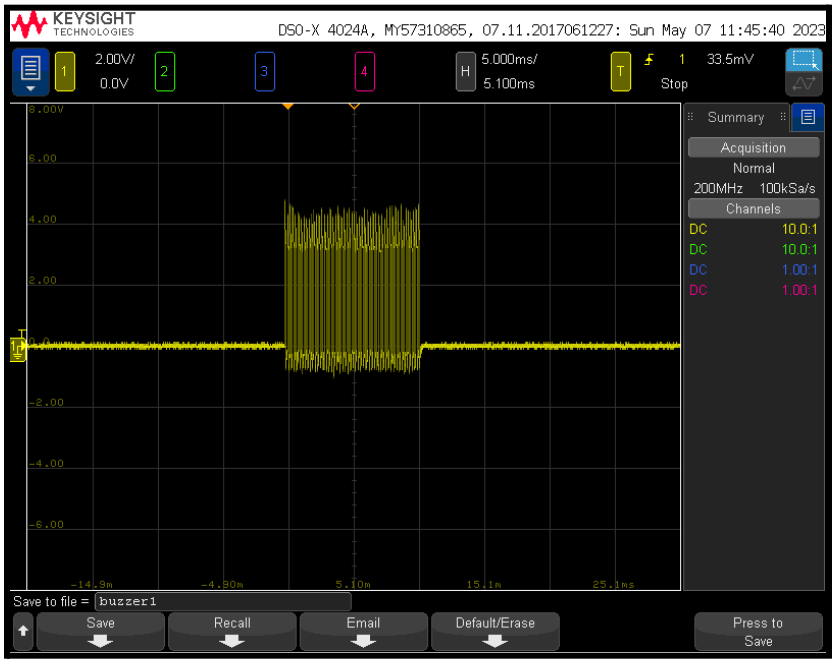


Figure 8: Output on Buzzer Test Point

DAC Module Test

The Instrument Droid Circuit's crucial component is the DAC module since it regulates the switching of the FET, which in turn makes it possible to monitor Loaded and Unloaded Voltage values. Using the Arduino code from the preceding section, where the DAC signal is steadily growing with respect to the voltage drop for each iteration.

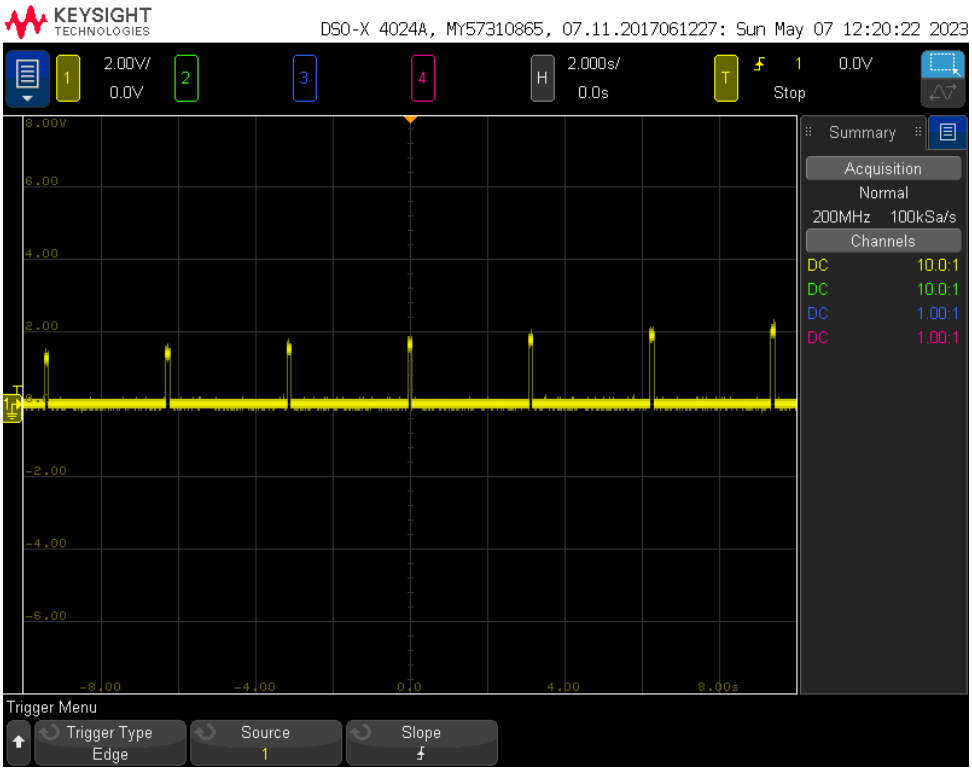


Figure 9: DAC Output controlled by the Atmega328

As can be seen in the scope above, the DAC output is gradually stepping up the input voltage to the FET.

OP-AMP Output

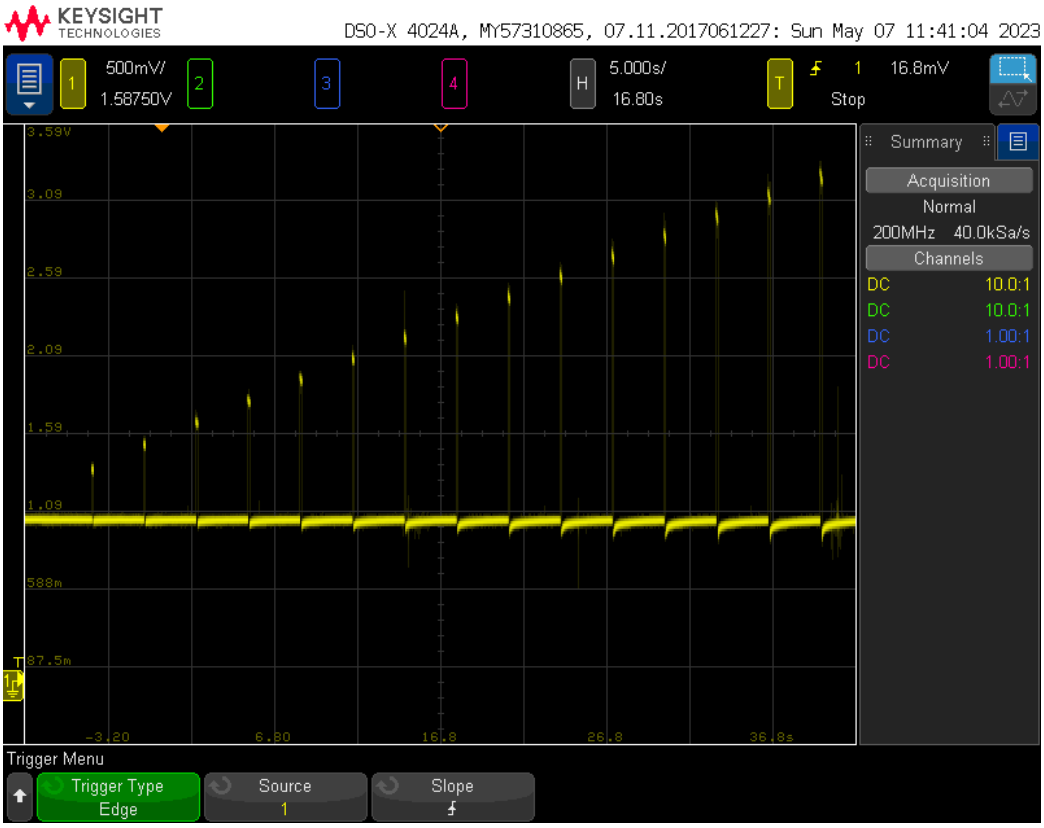


Figure 10: Op-amp Output

ADC voltage Measurement from the inputs

All measurements for a power source begin with the conversion of the Analog Input value into a digital value, which is then fed into the equations that compute Thevenin parameters. The Arduino code for the instrument droid presented above can be used again to test the ADC module and verify if it is working properly because it measures two differential-ended voltages from the Loaded and Unloaded Voltage drop locations in the circuit.

To demonstrate the module's functionality, printing the digital values converted by the ADC on the serial console/excel data streamer is a good test. When the Droid was given 9V power, the serial console displayed the following readings, proving that the ADC module works.

		Current (mA)	Vth	V Loaded	Rth
12:41:25.00	1	13.124	9.1916	9.1806	0.8399
12:41:28.12	2	25.754	9.1916	9.1694	0.86
12:41:31.26	3	38.318	9.1917	9.16	0.8269
12:41:34.39	4	50.834	9.1912	9.1502	0.807
12:41:37.52	5	63.699	9.191	9.1402	0.7965
12:41:40.65	6	76.325	9.1914	9.131	0.7914
12:41:43.78	7	88.861	9.1912	9.1195	0.8064
12:41:46.92	8	101.408	9.1911	9.1108	0.7915
12:41:50.05	9	113.927	9.1919	9.1016	0.7929
12:41:53.18	10	126.639	9.1908	9.0913	0.7857
12:41:56.31	11	139.24	9.1913	9.0821	0.7844
12:41:59.45	12	151.67	9.1909	9.0718	0.7858
12:42:02.58	13	164.184	9.1917	9.0624	0.7875
12:42:05.71	14	176.902	9.1913	9.0533	0.7796
12:42:08.84	15	189.436	9.1914	9.0428	0.7843
12:42:11.97	16	202.153	9.191	9.0325	0.7838
12:42:15.11	17	214.84	9.1914	9.0229	0.7845
12:42:18.24	18	227.363	9.1915	9.0138	0.7813
12:42:21.37	19	240.108	9.1913	9.0041	0.7796
12:42:24.50	20	252.996	9.1916	8.994	0.7812
12:42:24.50	done				

Figure 11: ADC Module Output on Serial Channel using Excel Data Streamer

From the serial plot above, the current sense resistor draws current at any point in the measurement cycle, and it increases gradually as the switch on time for the MOSFET increases with an increasing DAC output.

Also, we can observe a small difference between Vth and VLoaded that is caused by the element present at the switching end of the MOSFET. The difference in voltage values signifies the MOSFET switching on and off in response to the DAC signal.

After performing these unit tests, the following parameters can be measured using an Instrument droid:

1. Current through Sense Resistor
2. Loaded Voltage, Unloaded Voltage
3. Thevenin Resistance.

For the final test, the following two voltage sources were used:

1. 9V Power Adapter
2. Function Generator with constant 5V supply

1. 9V Power Adapter

		Current (mA)	Vth	V Loaded	Rth
12:41:25.00	1	13.124	9.1916	9.1806	0.8399
12:41:28.12	2	25.754	9.1916	9.1694	0.86
12:41:31.26	3	38.318	9.1917	9.16	0.8269
12:41:34.39	4	50.834	9.1912	9.1502	0.807
12:41:37.52	5	63.699	9.191	9.1402	0.7965
12:41:40.65	6	76.325	9.1914	9.131	0.7914
12:41:43.78	7	88.861	9.1912	9.1195	0.8064
12:41:46.92	8	101.408	9.1911	9.1108	0.7915
12:41:50.05	9	113.927	9.1919	9.1016	0.7929
12:41:53.18	10	126.639	9.1908	9.0913	0.7857
12:41:56.31	11	139.24	9.1913	9.0821	0.7844
12:41:59.45	12	151.67	9.1909	9.0718	0.7858
12:42:02.58	13	164.184	9.1917	9.0624	0.7875
12:42:05.71	14	176.902	9.1913	9.0533	0.7796
12:42:08.84	15	189.436	9.1914	9.0428	0.7843
12:42:11.97	16	202.153	9.191	9.0325	0.7838
12:42:15.11	17	214.84	9.1914	9.0229	0.7845
12:42:18.24	18	227.363	9.1915	9.0138	0.7813
12:42:21.37	19	240.108	9.1913	9.0041	0.7796
12:42:24.50	20	252.996	9.1916	8.994	0.7812
12:42:24.50	done				

Figure 12: Instrument Droid Results for 9V Adapter Input

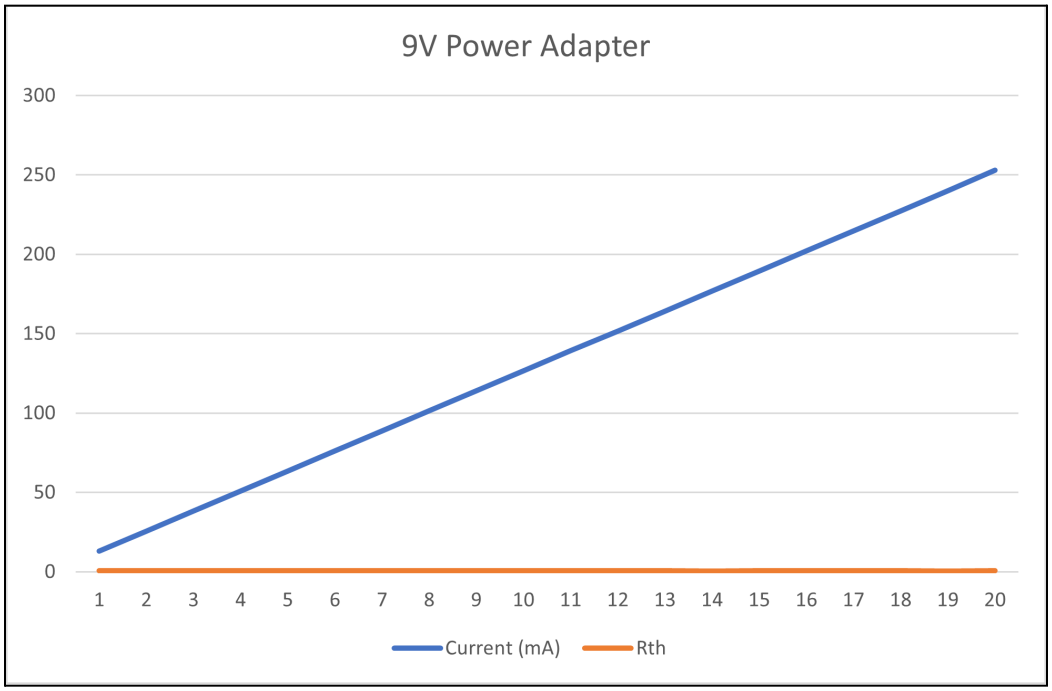


Figure 13: Excel Plot for 9V Adapter Input

The Excel plot and serial data above indicate that the instrument droid performs well since it produces results that meet expectations.

It would be interesting to see how the droid behaves with a Power Source that has a substantially higher Thevenin Resistance; the following test would aid in determining whether the Instrument Droid can achieve this when we try to assess the Thevenin Resistance of a function generator.

2. 5V Function Generator Input to the Instrument Droid Board

		Current (mA)	Vth	V Loaded	Rth
12:55:14.00	1	13.12	4.9957	4.3311	50.6512
12:55:17.14	2	25.761	4.9957	3.6923	50.5931
12:55:20.27	3	38.335	4.9957	3.0555	50.6116
12:55:23.40	4	50.866	4.9957	2.422	50.5982
12:55:26.53	5	63.718	4.9957	1.7704	50.618
12:55:29.67	6	76.349	4.9957	1.1326	50.5983
12:55:32.80	7	82.29	4.9958	0.8317	50.6032
12:55:35.93	8	82.289	4.9957	0.8318	50.6017
12:55:39.06	9	82.288	4.9957	0.8317	50.6024
12:55:42.19	10	82.288	4.9957	0.8317	50.6025
12:55:45.33	11	82.287	4.9957	0.8317	50.6033
12:55:48.46	12	82.288	4.9957	0.8317	50.6024
12:55:51.59	13	82.289	4.9957	0.8317	50.6029
12:55:54.72	14	82.283	4.9957	0.8318	50.6044
12:55:57.86	15	82.287	4.9957	0.8318	50.6023
12:56:00.99	16	82.288	4.9957	0.8317	50.6022
12:56:04.12	17	82.289	4.9958	0.8318	50.6017
12:56:07.25	18	82.29	4.9957	0.8318	50.6003
12:56:10.38	19	82.288	4.9957	0.8318	50.6016
12:56:13.52	20	82.29	4.9957	0.8318	50.6011
12:56:13.52	done				

Figure 14: Instrument Droid Results for 5V Function Generator Input

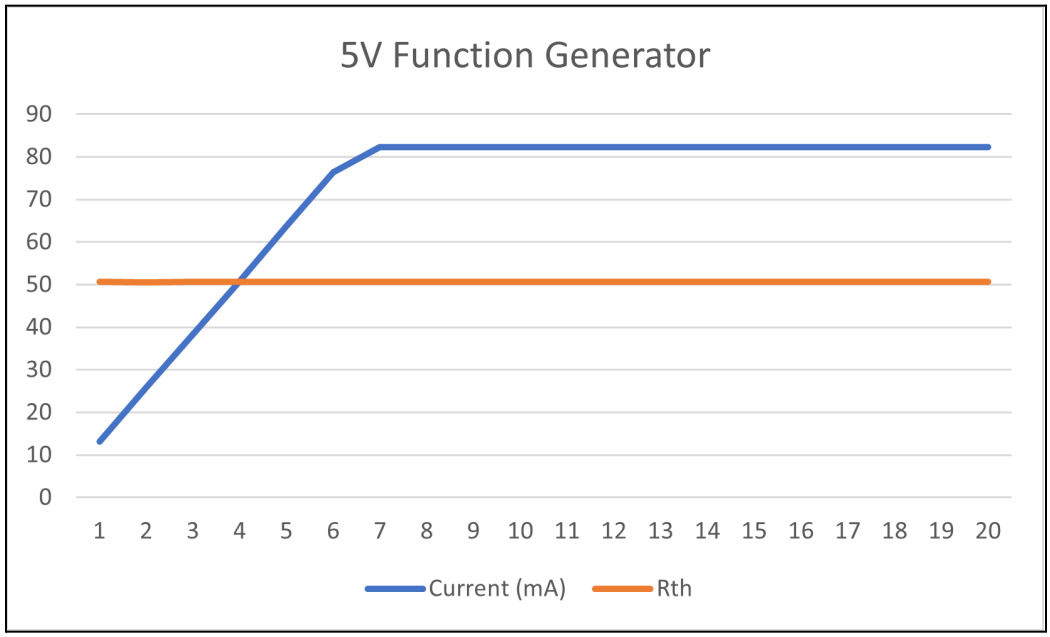


Figure 15: Excel Plot for 5V Function Generator Input

The Serial Data through Excel Data streamer and the Excel plot shows that the Instrument Droid measures the correct Thevenin Resistance, as it gives a Thevenin Resistance of 50 Ohms for the 5V function generator input, which is exactly what was expected prior to performing the measurement. It is worth noting that in the initial version of the Arduino Code, there was a threshold check to determine if the Voltage across Rsense fell below 75% of the input voltage, the ramping of the DAC value would halt, and the number of points of increments would be reduced.

What worked after assembly and testing?

The list of features and capabilities that were anticipated to operate on the final-built Instrument Droid Board is given below. These belonged to POR and were verified using various test points on the board and from the peripherals of the IC for the pins without a specific test point.

Sr. No.	Function	Working Status
1.	Power Rail: 5V & 3.3V	Yes
2.	Power Select (Adapter/USB)	Yes
3.	Oscillators - 12Mhz & 16Mhz	Yes
4.	USB detection on host PC	Yes

5.	Boot loading	Yes
6.	Dumping Blinky Code through IDE over USB	Yes
7.	Reset Circuitry	Yes
8.	TVS for ESD Protection	Yes
9.	Buzzer	Yes
10.	Smart LEDs	Yes
11.	On board droid circuit to measure Rth	Yes

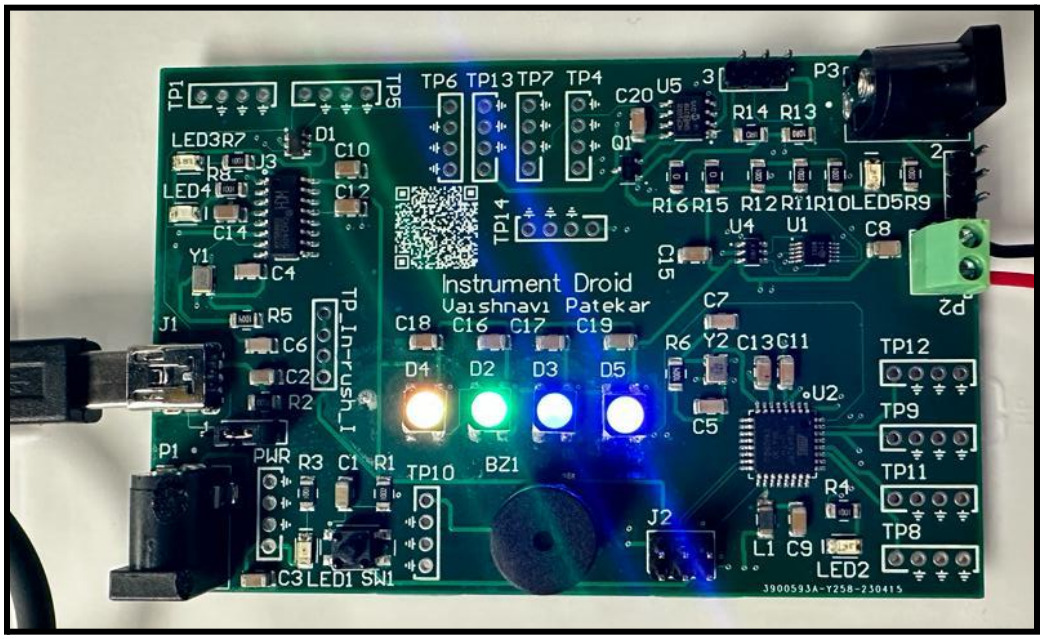
Figure 16: List of functions/features that worked

After a power connection to USB and correcting the Tx, and Rx connections, the board worked correctly as per the expectations and was successfully boot-loaded and programmed, fulfilling all the necessary requirements.

Mistakes Made & Final Working Board Image

There were no hard mistakes and the board worked as expected. However, the labeling of the test points could have improved. I had given labels to the test points as TP1, Tp2, etc, the exact naming like TP_Tx would have been easy for readability.

Working Board Image:



Key Learnings

- This design project aided in comprehending the additive routing advantages of a 4-layer board, in addition, to return planes.
- This project provided valuable experience in the routing and placement of several components on a 4-layer PCB.
- Analog value differential measurement aids in reducing common ground noise on the circuit.
- The quality of the circuit can be greatly impacted by a few straightforward design principles including the use of a common return ground plane, the positioning of decoupling capacitors, loop inductance, etc.
- When creating circuits like an instrument droid, MOSFET is a better option than a transistor because doing so would have led to additive current measurement due to the base current passing through the RSense.

References

- *CU ECEN-4/5730 Spring 2023 Workbook by Prof. Eric Bogatin*
- *Bogatin's Practical Guide to Prototype Breadboard and PCB Design by Eric Bogatin, published by Artech House Copyright: 2021 ISBN: 9781630818487*