Practical PCB Design and Manufacture
Fall 2024-ECEN 5730
Sepideh Mohammadi

# Board4 report: Instrument Droid

## Objective

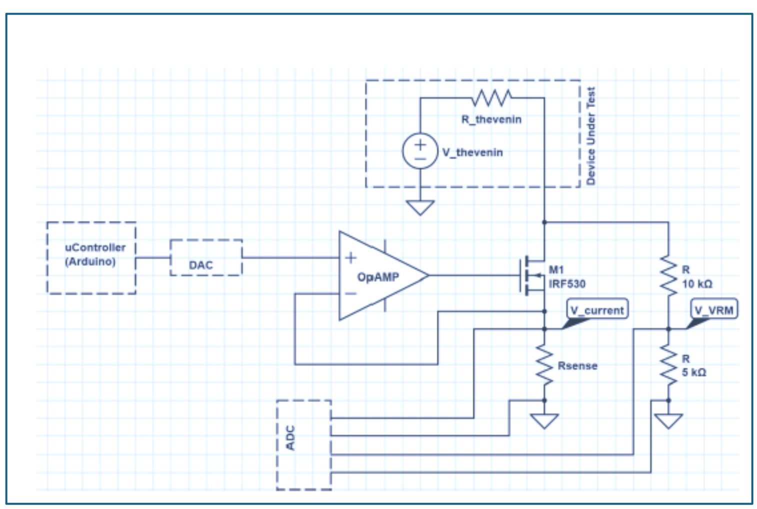
To design a board that is capable of measuring Rthevenin of any voltage supply up to 12V automatically based on the Atmega328 microcontroller.

## Plan of Record (POR)

The circuit should have,

1. A power plug for “AC to 5V DC adaptor” to supply power for the board.
2. A power plug for” measuring Rthevenin wall wart”, a connector to plug in the power supply using jumpers, and a USB C connector.
3. An OLED display for showing the Rthevenin.
4. Additionally, 3 smart LEDs and a buzzer as aesthetic features.
5. A USB mini-B with TVS diode for connecting to PC.
6. A MOSFET, Op-Amp, ADC, and DAC
7. A switch to select between 5V/USB supply.
8. ICSP header pin for bootloading Atmega 328.
9. Atmega 328 microcontroller and CH340g for USB to UART.
10. 0.5Ω sense resistor and a test point to measure inrush current
11. A reset circuitry
12. A ferrite filter on AVCC pin.
13. Crystal:16MHz for Atmega328 and 12MHz for CH340g
14. Indicator LED for the power supply and VRM
15. Test points for 5V rail, I2C, UART, USB (D+, D-)
16. Isolation header pins for Atmega 328
17. A switch for selecting Rsense 1/10 ohm at the source of the MOSFET

## Napkin Sketch:

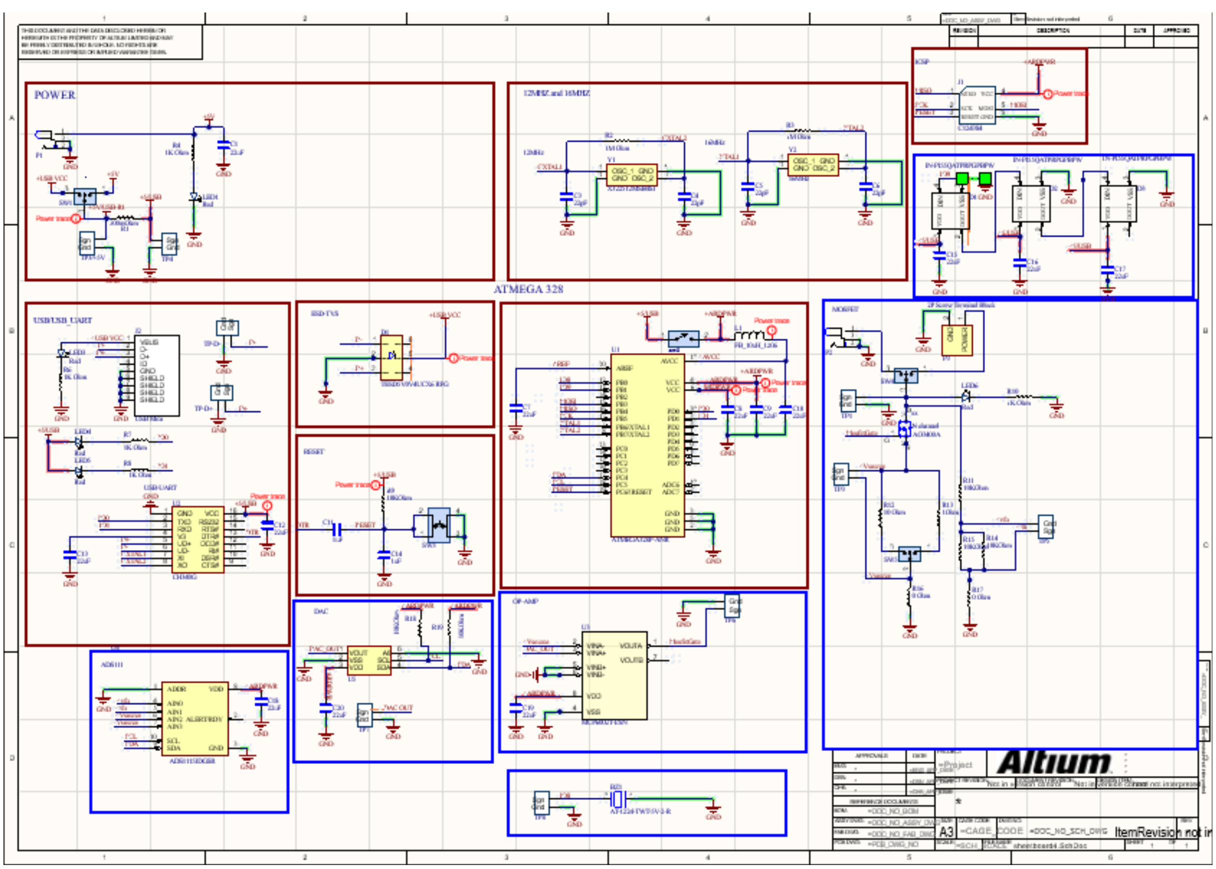


**Components Required/ BOM:**

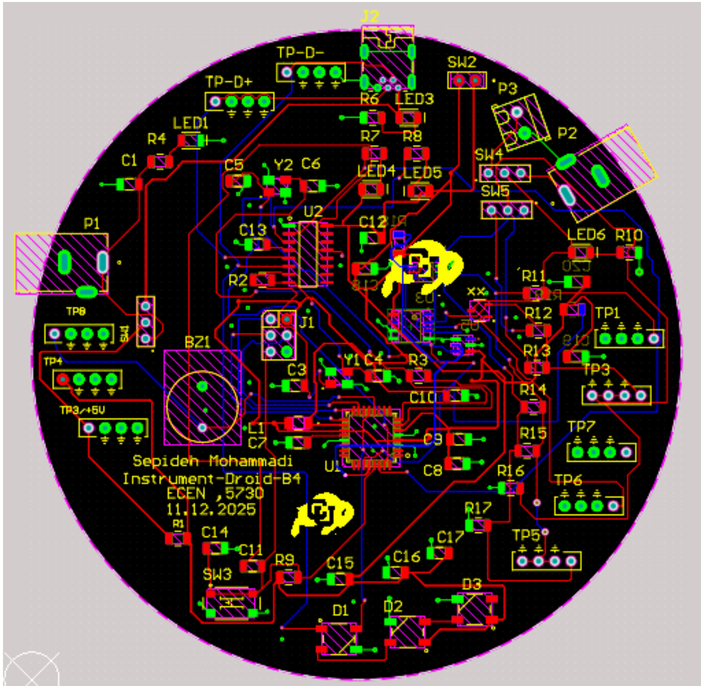
| Comment | Description (Short) | Designator | Footprint | LibRef | Qty |
|---|---|---|---|---|---|
| AT-1224-TWT-5V | 5V buzzer | BZ1 | BUZZER-TH | BUZZER_1 | 1 |
| 22uF | Capacitor 22µF 1206 | C1, C7–C10, C12–C20 | 1206_Passive_Capacitor | C_22uF_1206 | 13 |
| 22pF | Capacitor 22pF 1206 | C3–C6 | 1206_Passive_Capacitor | C_22pF_1206 | 4 |
| 1uF | Capacitor 1µF 1206 | C11, C14 | 1206_Passive_Capacitor | C_1uF_1206 | 2 |
| IN-PI55Q | RGB LED SMD | D1–D3 | SERIAL_LED | LED_Smart | 3 |
| NRPN032PAEN-RC | ICSP header 6-pin | J1 | ICSP | J_ICSP | 1 |
| USB Mini | USB Mini-B | J2 | USB_MINI_B | J_USB_B_Mini | 1 |
| FB_10uH_1206 | Inductor 10µH 1206 | L1 | AIML-1206 | FB_10uH_1206 | 1 |
| Red LED | LED Red 1206 | LED1, LED3–LED6 | LED_1206 | LED_RED_1206 | 5 |
| Power Jack | Barrel jack | P1, P2 | Power Jack | P_Power_Jack | 2 |
| 2P Terminal Block | Terminal block | P3 | TerminalBlock_3.5mm | P_TerminalBlock | 1 |
| 500mΩ | Resistor 500mΩ | R1 | 1206_Passive_Resistor | R_500mOhm_1206 | 1 |
| 1MΩ | Resistor 1MΩ | R2, R3 | 1206_Passive_Resistor | R_1M_1206 | 2 |
| 1kΩ | Resistor 1kΩ | R4, R6–R8, R10 | 1206_Passive_Resistor | R_1K_1206 | 5 |
| 10kΩ | Resistor 10kΩ | R9, R11, R14, R15, R18, R19 | 1206_Passive_Resistor | R_10K_1206 | 6 |
| 10Ω | Resistor 10Ω | R12 | 1206_Passive_Resistor | R_10Ohm_1206 | 1 |
| 1Ω | Resistor 1Ω | R13 | 1206_Passive_Resistor | R_1Ohm_1206 | 1 |
| 0Ω | Jumper resistor | — | 1206_Passive_Resistor | R_0Ohm_1206 | 2 |
| PREC003SAAN | 3-pin switch header | SW1, SW4, SW5 | 3_PIN_100mil | SW_3Pin_100mil_Switch | 3 |
| 2-pin switch header | 2-pin header | SW2 | 2Pin_Header | SW_2Pin_100mil_Switch | 1 |

| | | | | | |
|---|---|---|---|---|---|
| TL3305AF160QG | Tactile switch SMD | SW3 | Switch_JLC | SW_Push_Button_JLC | 1 |
| 10x Probe TP | Test points | TP1, TP3, TP3/+5V, TP4–TP8, TP-D-, TP-D+ | TP10x_Probe | TP_10x_Probe | 10 |
| ATMEGA328P-ANR | Microcontroller | U1 | ATMega328P | U_ATMega328P | 1 |
| CH340G | USB-serial IC | U2 | USB_to_serial | U_USBtoSerial | 1 |
| MCP6002T-I/SN | Dual op-amp | U3 | SOIC-8 | U_MCP6002T-I/SN | 1 |
| ADS1115IDGSR | 16-bit ADC | U4 | VSSOP-10 | U_ADC_ADS1115IDGSR | 1 |
| MCP4725 | 12-bit DAC | U5 | SOT23-6 | U_MCP4725A0T-E/CH | 1 |
| AO3400A | N-MOSFET | Qx | N-MOSFET | Q_Nch_MOSFET | 1 |
| X322512MSB4SI | 12 MHz crystal | Y1 | SMD-3225 | X_12MHz_Res | 1 |
| X322516MLB4SI | 16 MHz crystal | Y2 | SMD-3225 | X_16MHz_Res | 1 |

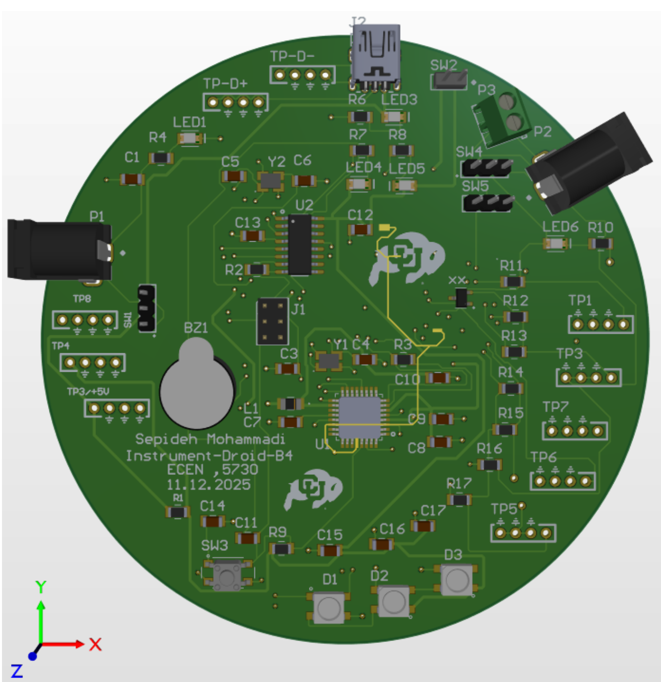
**Schematic:** **Instrument Droid Schematic in Altium software**

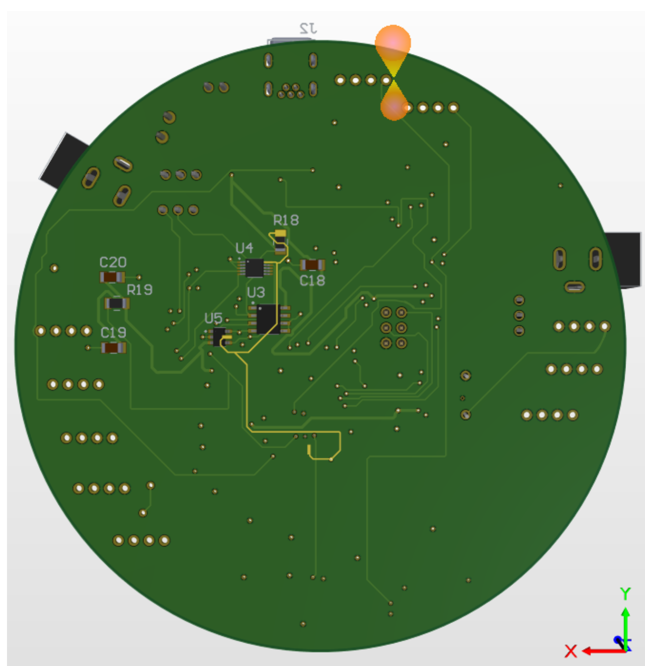


**Layout:** **Instrument layout done in Altium Software**

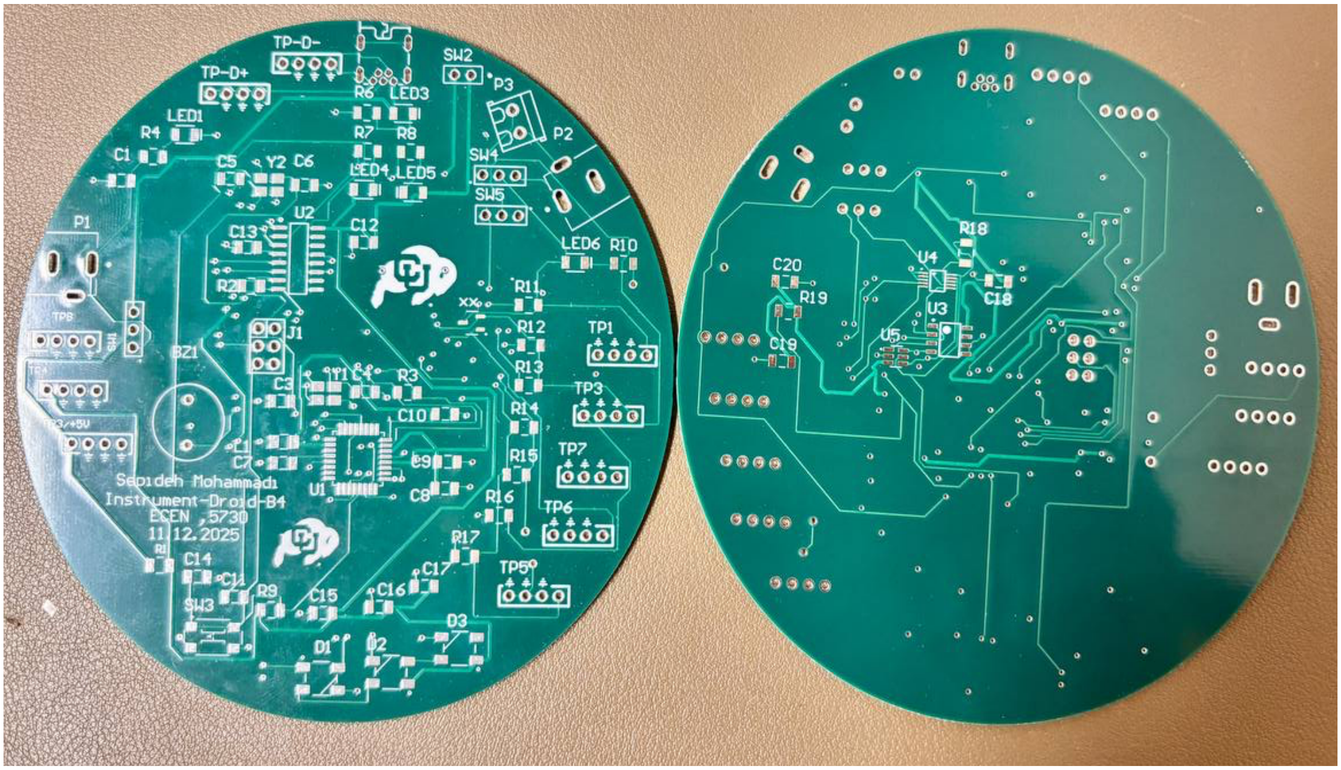



**3D View: front view and back view**

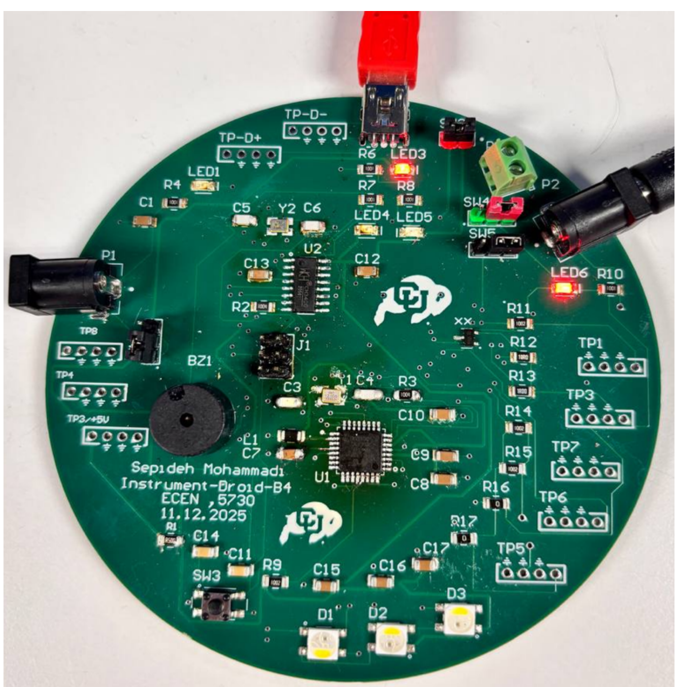


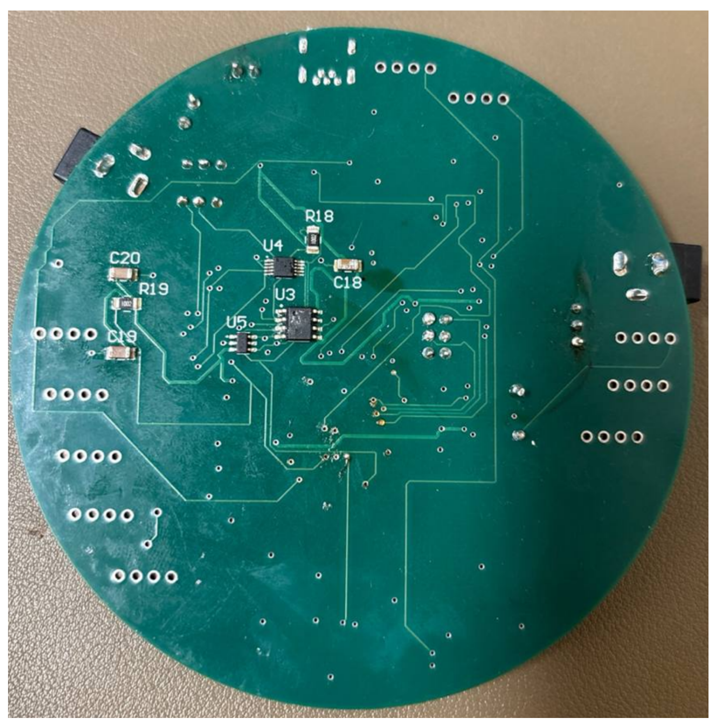



**Before assembling:** **Unassembled from front / Unassembled board-back**




**After assembling: Assembled board-front / Assembled board-back**

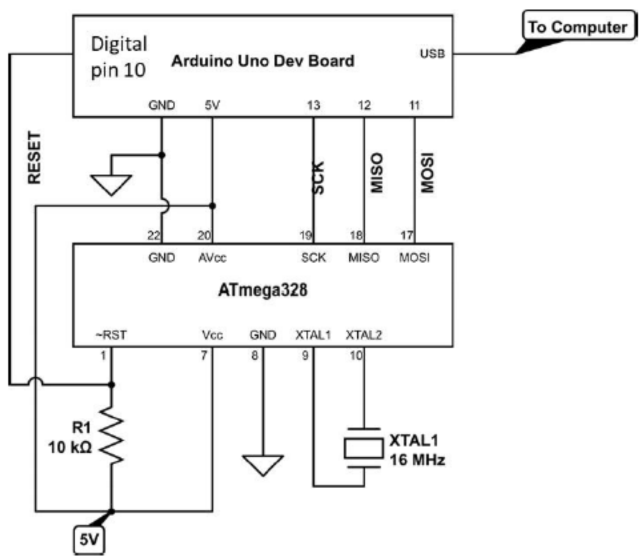


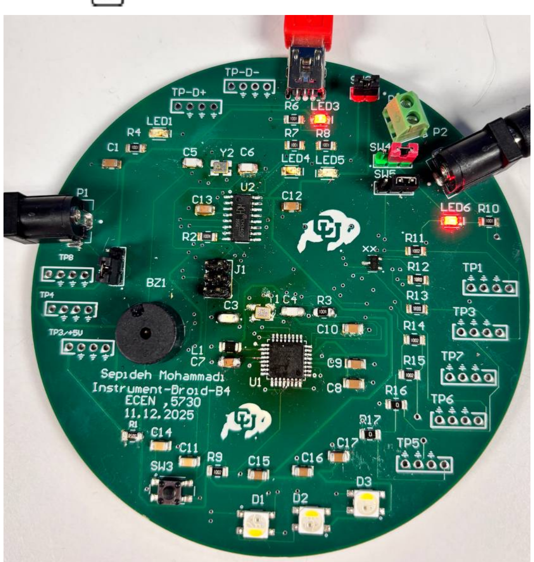
**What it means to work**

1. Indicator LED1 should turn ON and if VRM is connected, LED4 should be ON.
2. The board can be powered with a 5 V AC to DC adapter /USB which can be measured at TP3/+5V
3. Using a commercial Arduino UNO programmed as ISP, the Atmega can be boot loaded to turn it into Arduino UNO using ICSP/ SPI header pins
4. After bootloading, 16MHz crystal should resonate at 16MHz and it should run codes uploaded from IDE.
5. It should be able to measure Rthevenin of any power supply up to 12V after uploading the appropriate code.

**Bootloader procedure:**




-Connect SCK of ICSP to D13 of UNO
-MISO of ICSP to D12 of UNO
-MOSI of ICSP to D11 of UNO
-VCC and gnd of ICSP to 5V header and gnd socket of UNO
-Reset header pin of ICSP to D10 of UNO
-After this is done, connect commercial Arduino to PC and go to
-File>examples>11(Arduino ISP), upload it.
- Tools>programmer (Arduino as ISP) and make sure port and board is selected appropriately. Then click Burn bootloader.

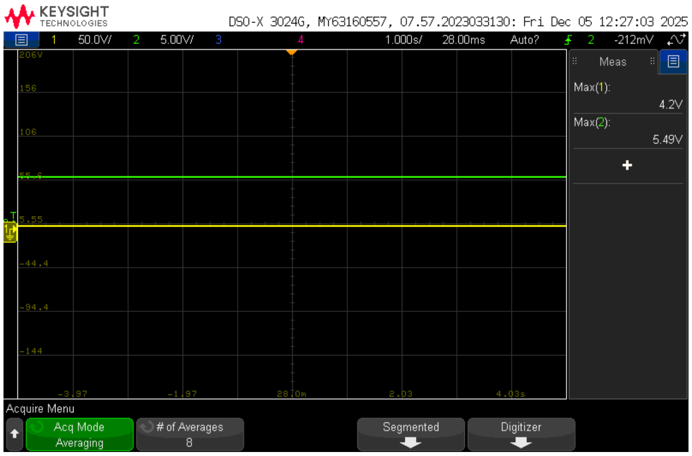



**What worked/ Bring-up:**
All the anticipated LEDs turned ON, the ATMEGA 328 is boot-loaded successfully using ICSP. After uploading the appropriate code, it was able to measure Rthevenin.

**What did not work?**
Everything worked as expected and there were no Hard/soft errors.

## Scope shots and Analysis

1. Figure 1 indicates 5V (Yellow) and 5V -5VVRM (Green) measured at TP3/+5V and TP5 respectively. Figure 2 shows DAC signal when the code is running.



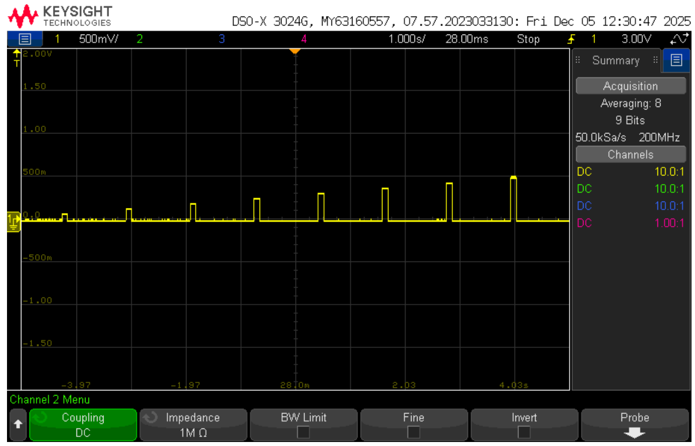
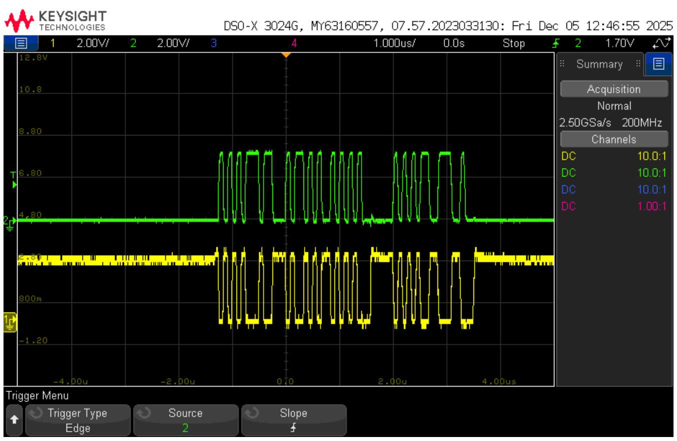




**Figure 1 Power Rails (5V and 5- VRM).**

**Figure 2 DAC signal**

2.USB signals are sniffed from TP-D+, and TP-D-respectively when the code is uploaded. It is captured using Normal trigger mode. I2C and RESET signals: I2C and RESET signals are sniffed from TP6 and directly from the bottom for the reset and ICSP respectively. It is captured using Normal trigger mode.



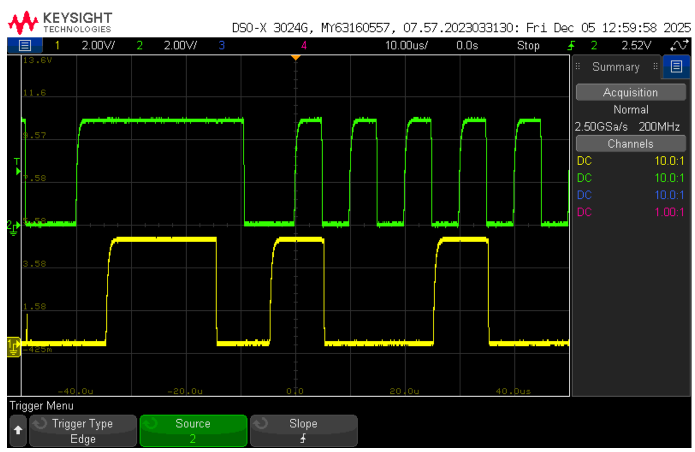




**Figure 3 USB signals, yellow – D-, Green- D+**

**Figure 4 I2C signal, yellow - SCL, green-SDA**

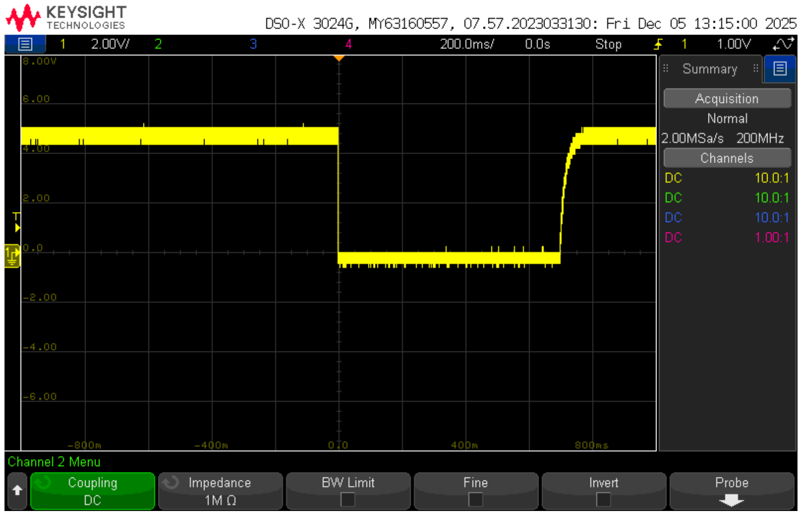



**Figure 5: RESET signal when switch Is pressed. I measured it directly from the push bottom.**

**3.**Inrush current: With normal trigger mode, the inrush current sunk by the decoupling capacitors is captured across 0.5Ω. In Figure 19, Yellow is the high side of the resistor, green is the low side, and pink is the math function that subtracts the high side from the low side. So, Inrush current= 2.57/0.5 = 5.14A!

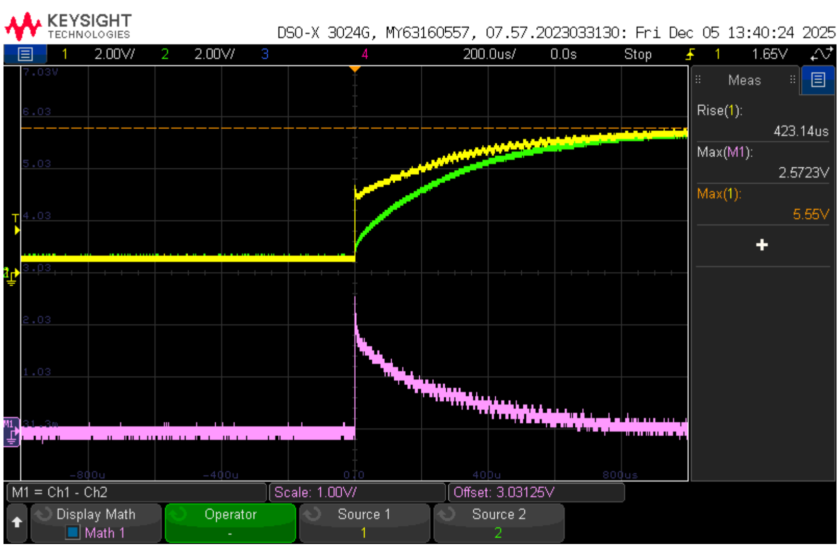
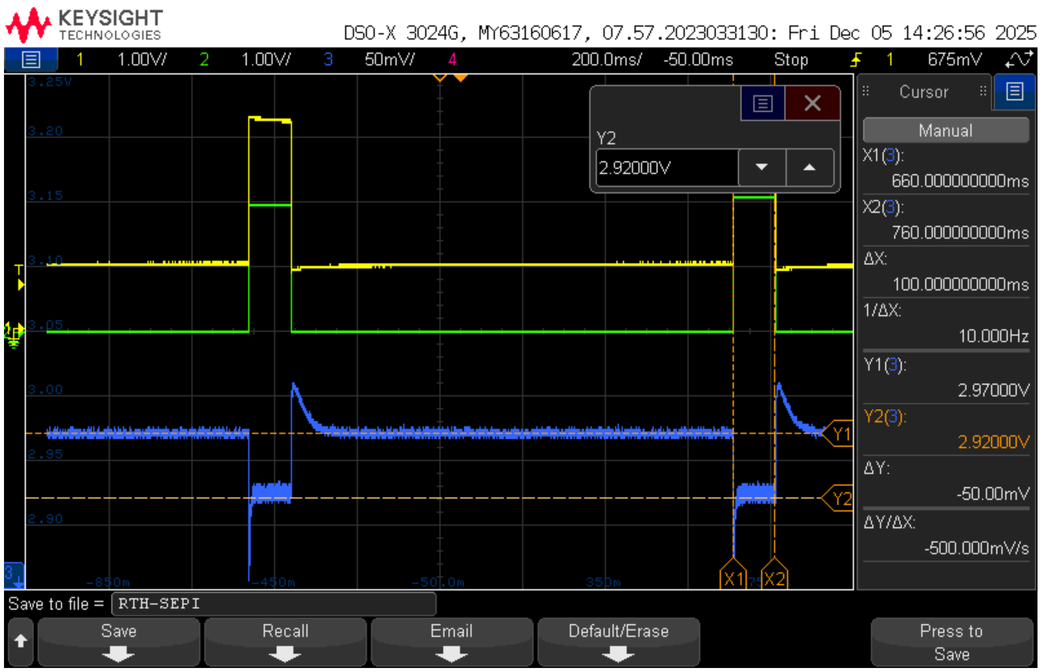



**Figure 6: Inrush Current**

1. GATE, Vth, Vload signals for Thevenin Resistance: This is an instance where Gate signal, Vth and Vlaod are captured at test points, when a **5v wall wart is connected to VRM.**




## Rth Calculation (Using New Scope Data)

We calculate the source output resistance using:

$$R_{th} = \frac{V_{th} - V_{load}}{I_{load}}$$

### 1. Read voltages from the oscilloscope

From the new image:

- **Before dip (blue trace):**

$$V_{\text{meas,before}} = 2.97\text{ V}$$

- **During dip (blue trace):**

$$V_{\text{meas,dip}} = 2.92\text{ V}$$

### 2. Convert measured voltage to actual voltage

The blue signal is measured **after a voltage divider**:
10 kΩ (top) and 5 kΩ (bottom)

Divider ratio:

$$\frac{5k}{10k + 5k} = \frac{1}{3}$$

So, the real voltage is:

$$V_{\text{actual}} = 3 \times V_{\text{measured}}$$

**Actual Vth (before dip)**

$$V_{th} = 2.97 \times 3 = 8.91\ \text{V}$$

**Actual Vload (during dip)**

$$V_{load} = 2.92 \times 3 = 8.76\ \text{V}$$

**3. Load current calculation**

Green trace shows **1.5 V across a 10 Ω resistor:**

$$I_{load} = \frac{1.5}{10} = 0.15\ \text{A}$$

So, the load current is:

$$I_{load} = 150\ \text{mA}$$

4**. Compute Rth**

$$R_{th} = \frac{V_{th} - V_{load}}{I_{load}}$$

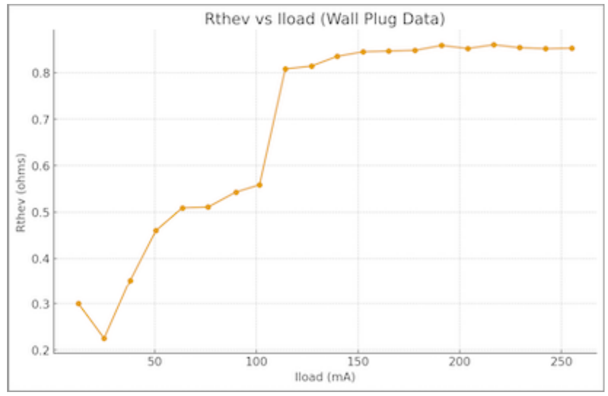
$$R_{th} = \frac{8.91 - 8.76}{0.15}$$

$$R_{th} = \frac{0.15}{0.15} = 1\Omega$$

The following values are copied from serial monitor of Arduino IDE to Excel for analysis. The graphs show the non-linear behavior of power supplies.

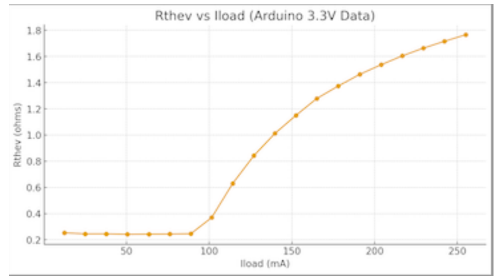
**With 5V wall wart:** 0.2 ohms

| Index | Iload (mA) | Vth (V) | Vload (V) | Rthev (Ω) |
|---|---|---|---|---|
| 1 | 12.387 | 5.3529 | 5.3492 | 0.3002 |
| 2 | 25.066 | 5.3532 | 5.3476 | 0.2251 |
| 3 | 37.853 | 5.3525 | 5.3391 | 0.3521 |
| 4 | 50.566 | 5.3530 | 5.3297 | 0.4602 |
| 5 | 63.537 | 5.3526 | 5.3208 | 0.5091 |
| 6 | 76.250 | 5.3527 | 5.3137 | 0.5108 |
| 7 | 89.903 | 5.3524 | 5.3043 | 0.5431 |
| 8 | 101.543 | 5.3524 | 5.2957 | 0.5588 |
| 9 | 114.199 | 5.3527 | 5.2604 | 0.8082 |
| 10 | 127.073 | 5.3528 | 5.2493 | 0.8144 |
| 11 | 139.812 | 5.3530 | 5.2362 | 0.8354 |
| 12 | 152.398 | 5.3526 | 5.2238 | 0.8450 |
| 13 | 165.057 | 5.3521 | 5.2089 | 0.8467 |
| 14 | 178.050 | 5.3527 | 5.2017 | 0.8482 |
| 15 | 191.024 | 5.3521 | 5.1898 | 0.8588 |
| 16 | 203.908 | 5.3526 | 5.1789 | 0.8522 |
| 17 | 216.738 | 5.3529 | 5.1664 | 0.8603 |
| 18 | 229.426 | 5.3529 | 5.1570 | 0.8540 |
| 19 | 242.243 | 5.3524 | 5.1461 | 0.8518 |
| 20 | 255.091 | 5.3528 | 5.1352 | 0.8530 |



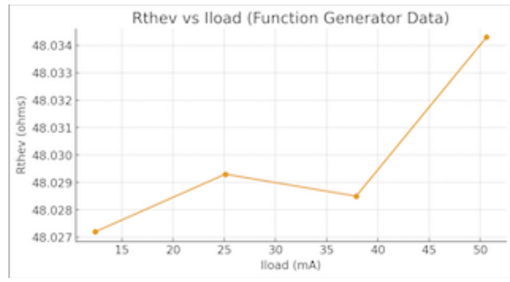
**With 3.3V Arduino Data Table**

| Index | Iload (mA) | Vth (V) | Vload (V) | Rthev (Ω) |
|---|---|---|---|---|
| 1 | 12.389 | 3.3593 | 3.3562 | 0.2530 |
| 2 | 25.070 | 3.3593 | 3.3532 | 0.2450 |
| 3 | 37.850 | 3.3593 | 3.3500 | 0.2454 |
| 4 | 50.563 | 3.3593 | 3.3470 | 0.2420 |
| 5 | 63.537 | 3.3593 | 3.3438 | 0.2434 |
| 6 | 76.250 | 3.3593 | 3.3407 | 0.2441 |
| 7 | 88.882 | 3.3593 | 3.3374 | 0.2460 |
| 8 | 101.529 | 3.3592 | 3.3217 | 0.3699 |
| 9 | 114.185 | 3.3593 | 3.2874 | 0.6296 |
| 10 | 127.055 | 3.3593 | 3.2522 | 0.8431 |
| 11 | 139.800 | 3.3593 | 3.2178 | 1.0127 |
| 12 | 152.390 | 3.3593 | 3.1841 | 1.1493 |
| 13 | 165.045 | 3.3593 | 3.1472 | 1.2790 |
| 14 | 178.038 | 3.3594 | 3.1148 | 1.3740 |
| 15 | 191.014 | 3.3593 | 3.0800 | 1.4633 |
| 16 | 203.912 | 3.3594 | 3.0461 | 1.5367 |
| 17 | 216.745 | 3.3594 | 3.0116 | 1.6045 |
| 18 | 229.424 | 3.3594 | 2.9778 | 1.6633 |
| 19 | 242.225 | 3.3594 | 2.9438 | 1.7159 |
| 20 | 255.080 | 3.3595 | 2.9091 | 1.7658 |



**Function Generator Table: With Function generator using jumper (@P2 ref designator)**: 50 ohms

| Index | Iload (mA) | Vth (V) | Vload (V) | Rthev (Ω) |
|---|---|---|---|---|
| 1 | 12.396 | 9.6039 | 9.0085 | 48.0272 |
| 2 | 25.089 | 9.6040 | 8.3990 | 48.0293 |
| 3 | 37.880 | 9.6040 | 7.7847 | 48.0285 |
| 20 | 50.615 | 9.6039 | 7.1727 | 48.0343 |




**Best design practices followed**:

1. During component selection, unique parts were reduced.
2. The decoupling capacitor is placed near the ICs, and a ferrite bead is used for the AVCC pin which is more sensitive to noises.
3. 3 X 3 inches board dimensions are used which has the lower cost at JLC PCB fabrication vendor
4. Test points were labeled to reduce soft errors.
5. Indicator LED and isolation switches were included for faster debugging/ troubleshooting.
6. Continuous return planes are used in layer 2 and layer 3.
7. Signal transitions between layers were surrounded by stitching ground vias for signal integrity.
8. 20 mil width is used for power traces (ensures high current carrying capacity) and 6 mil width for all other traces (ensures highest routing density with lowest cost allowed by the fabrication vendor).

**Things could be improved.**

1. The board can be made smaller by making a custom footprint for test points
2. OLED has pins in the following order: VCC, Gnd, SCL, SDA. But most of the stock available online is of the order: Gnd, VCC, SCL, SDA. So, this order should be followed for the next implementation of OLED.

## Conclusion

The designed instrument droid is successfully able to measure Rthevenin of the power supplies which is rare data to be found in the datasheet. This is important as it characterizes the first-order model of the power supply.

```cpp
// vrm characterizer board
#include <Wire.h>
#include <Adafruit_MCP4725.h>
#include <Adafruit_ADS1X15.h>
Adafruit_MCP4725 dac;
// OLED
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
//#include <Fonts/FreeSerif9pt7b.h>
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
// LED
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif

#define LEDPIN 9
Adafruit_NeoPixel pixels(3, LEDPIN, NEO_GRB + NEO_KHZ800);
// buzzer
float R_sense = 10.0; // current sense resistor
long itime_on_msec = 100;   // time for taking measurements
long itime_off_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 division on VRM measurement
int t_step_iA = 1; // ramp step of DAC
int I_A = 0;       // current in A (ADU to A conversion)
float V_RM_on_v, V_RM_off_v;
float I_sense_on_A, I_sense_off_A;
float R_thevenin;
int npts = 20;     // number of measurement steps
// values for converting ADC readings
float ADC_V_per_ADU = 1.0 / 1875.0;   // ADS1115 in GAIN_TWOTHIRDS
float ADC_V_per_ADU_2 = 1.0 / 1875.0;
```

```cpp
// SETUP
//------------------------------------------------------------------
void setup() {
  Serial.begin(115200);
  pinMode(8, OUTPUT);   // buzzer
  pixels.begin();
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println("SSD1306 allocation failed");
    for (;;);}
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(1, 8);
  display.println("Instrument");
  display.setCursor(0, 40);
  display.println("Measures Rthev");
  display.display();
  delay(1000);
  display.clearDisplay();
  display.display();
  dac.begin(0x60);  // MCP4725 default address
  dac.setVoltage(0, false); // start at 0 output
  Adafruit_ADS1115 ads;
  ads.setGain(GAIN_TWOTHIRDS); // gain setting
  ads.begin();
  ads.setDataRate(RATE_ADS1115_860SPS);}
// MAIN LOOP
//------------------------------------------------------------------
void loop() {
  pixels.clear();
  pixels.show();
  noTone(8);
  for (int i = 1; i <= npts; i++) {
    display.clearDisplay();
    I_A = i * t_step_iA;
    dac.setVoltage(I_A, false); // output current step
    // MEASURE WITH LOAD ON
    func_meas_on();
    // MEASURE WITH LOAD OFF
```

```cpp
    Serial.print(", ");
    Serial.print(V_RM_loaded_v, 4);
    Serial.print(", ");
    Serial.println(R_thevenin, 4);
    //-------------------------------------------------------
    // OLED Output
    //-------------------------------------------------------
    display.setCursor(0, 0);
    display.setTextSize(1);
    display.print("index: ");
    display.print(i);
    display.display();
    display.setCursor(0, 10);
    display.print("I=");
    display.print(I_load_A * 1000.0, 1);
    display.print(" mA");
    display.display();
    display.setCursor(0, 20);
    display.print("Vth=");
    display.print(V_RM_thevenin_v, 4);
    display.print(" V");
    display.display();
    display.setCursor(0, 30);
    display.print("Vload=");
    display.print(V_RM_loaded_v, 4);
    display.print(" V");
    display.display();
    display.setCursor(0, 40);
    display.print("Rth=");
    display.print(R_thevenin, 3);
    display.print(" ohms");
    display.display();
    tone(8, 1000, 50);
    pixels.clear();
    pixels.setPixelColor((i % 3), pixels.Color(255, 50, 0));
    pixels.show();
  }
```

```cpp
  Serial.println("done!");
}
//------------------------------------------------------------------
// MEASUREMENT FUNCTIONS
//------------------------------------------------------------------
void func_meas_off() {
  dac.setVoltage(0, false); // remove load
  icounter_off = 0;
  V_RM_off_v = 0.0;
  I_sense_off_A = 0.0;
  long t_stop = micros() + itime_off_msec * 1000;
  while (micros() < t_stop) {
    float v_off_raw = ads.readADC_Differential_0_1() * ADC_V_per_ADU + V_RM_off_v;
    float i_off_raw = ads.readADC_Differential_2_3() * ADC_V_per_ADU_2 / R_sense;
    V_RM_off_v += v_off_raw;
    I_sense_off_A += i_off_raw;
    icounter_off++;
  }
  V_RM_thevenin_v = V_RM_off_v / icounter_off;
}
void func_meas_on() {
  long t_stop = micros() + itime_on_msec * 1000;
  icounter_on = 0;
  V_RM_on_v = 0.0;
  I_sense_on_A = 0.0;
  while (micros() < t_stop) {
    float v_on_raw = ads.readADC_Differential_0_1() * ADC_V_per_ADU + V_RM_on_v;
    float i_on_raw = ads.readADC_Differential_2_3() * ADC_V_per_ADU_2 / R_sense;
    V_RM_on_v += v_on_raw;
    I_sense_on_A += i_on_raw;
    icounter_on++;
  }
  V_RM_loaded_v = V_RM_on_v / icounter_on;
  I_load_A = I_sense_on_A / icounter_on;
}
```