<u>ARDUINO PROJECT</u>

<u>TITLE : A VOLTMETER OF RANGE 0-5V</u>

<u>TABLE OF CONTENTS</u>

INTRODUCTION

This is an Arduino project, a VOLTMETER I made that can measure voltages from 0V to 5V. It has an error range of -0.02V to +0.02V, which I believe to be an excellent one for my first time in electronics and circuitry.

USAGE AND PURPOSE

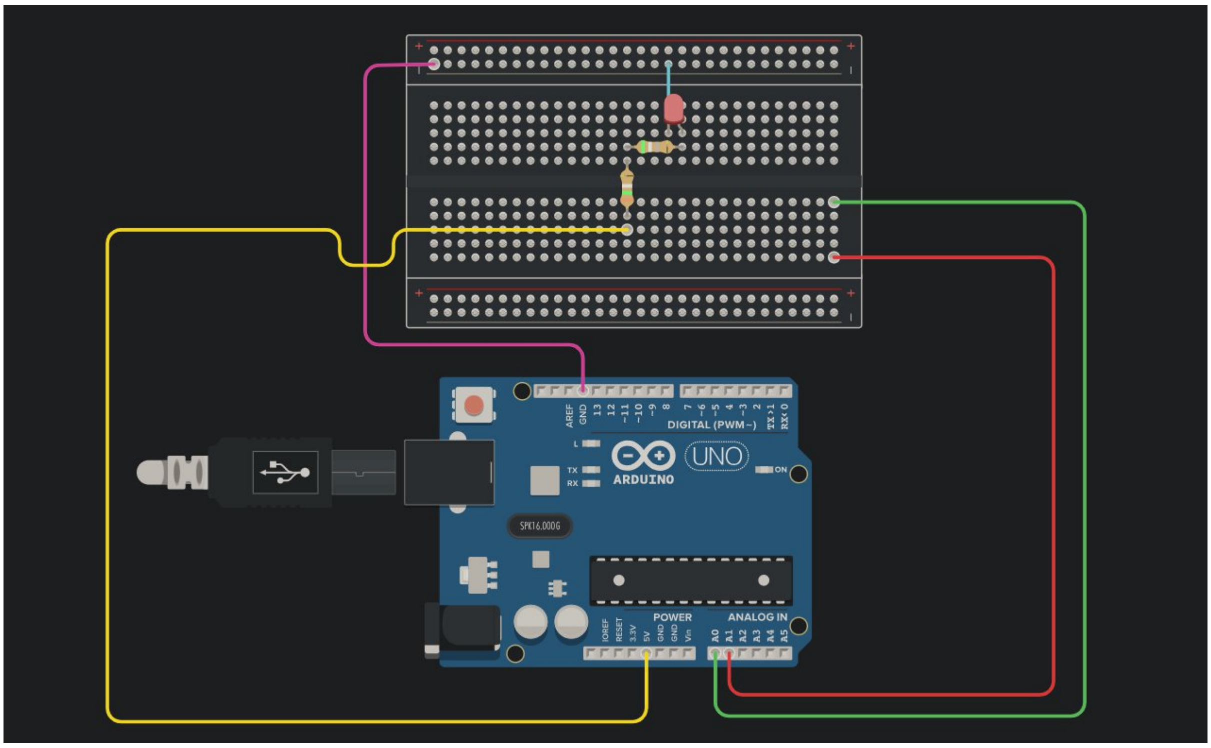
I don't have a voltmeter, and I started my journey in electronics, circuitry and Arduino. Usually when I face a problem, it often takes me 2 hours or more to debug to the most plausible problem. I eventually got demotivated with this. But then I got a plan to make a voltmeter using the knowledge of Analog inputs. Now, it made my diagnosis time to 2 minutes, and at maximum, 10 minutes for now. The purpose of this project is to share my views and my method on a problem I believe a lot of people face, cause electronic diagnosis is often a challenge cause we almost did everything right, and yet the circuit isn't working, or at worst, it fried, and we don't know why and how.

COMPONENTS AND TOOLS LIST

MALE TO MALE JUMPER WIRES	*4*
LED	*1*
RESISTORS (220 ohms)	*2*
ARDUINO UNO	*1*
USB CABLE FOR ARDUINO UNO	*1*

CIRCUIT DIAGRAM AND EXPLANATION

CIRCUIT DIAGRAM



EXPLANATION

- *This circuit is designed for making a Voltmeter and testing it and making sure the following points and how I made sure of it.*
- *We need to make sure that the voltage reading doesn't randomize, that's why I used 2 resistors of resistance 220 ohms in series. If the reading isn't randomised, then there has to be the same reading across both the resistors.*
- *We need to make sure that the circuit is closed and working, and the simple way to do is by adding an LED. This gives a confirmation that the circuit works. Hence if there is any problem in the reading, we would know that it is the problem with the voltmeter setup and not the circuit.*
- *The wires connected to A0 and A1 can move freely to probe and are not fixed, despite the implications by the circuit image*

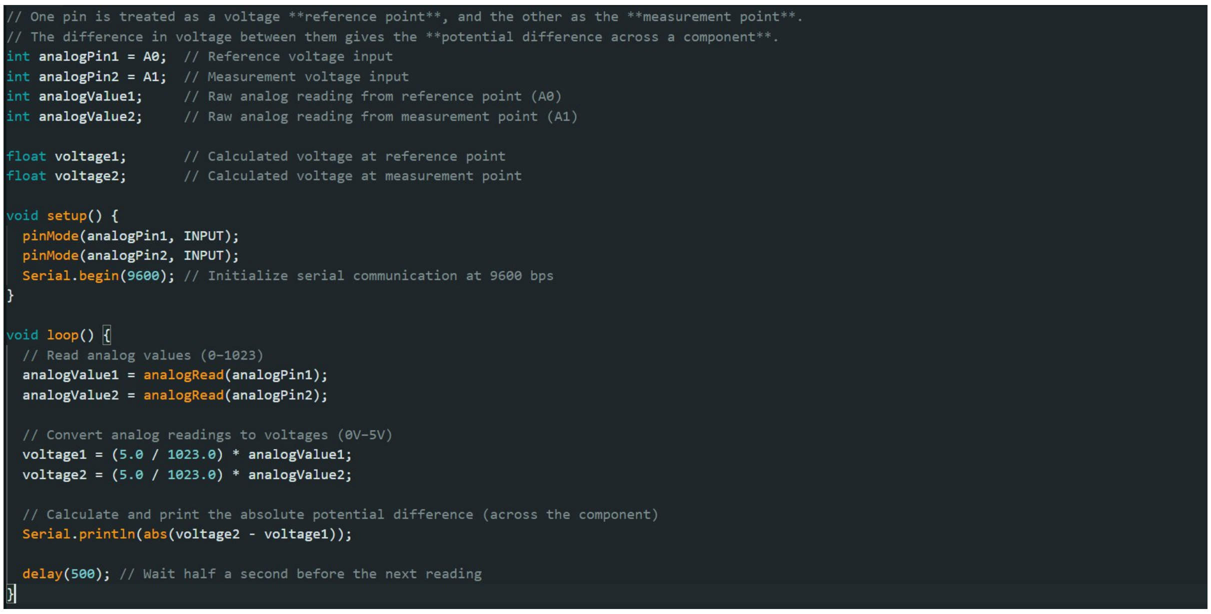
PROCEDURE

- *Take 2 of the 4 jumper wires and insert them in any of the 2 Analog pins (A0 to A5). I used A0 and A1 in my circuit. Leave them as is.*
- *Now connect the 5V pin on the Arduino with the 1st resistor on the breadboard using a jumper wire.*
- *Connect the 2nd resistor with the 1st resistor on the breadboard.*
- *Connect the long leg of the LED to the other end leg of the 2nd resistor.*
- *Connect the short leg of the LED to the GND on Arduino pin. I used the ground rail for presentation purposes. One may do it if they want to.*

CAUTION AND WARNINGS

- *Make sure the wires are properly connected. To ensure that, just press the components a little firmly. That makes sures that the connection is not loose. Some of the most common problems appear here. One does every checkup and doesn't find anything, cause the wiring is loose. Loose wirings often create a misdirection and problems in circuitry.*
  Bonus Tip: if the circuit doesn't work, check the wiring first.
- *Don't connect the ground rail to the red rail often marked as '+'. It might cause problems.*
- *Make sure that the LED is connected to properly. Its long leg should be towards high voltage, and short leg towards low voltage or GND. Otherwise the circuit just won't work.*
  Tip to remember: long = high/more, Short = low/less.
- *Follow all the standard procedures of a breadboard. Sometimes the circuit doesn't work cause one either connected in completely different rail, or just connected the component on the same rail, often causing open circuit and short circuit, respectively.*
- *Don't forget to use a resistor with LED. It's a common mistake and personally, **I ended up almost frying my Arduino**. Remember this as a rule of thumb, to use resistor with LED.*

CODE AND EXPLANATION



```
// One pin is treated as a voltage **reference point**, and the other as the **measurement point**.
// The difference in voltage between them gives the **potential difference across a component**.
int analogPin1 = A0;  // Reference voltage input
int analogPin2 = A1;  // Measurement voltage input
int analogValue1;     // Raw analog reading from reference point (A0)
int analogValue2;     // Raw analog reading from measurement point (A1)

float voltage1;       // Calculated voltage at reference point
float voltage2;       // Calculated voltage at measurement point

void setup() {
  pinMode(analogPin1, INPUT);
  pinMode(analogPin2, INPUT);
  Serial.begin(9600); // Initialize serial communication at 9600 bps
}

void loop() {
  // Read analog values (0-1023)
  analogValue1 = analogRead(analogPin1);
  analogValue2 = analogRead(analogPin2);

  // Convert analog readings to voltages (0V-5V)
  voltage1 = (5.0 / 1023.0) * analogValue1;
  voltage2 = (5.0 / 1023.0) * analogValue2;

  // Calculate and print the absolute potential difference (across the component)
  Serial.println(abs(voltage2 - voltage1));

  delay(500); // Wait half a second before the next reading
}
```

EXPLANATION

- *This code is sketched to support the circuit. This code treats one of the 2 points as reference and the other point as measurement point.*
- *Arduino reads the inputs of 0V as 0 and 5V as 1023. 1023 is the maximum value of output that the Arduino can measure and sustain.*
- *But we don't want to read the values like 1023, we want to read values in meaningful ways to humans, like 5V.*
- *information: The **linearity is a built-in feature** of the Arduino ADC (Analog-to-Digital Converter) and is accepted as fact for typical applications — not just an assumption.*
- *Hence using the knowledge of linear nature, we used the fraction '5./1023.' for the conversion of Analog inputs to meaningful values. One can also use '5.0/1023.0', that's also valid.*
- *We use floating point values for extra precision.*
- *The abs() function makes sure that the value is sign independent, cause all that matters in this project is potential difference, determining the voltage across certain component. That's all is the objective. This also eases the hassle of being unnecessarily too particular.*
- *The delay of 500ms is given so that it is easier to read. We don't want to overflow the Arduino so fast that we can't read values.*

TESTING AND RESULTS

To test the circuit, I will share the following observations that I found.

- ***The LED glows**, implying that the circuit works.*
- *Placed one end of the wire connected to the Analog pin A0 to one leg of 1st resistor, then placed one end of the wire connected to the Analog pin A1 to the other leg of the 1st resistor.*
- *The reading I got was consistently **1.55V**. it may fluctuate to +0.01V or -0.01V.*
- *Do the same for the **second resistor**, one should get **the same reading** since the resistor values are same.*
- *Now check the **voltage across the LED**, it should have around **1.9V.***
- *When one adds all those values, they could **get 4.99V.***
- *Now connect the voltmeter across the circuit, with one point connected with the start, and other point connected to the end, the reading should come as **5V exact**.*
- *This discrepancy of 0.01V is just due to **ADC conversion errors or minor noise**. But this is precise indeed.*

CHALLENGES AND LEARNINGS

- *I faced a lot of challenges on the way to make my voltmeter.*
- *One of those was that I **messed up python with C++** and I faced a lot of errors, and sometimes I just never understood the problem but **with practice and focused sessions**, I managed to reduce the number of errors, although they still exist, but progress is happening.*
- *My other problem was the very cause of this projects. My circuits failing and I can't find out the reason for the cause. It often took me 2 hours and my entire learning session sometimes gets completed with the different set of goals being completed. Where I plan on finish learning certain level of coding and hardware, I end up learning how to diagnose. But my DIY Voltmeter can just make it easier to work with. Now I can get my diagnosis with 100% precision whether it's my broken wires, or broken LEDs, or incorrect grounding, everything can be detected without spending for 2 hours and it increases my learning efficiency.*
- *The other challenge I faced this entire journey was **self-doubt**. I never made anything in my life and this was the first time I ever made an attempt for myself to make something out of my will. But as this journey progressed and as I kept on learning, I was able to see how far I was able to come in making something. If anything, my other big win was winning over my fear and self-doubt. And if I can do it, I believe anyone can.*
- *Even if this wasn't a major or a very big project in world's views, it's at the very least a **very big first step for me**, and I am happy and had fun in this journey overall, despite struggles and other self-resisting factors.*

<u>CONCLUSION</u>

This part is going to be short, but I would like to end this journey in this way.

*“**<u>It’s just the beginning. It’s not the end. The journey isn’t over yet.”</u>***

Cause it was my first project, and I would make many more. Not cause the world asked it or anything, but just because I want to. I want to build, build and build. That’s just who I am.

I would really welcome any suggestions or challenges, and I am looking for a mentor who can guide me in my goal. I would to appreciate and thank all of you who read this all the way to the end.

For any of these subjects mentioned above, contact me on: ***arduinomechtech@gmail.com***

<u>THANK YOU….</u>

CODE TO COPY…

```
// One pin is treated as a voltage **reference point**, and the other as
the **measurement point**.
// The difference in voltage between them gives the **potential
difference across a component**.
int analogPin1 = A0;  // Reference voltage input
int analogPin2 = A1;  // Measurement voltage input
int analogValue1;     // Raw analog reading from reference point (A0)
int analogValue2;     // Raw analog reading from measurement point (A1)

float voltage1;       // Calculated voltage at reference point
float voltage2;       // Calculated voltage at measurement point

void setup() {
  pinMode(analogPin1, INPUT);
  pinMode(analogPin2, INPUT);
  Serial.begin(9600); // Initialize serial communication at 9600 bps
}

void loop() {
  // Read analog values (0–1023)
  analogValue1 = analogRead(analogPin1);
  analogValue2 = analogRead(analogPin2);

  // Convert analog readings to voltages (0V–5V)
  voltage1 = (5.0 / 1023.0) * analogValue1;
  voltage2 = (5.0 / 1023.0) * analogValue2;

  // Calculate and print the absolute potential difference (across the
component)
  Serial.println(abs(voltage2 - voltage1));

  delay(500); // Wait half a second before the next reading}
```