Smart Multimeter Using Microcontroller Systems

Assignment 1

Chhaya, 240307

Introduction

This project marks the transition from theoretical circuit analysis to applied engineering practice. The objective of this assignment was to design and simulate a measurement module capable of performing voltage, capacitance, and resistance measurements using an Arduino Uno in Tinkercad. This report documents the construction, code logic, and analysis of three core measurement sub-systems: a Voltage Divider, a Capacitance Meter, and an Ohmmeter.

Task A: Voltage Divider Analysis

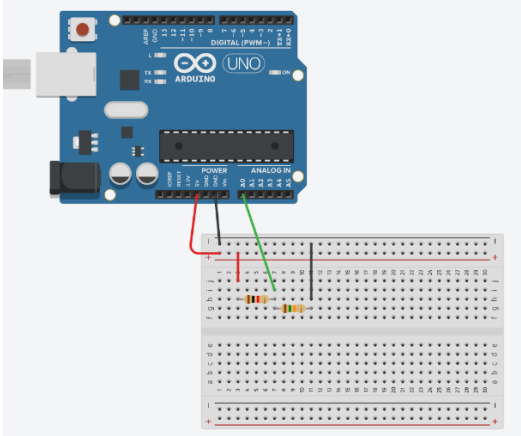
Objective

To build a voltage divider circuit and interface the midpoint with the Arduino Analog-to-Digital Converter (ADC) to simulate a digital voltmeter.

Circuit & Implementation

The circuit consists of two resistors in series (R_1 and R_2) connected between 5V and GND. The midpoint voltage (V_out) is measured by Arduino Pin A0.

Circuit Diagram:



Arduino Code:

The code reads the analog input (0-1023) and converts it to a voltage value using the reference voltage of 5.0V.

```
const float refVoltage = 5.0;

void setup() {
  Serial.begin(9600);
  delay(100);

  int Analog = analogRead(A0);

  //Convert the analog reading to a voltage(0 - 5V):
  float voltage = Analog*(refVoltage / 1023.0);

  Serial.print(" | Measured Voltage: ");
  Serial.print(voltage);
  Serial.println(" V");
}

void loop() {

}
```

Serial Monitor

```
| Measured Voltage: 2.50 V
| Measured Voltage: 3.40 V
| Measured Voltage: 4.69 V
```

Results & Comparison

Three different resistor pairs were tested to verify the voltage divider formula:

V_out = V_in * {R_2/ (R_1 + R_2)}

S.no.	**R_1**	**R_2**	**Theoretical V_out**	**Measured V_out**	**Error**
1	10 kΩ	10 kΩ	2.50 V	2.50 V	0.0%
2	4.7 kΩ	10 kΩ	3.40 V	3.40 V	0.0%
3	1 kΩ	15 kΩ	4.69 V	4.69 V	0.0%

* **Purpose:** A voltage divider scales down high input voltages to a safe range (0-5V) that the microcontroller can process, preventing damage to the ADC pins.

* Conversion Formula: The ADC converts voltage into an integer ($0-1023$). The voltage is recovered using:

Voltage = ADC_Value * (5.0/1023) Volt

* **Observations:** The results in Tinkercad were ideal (0% error) because simulated components have perfect tolerance. In a real-world scenario, resistor tolerance and ADC quantization noise would introduce discrepancies.

Task B: Capacitance Measurement (RC Time Constant)

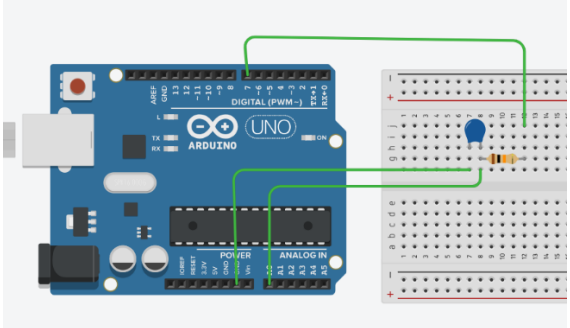
Objective

To simulate a capacitance meter by measuring the time required to charge a capacitor to 63.2% of the supply voltage through a known series resistor.

Circuit & Implementation

A series circuit was created with a 10 Ohm resistor and a capacitor under test. Pin 7 acts as the voltage source (charging pin), while Pin A0 monitors the capacitor voltage.

Circuit Diagram:



Arduino Code:

The code measures the duration (micro sec) for the voltage to rise from 0V to approx 3.16V (ADC value ~647).

```
const int chargePin = 7;
const int monitorPin = A0;
const float resistance = 10000.0;
const int targetReading = 647;//63.2% of 1023

void setup() {
  Serial.begin(9600);
  pinMode(chargePin, OUTPUT);
}

void loop(){
  digitalWrite(chargePin, LOW);
  pinMode(monitorPin, OUTPUT);  // A0 set to output to drain faster
  digitalWrite(monitorPin, LOW);

  delay(2000);
  pinMode(monitorPin, INPUT);  // Reseting A0 back to Input for reading
  unsigned long startTime = micros();
  digitalWrite(chargePin, HIGH);
  while(analogRead(monitorPin) < targetReading){
  }
  unsigned long elapsedTime = micros() - startTime;
  float capacitance = ((float)elapsedTime / resistance);

  Serial.print("Time to 63.2%: ");
  Serial.print(elapsedTime / 1000.0);
  Serial.print(" ms");
  Serial.print(" | Calculated Capacitance: ");
  Serial.print(capacitance);
  Serial.println(" uF");
  Serial.println("---------------------------------");
  delay(3000); }
```

Serial Monitor

```
Time to 63.2%: 4721.89 ms | Calculated Capacitance: 472.19 uF
---------------------------------
Time to 63.2%: 1004.62 ms | Calculated Capacitance: 100.46 uF
---------------------------------
```

Results

Using the formula C = Time/R, the capacitance was derived.

Component	Expected Time (R×C)	Measured Time	Calculated C	Expected C	Error
Test 1	1000 ms	1004.62 ms	100.46 µF	100 µF	+0.46%
Test 2	4700 ms	4721.89 ms	472.19 µF	470 µF	+0.46%

- **Why 63.2%?** The voltage across a charging capacitor follows the equation

  V(t) = V_in*(1 - e^{-t/RC}). When t = RC (one time constant), the voltage reaches 1 - e^(-1) which is approx 0.632 (63.2%). Measuring the time to this specific threshold allows us to find capacitance without complex logarithmic math (C = t/R).

- **Error Sources:** The slight positive error (+0.46%) is because the Arduino takes time to execute analogRead(), meaning the voltage slightly exceeds the threshold before the code registers it, adding a small delay to the measured time.

<u>Task C: Beginner Ohmmeter Prototype</u>

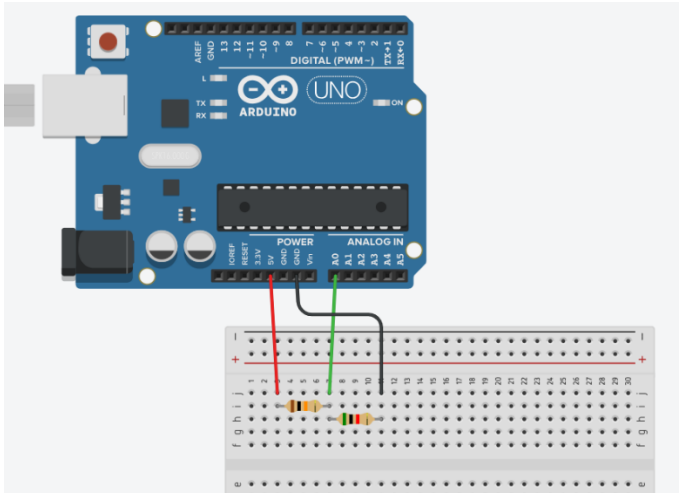
Objective

To determine the value of an unknown resistor (R_x) using a voltage divider configuration with a known reference resistor (R_ref).

Circuit & Implementation

- **R_ref (Known):** 10 kΩ connected between 5V and A0.
- **R_x (Unknown):** Connected between A0 and GND.

Circuit Diagram:



Theory & Calculation

Using the voltage divider rule, we solved for R_x:

$R_x = (V_{out} * R_{ref})/(V_{in} - V_{out})$

Converted to ADC steps (0-1023):

$R_x = ADC * 10{,}000/(1023 – ADC)$

Test Case	Set Value (Rx)	Measured Value	ADC Reading	Error Analysis
Low	5 kΩ	5000.00 Ω	341	Perfect match (Integer division aligned).
Mid	13 kΩ	12988.76 Ω	578	**0.09% Error**. Caused by ADC rounding (Quantization).
High	100 kΩ	100,000.00 Ω	930	Perfect match.

Quantization Error: As seen in the 13kΩ test, the ADC can only read integers. The theoretical ADC value for 13kΩ is 578.21, but the Arduino reads 578. This rounding causes the calculated resistance to be 12,988 Ohm instead of exactly 13,000 Ohm. This is a fundamental limitation of digital measurement systems.

Conclusion

This assignment successfully demonstrated the core operating principles of digital multimeters. By utilizing the Arduino ADC and fundamental circuit laws (Ohm's Law, RC Time Constant), we created functional tools in tinkercad for measuring voltage, capacitance, and resistance. The experiments highlighted the importance of **ADC resolution** and **timing latency** as primary sources of error in digital instrumentation.