assignment 1

Design and Simulation of Basic Measurement Modules Using Arduino Uno

Arushi Kondaskar
240200

Introduction

Using Tinkercad Circuits and an Arduino Uno, this project simulates three fundamental measurement modules, voltage measurement, capacitance measurement, and resistance measurement which together form the conceptual basis of a smart multimeter. Emphasis is placed on circuit design, mathematical modeling, experimental observation, and engineering documentation.

Task A: Voltage Divider Analysis and Measurement Module

1 Objective

To understand how a voltage divider is used inside measurement systems to safely scale voltages and how an Arduino measures voltage using its internal ADC.

2 Circuit Description

A voltage divider consisting of two resistors (R1) and (R2) is connected between the 5 V supply and ground. The midpoint of the divider is connected to Arduino analog pin A0.

The Arduino reads the scaled voltage and displays it on the Serial Monitor.

2 Theoretical Background

A voltage divider outputs a fraction of the input voltage given by:

vout=vin*R2/(R1+R2)

This principle allows higher voltages to be reduced to safe levels for ADC measurement.

2.4 ADC Conversion Formula

Arduino Uno uses a 10-bit ADC:

- ADC range: 0–1023
- Reference voltage: 5 V

$Vin=Vmeasured(R1+R2)/R2$

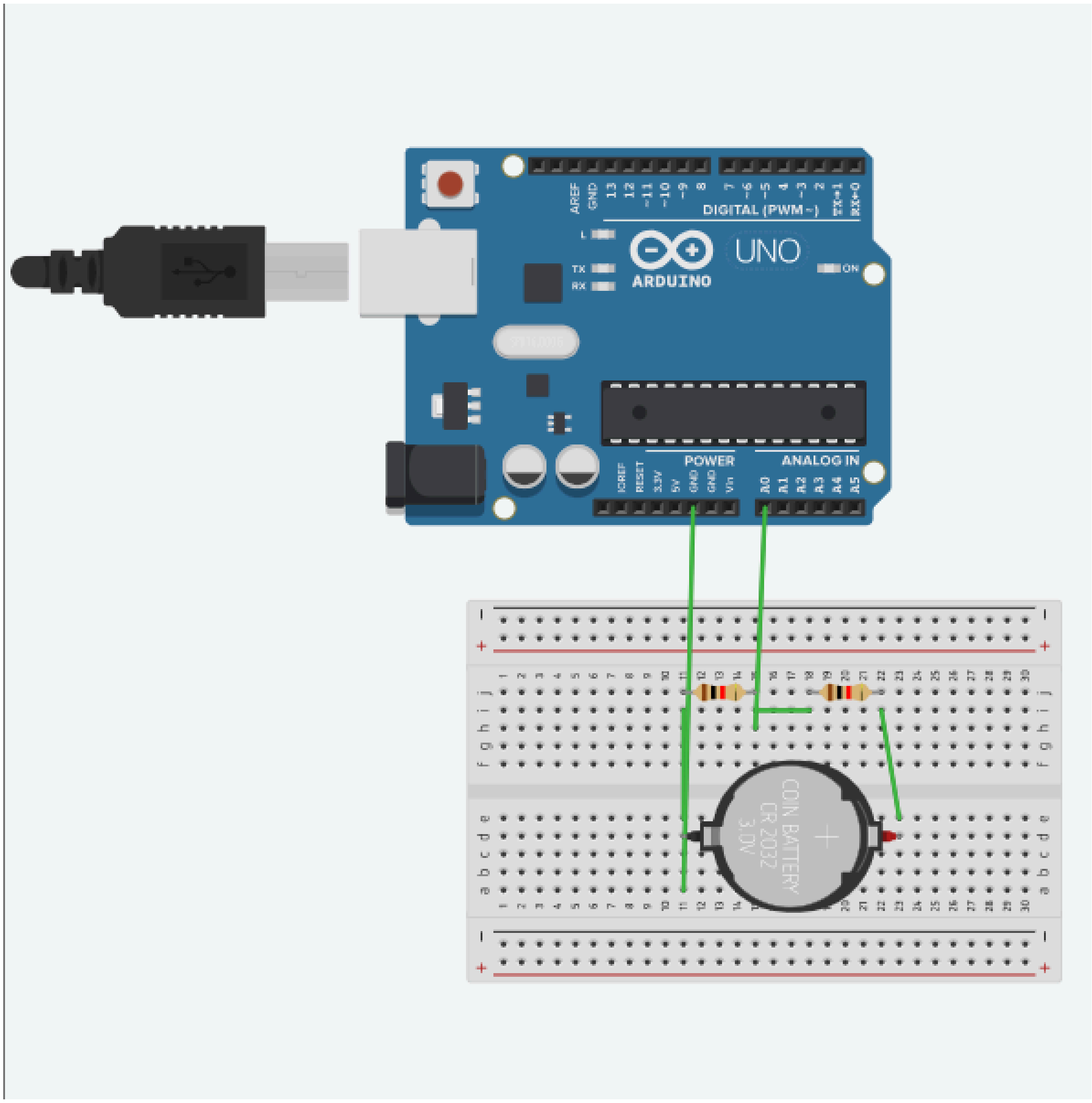
2.5 Experimental Results

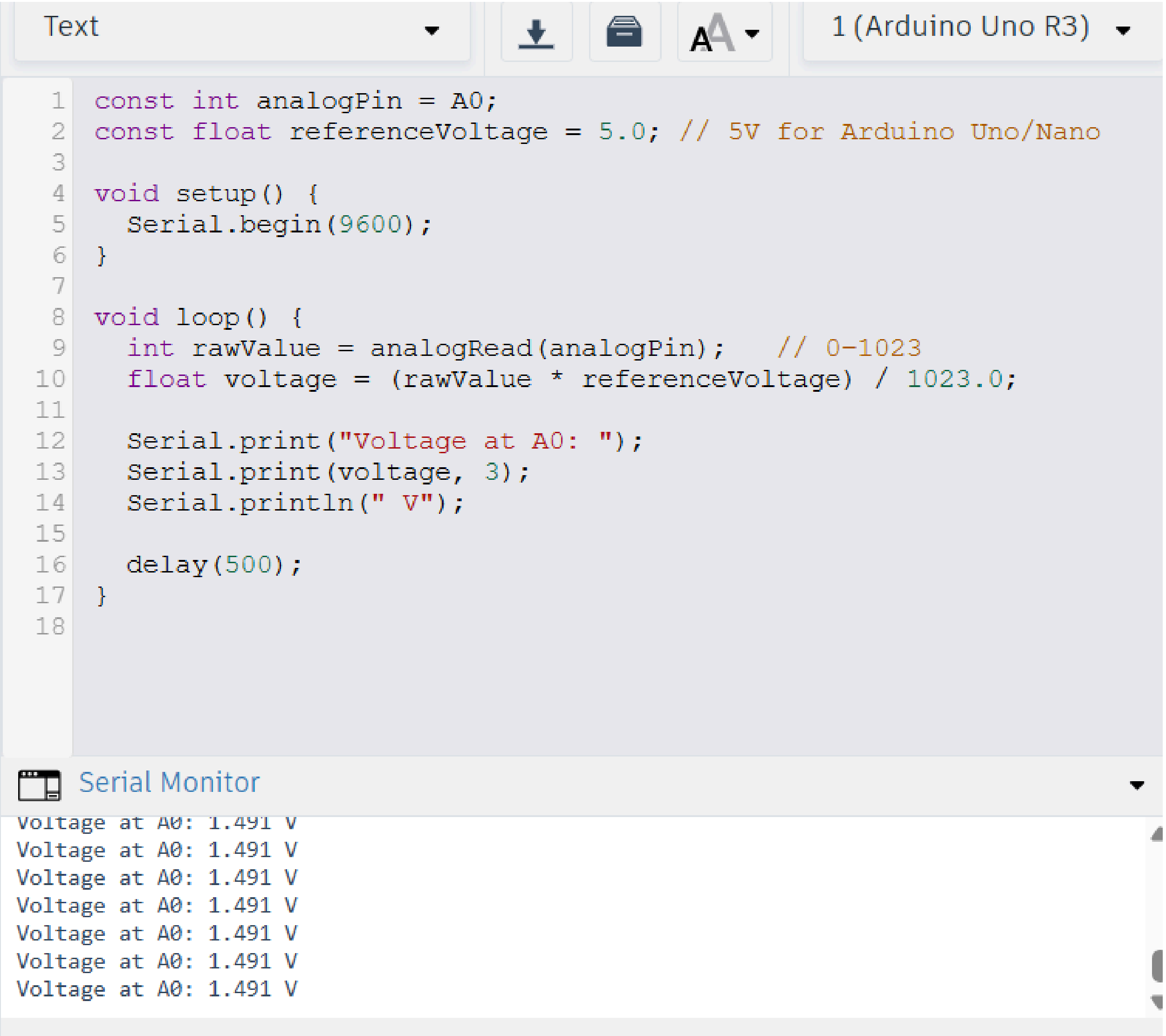
R1 (Ω)	R2 (Ω)	Theoretical Vout (V)	Measured Vout (V)
10k	10k	2.50	2.47
4.7k	10k	3.40	3.36
1k	15k	4.69	4.64

(Measured values obtained from Tinkercad simulation)

2.6 Observations

- Small deviations were observed due to ADC quantization.
- The measured voltage fluctuated slightly due to digital noise.
- Proper grounding was essential; missing a common ground resulted in zero readings.


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
COIN BATTERY
CR 2032
3.0V
+



```
const int analogPin = A0;
const float referenceVoltage = 5.0; // 5V for Arduino Uno/Nano

void setup() {
  Serial.begin(9600);
}

void loop() {
  int rawValue = analogRead(analogPin);   // 0-1023
  float voltage = (rawValue * referenceVoltage) / 1023.0;

  Serial.print("Voltage at A0: ");
  Serial.print(voltage, 3);
  Serial.println(" V");

  delay(500);
}
```

```
Voltage at A0: 1.491 V
Voltage at A0: 1.491 V
Voltage at A0: 1.491 V
Voltage at A0: 1.491 V
Voltage at A0: 1.491 V
Voltage at A0: 1.491 V
Voltage at A0: 1.491 V
```

Task B — Capacitance Measurement Using RC Time Constant

1 Objective

To simulate a capacitance measurement system by observing the charging behavior of an RC circuit and measuring the RC time constant.

2 Circuit Description

A resistor and capacitor are connected in series. The Arduino charges the capacitor through a digital pin while monitoring voltage across the capacitor using analog pin A0.

3.3 Theory: RC Time Constant

The voltage across a charging capacitor is given by:
V(t) = Vcc (1 - e^{-t/RC})

At time (t = RC), the voltage reaches approximately **63% of the supply voltage**. This point is used to calculate capacitance:
C = t/R

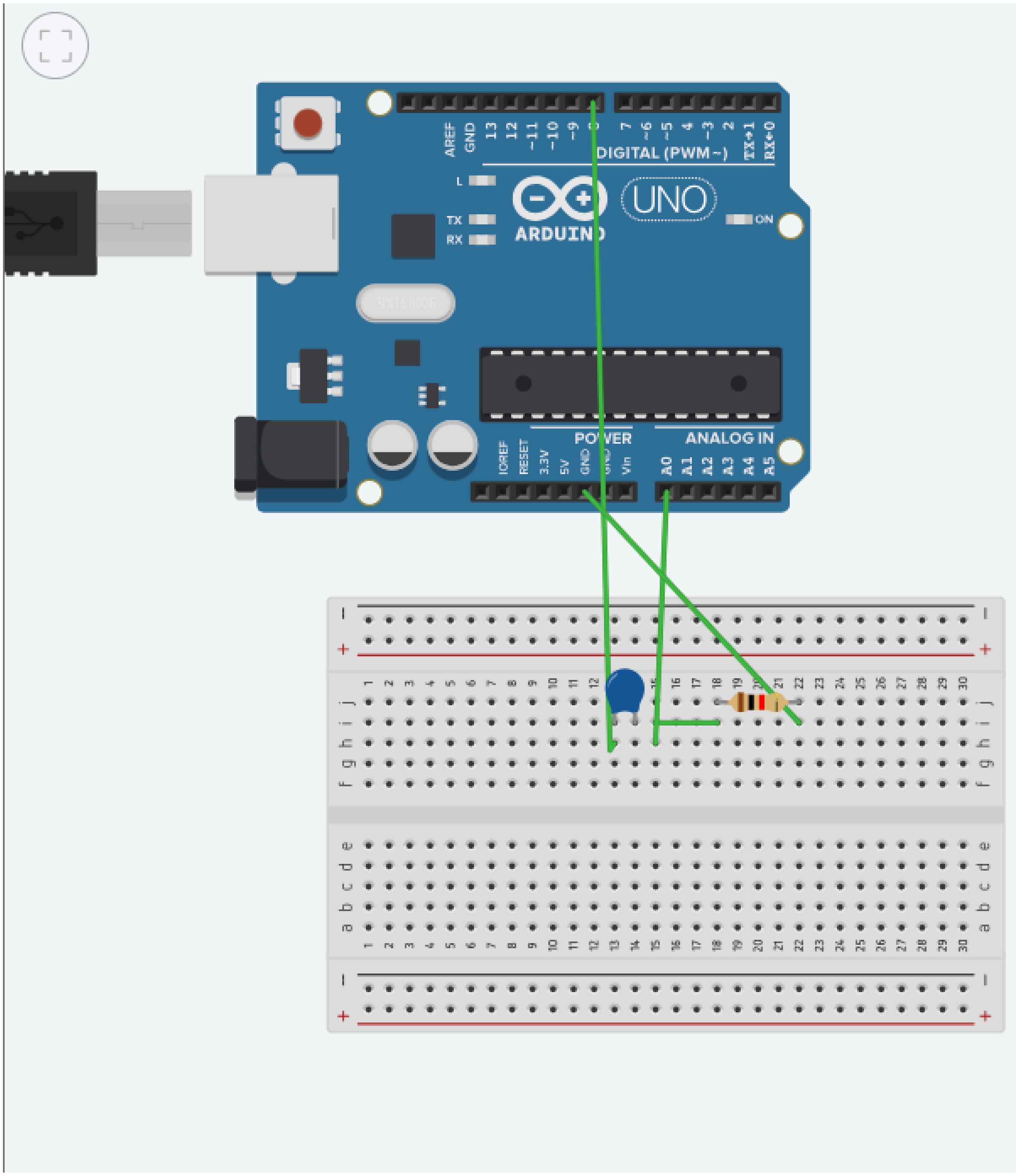
3.4 Experimental Results

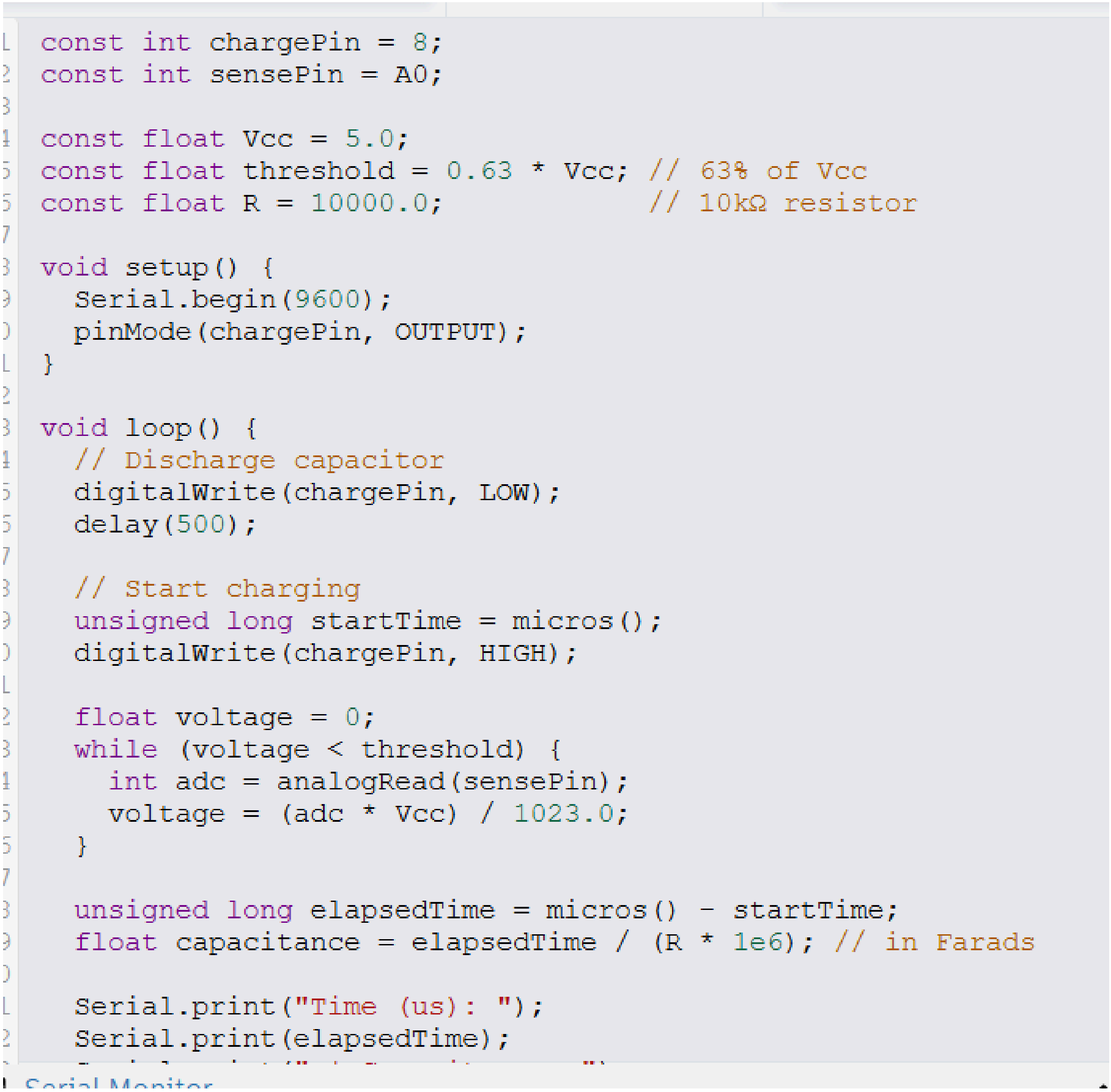
Capacitor Value	**Expected Time (ms)**	**Measured Time (ms)**
10 μF	100	105
100 μF	1000	980

(R = 10 kΩ)

3.5 Observations and Error Sources

- Capacitor tolerance (±10% to ±20%) significantly affected accuracy.
- ADC resolution limited exact detection of the 63% threshold.
- Small timing delays occurred due to loop execution and sampling speed.
- Parasitic capacitance from the breadboard introduced minor error.


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
IOREF
RESET
3.3V
5V
GND
GND
Vin
ANALOG IN
A0
A1
A2
A3
A4
A5



```
const int chargePin = 8;
const int sensePin = A0;

const float Vcc = 5.0;
const float threshold = 0.63 * Vcc; // 63% of Vcc
const float R = 10000.0;            // 10kΩ resistor

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

void loop() {
  // Discharge capacitor
  digitalWrite(chargePin, LOW);
  delay(500);

  // Start charging
  unsigned long startTime = micros();
  digitalWrite(chargePin, HIGH);

  float voltage = 0;
  while (voltage < threshold) {
    int adc = analogRead(sensePin);
    voltage = (adc * Vcc) / 1023.0;
  }

  unsigned long elapsedTime = micros() - startTime;
  float capacitance = elapsedTime / (R * 1e6); // in Farads

  Serial.print("Time (us): ");
  Serial.print(elapsedTime);
```

Task C — Beginner Ohmmeter Prototype

1 Objective

To design a basic ohmmeter using a voltage divider and calculate an unknown resistance using Arduino.

2 Circuit Description

A known reference resistor Rref is connected in series with an unknown resistor Rx. The midpoint voltage is measured by Arduino A0.

4.3 Theory and Formula

Voltage divider equation:

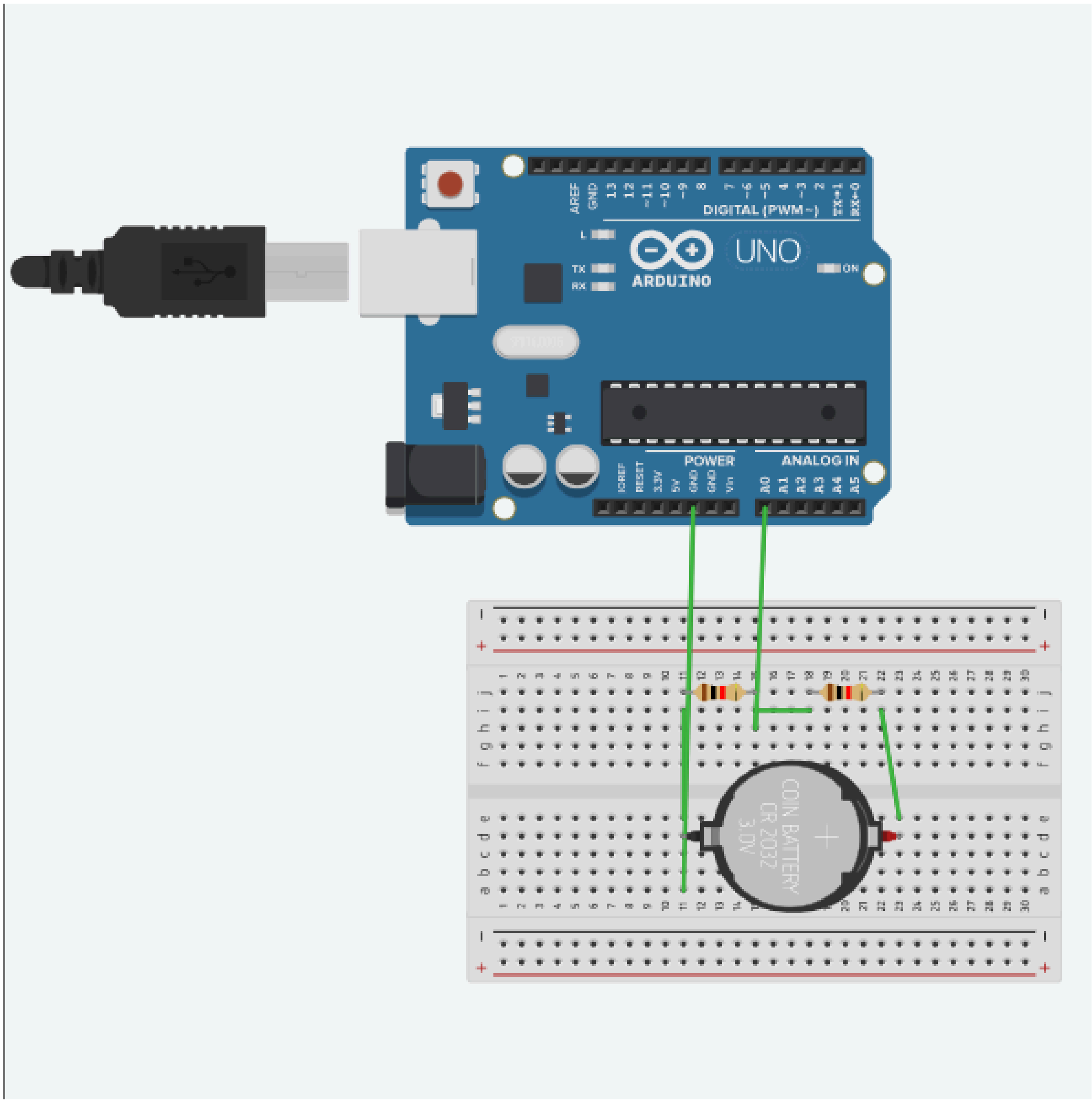
$$Vout = Vcc *\{Rx\}/(Rref + Rx)$$

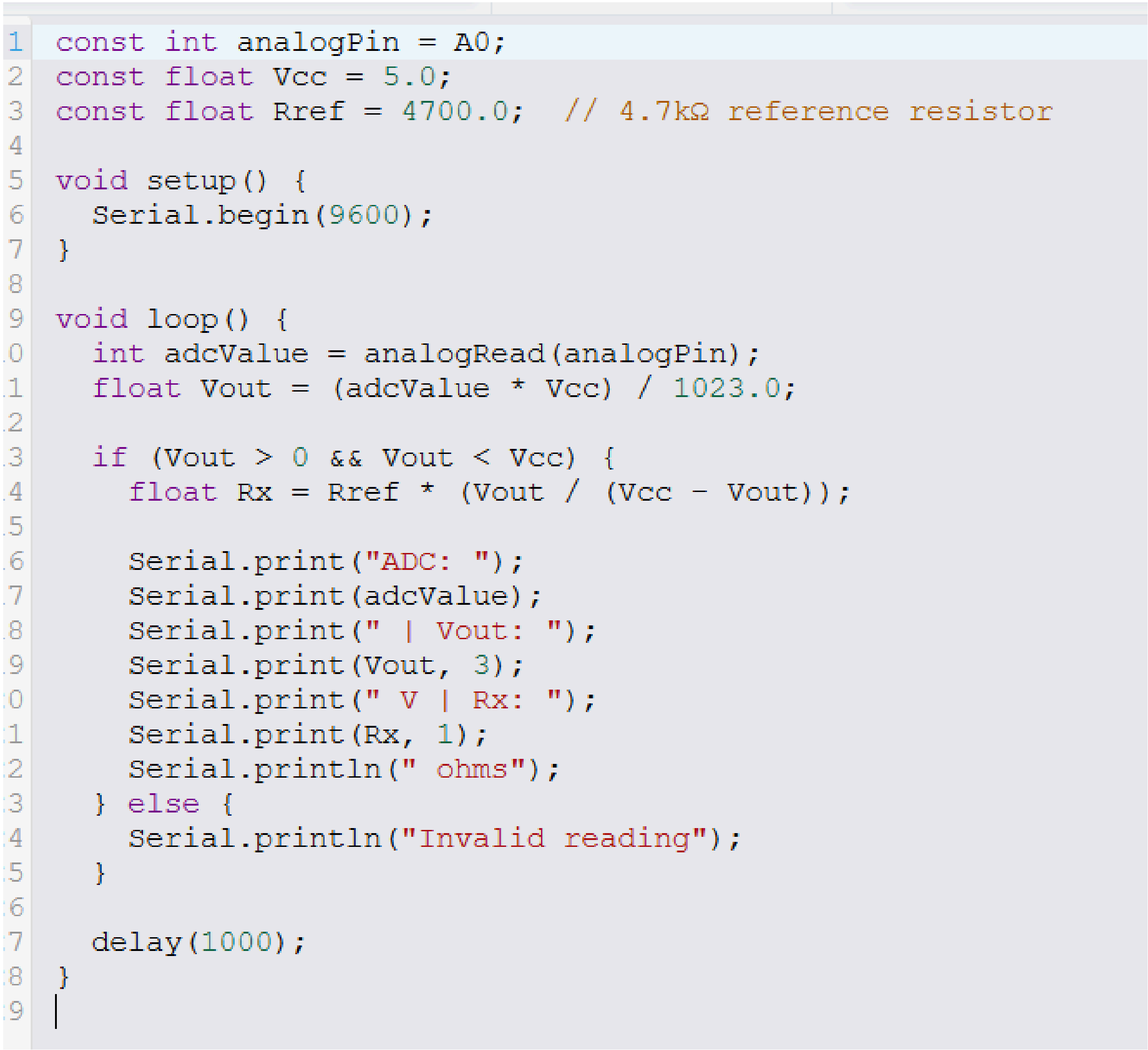
4 Comparison Table

Actual Resistance (Ω)	Measured Resistance (Ω)	Error (%)
1,000	1,050	+5.0
2,200	2,350	+6.8
6,800	6,550	−3.7

4 Measurement Uncertainty

- Resistor tolerance introduced systematic error.
- ADC resolution limited voltage precision.
- Measurement accuracy reduced when (Rx) was much larger or smaller than (Rref)


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX➔1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
COIN BATTERY
CR 2032
3.0V
+



```
const int analogPin = A0;
const float Vcc = 5.0;
const float Rref = 4700.0;  // 4.7kΩ reference resistor

void setup() {
  Serial.begin(9600);
}

void loop() {
  int adcValue = analogRead(analogPin);
  float Vout = (adcValue * Vcc) / 1023.0;

  if (Vout > 0 && Vout < Vcc) {
    float Rx = Rref * (Vout / (Vcc - Vout));

    Serial.print("ADC: ");
    Serial.print(adcValue);
    Serial.print(" | Vout: ");
    Serial.print(Vout, 3);
    Serial.print(" V | Rx: ");
    Serial.print(Rx, 1);
    Serial.println(" ohms");
  } else {
    Serial.println("Invalid reading");
  }

  delay(1000);
}
```

Conclusion

Through simulation and analysis, voltage measurement, capacitance measurement, and resistance measurement were successfully implemented using Arduino Uno. The project reinforced theoretical concepts while introducing practical challenges faced in real measurement systems. These experiments form the foundation for understanding the internal operation of digital multimeters and smart measurement devices.