ASSIGNMENT 1

Aarzoo
240019

4.1

1. Purpose of a Voltage Divider in Measurement Systems

A voltage divider is used in measurement systems to scale down an input voltage. This makes it safe and precise for electronic circuits, like microcontrollers and analog-to-digital converters (ADCs), to measure. Since ADC inputs have a limited voltage range, specifically 0 to 5 V for the Arduino Uno, a voltage divider reduces higher or varying voltages proportionally while keeping their relative magnitudes intact. Digital multimeters use voltage dividers inside to switch between different voltage ranges, ensuring accurate measurements and protecting their internal components.

2. ADC Reading to Voltage Conversion Formula

The Arduino Uno features a 10-bit ADC. This converts an analog voltage into a digital value ranging from 0 to 1023, which corresponds to 0 V to the reference voltage, typically 5 V.

The conversion formula is:

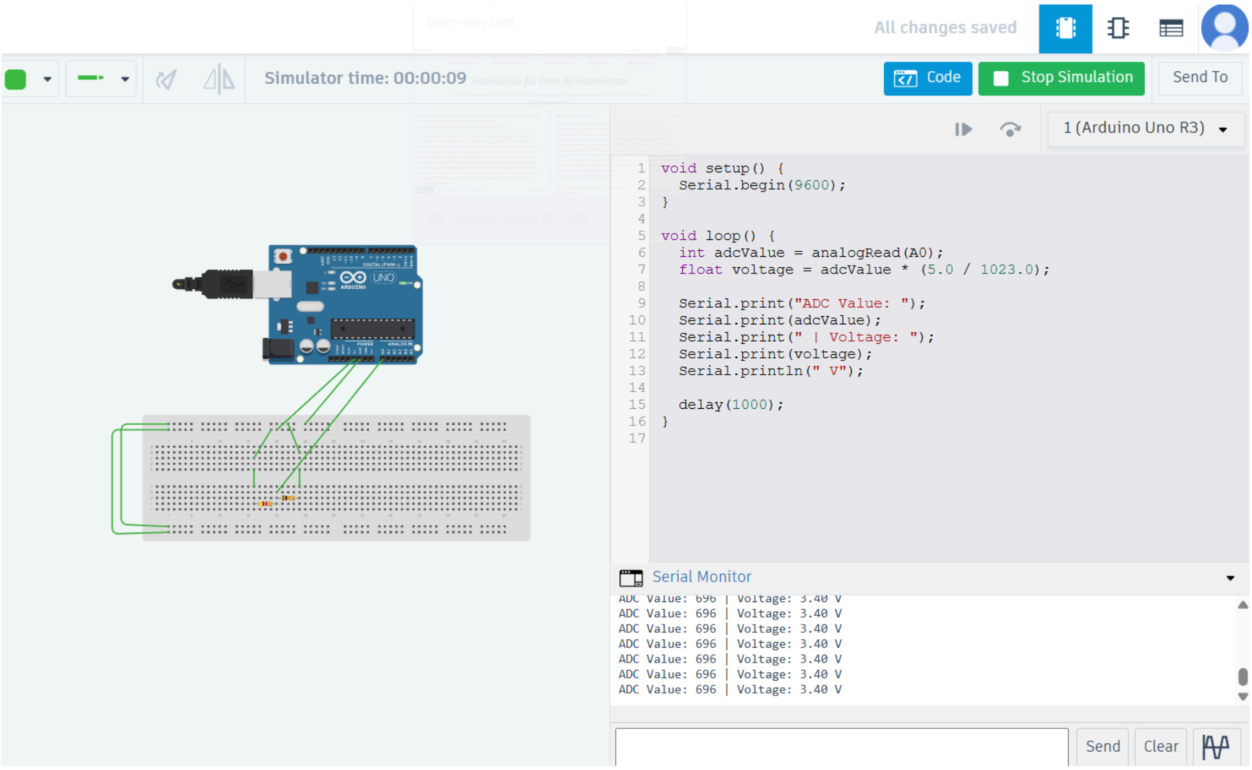
V measured =ADC Reading/1023×Vref

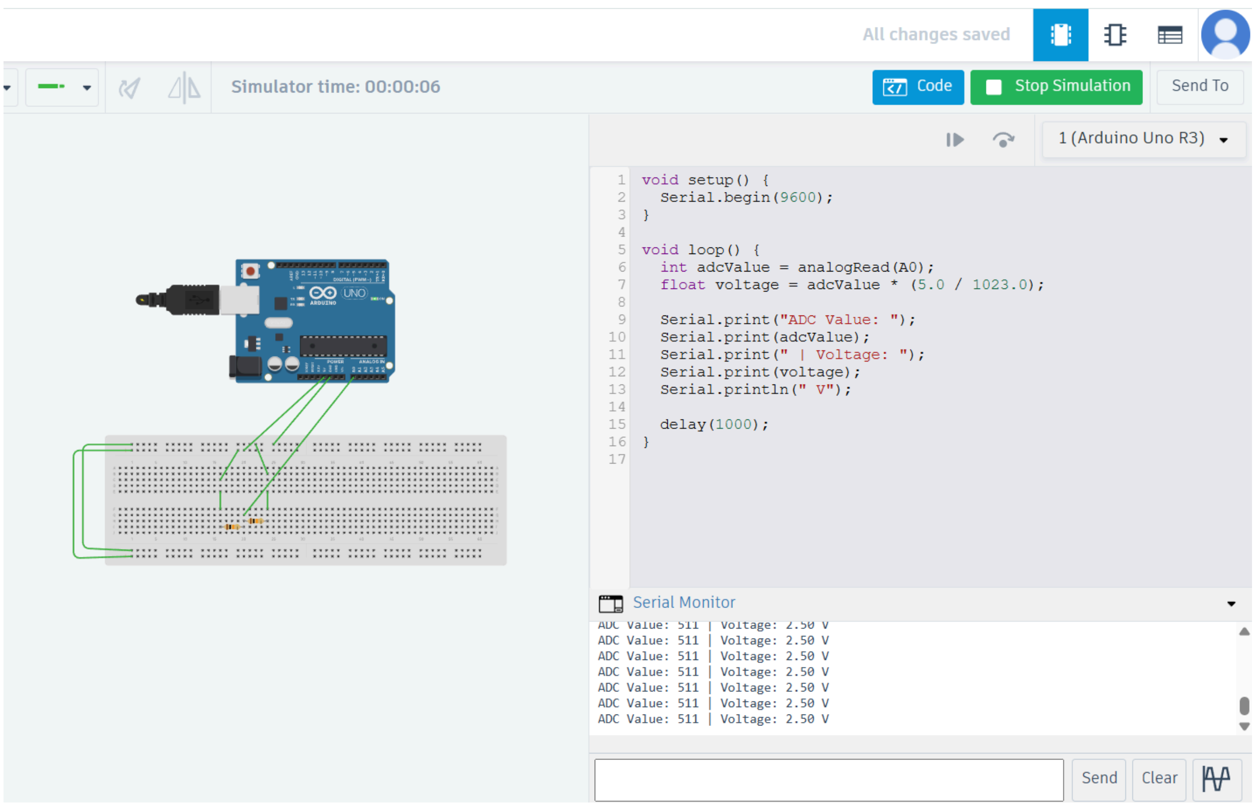
V measured =ADC Reading/1023×Vref

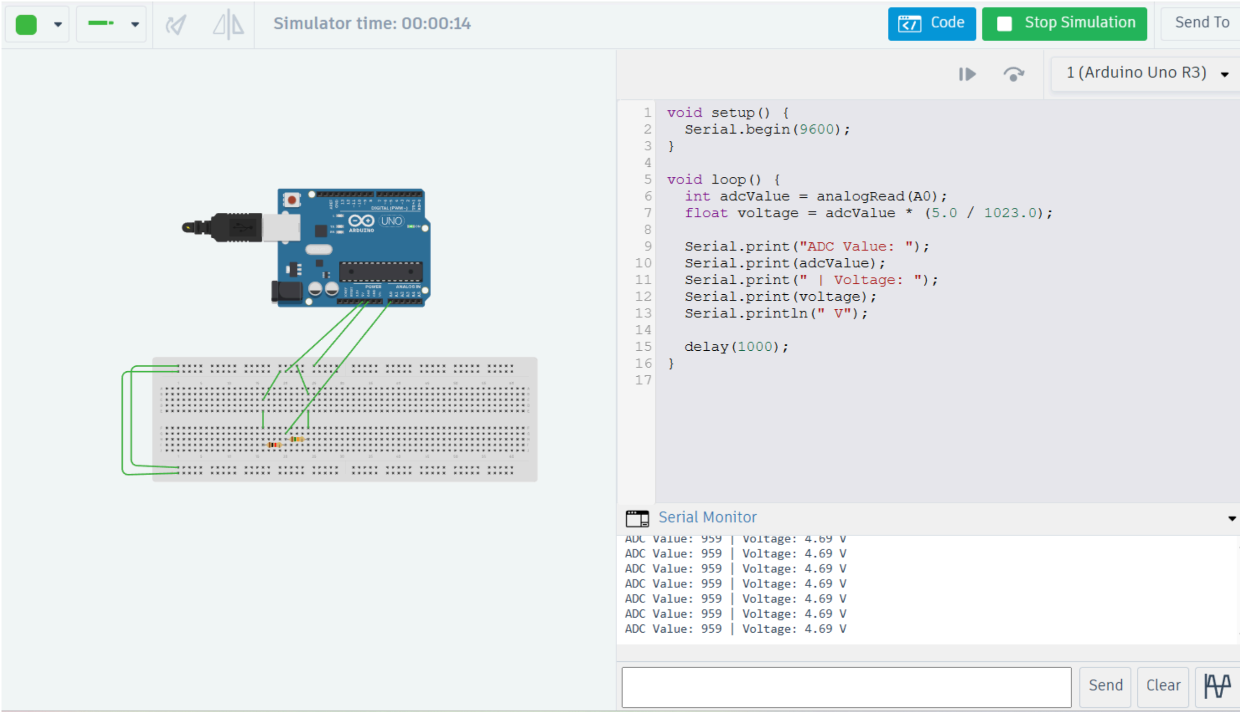
This formula lets you convert the ADC's digital output back to the actual analog voltage present at the input pin.

3. Observations (Error, Noise, and Unexpected Results)

The voltages measured by the Arduino were close to the theoretical values calculated with the voltage divider formula. Minor differences were observed due to ADC quantization error; the ADC can only show voltages in discrete steps. Small fluctuations in readings were noticed, which may be due to electrical noise and the ADC's limited resolution. In some cases, incorrect wiring led to zero voltage readings, underlining the importance of proper connections on the breadboard and maintaining a common ground reference. Overall, the voltage divider and ADC provided reliable and accurate voltage measurements within expected limits.


All changes saved
Simulator time: 00:00:09
Code
Stop Simulation
Send To
1 (Arduino Uno R3)
void setup() {
Serial.begin(9600);
}
void loop() {
int adcValue = analogRead(A0);
float voltage = adcValue * (5.0 / 1023.0);
Serial.print("ADC Value: ");
Serial.print(adcValue);
Serial.print(" | Voltage: ");
Serial.print(voltage);
Serial.println(" V");
delay(1000);
}
Serial Monitor
ADC Value: 696 | Voltage: 3.40 V
ADC Value: 696 | Voltage: 3.40 V
ADC Value: 696 | Voltage: 3.40 V
ADC Value: 696 | Voltage: 3.40 V
ADC Value: 696 | Voltage: 3.40 V
ADC Value: 696 | Voltage: 3.40 V
ADC Value: 696 | Voltage: 3.40 V
Send
Clear


All changes saved
Simulator time: 00:00:06
Code
Stop Simulation
Send To
1 (Arduino Uno R3)
void setup() {
Serial.begin(9600);
}
void loop() {
int adcValue = analogRead(A0);
float voltage = adcValue * (5.0 / 1023.0);
Serial.print("ADC Value: ");
Serial.print(adcValue);
Serial.print(" | Voltage: ");
Serial.print(voltage);
Serial.println(" V");
delay(1000);
}
Serial Monitor
ADC Value: 511 | Voltage: 2.50 V
ADC Value: 511 | Voltage: 2.50 V
ADC Value: 511 | Voltage: 2.50 V
ADC Value: 511 | Voltage: 2.50 V
ADC Value: 511 | Voltage: 2.50 V
ADC Value: 511 | Voltage: 2.50 V
ADC Value: 511 | Voltage: 2.50 V
Send
Clear



4.2
RC Time Constant and Significance of 63%

In a resistor–capacitor (RC) charging circuit, the voltage across the capacitor increases exponentially when a step voltage is applied. The charging equation is given by:

$V_c(t)=V_{max}(1-e^{-t/RC})$

The quantity $\tau=RC$ is known as the time constant of the RC circuit. At $t=\tau$ the capacitor voltage reaches approximately 63% of the supply voltage:

$V_c(\tau)=0.63\times V_{max}$

This occurs because:

$1-e^{-1}\approx0.63$

The 63% point is significant because it provides a direct and simple way to measure the time constant without needing curve fitting. By measuring the time taken for the capacitor voltage to reach 63% of the supply voltage, the capacitance can be calculated using:

$C=t/R$

This principle is used in the capacitance meter implemented in this experiment.

Measured vs. Expected Time Comparison

The experiment was performed using a 10 kΩ resistor and different capacitor values. The Arduino measured the time taken for the capacitor voltage to reach approximately 63% of 5 V (≈3.15 V).

Comparison Table

Capacitor Value (Nominal)	Expected Time t=RCt = RCt=RC	Measured Time (µs)	Calculated Capacitance (µF)
100 µF	1.0 s (1,000,000 µs)	≈ 999,248 µs	≈ 99.9 µF
500 µF	5.0 s (5,000,000 µs)	≈ 4,996,344 µs	≈ 499.6 µF

The measured times are very close to the theoretical values, indicating good agreement between practical and expected results.

Discussion of Error Sources

Although the measured values closely match theoretical expectations, some deviations can occur due to the following factors:

Component Tolerance

Resistors and capacitors have manufacturing tolerances. Electrolytic capacitors commonly have tolerances of ±20%, which can cause differences between nominal and actual capacitance values.

Incomplete Discharge of Capacitor

If the capacitor is not fully discharged before a new measurement, it may already be partially charged. This results in very small measured times (near zero), producing incorrect capacitance readings in subsequent measurements.

ADC Resolution

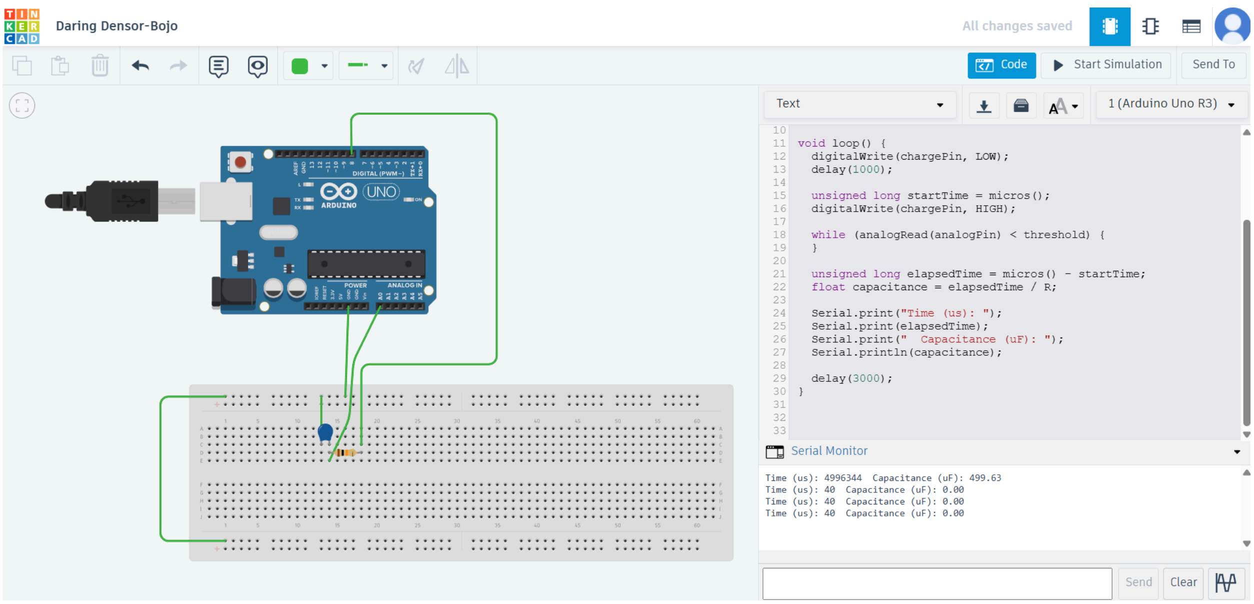
The Arduino Uno uses a 10-bit ADC, which divides the 0–5 V range into 1024 steps. This limits voltage measurement precision and affects the exact detection of the 63% threshold voltage.

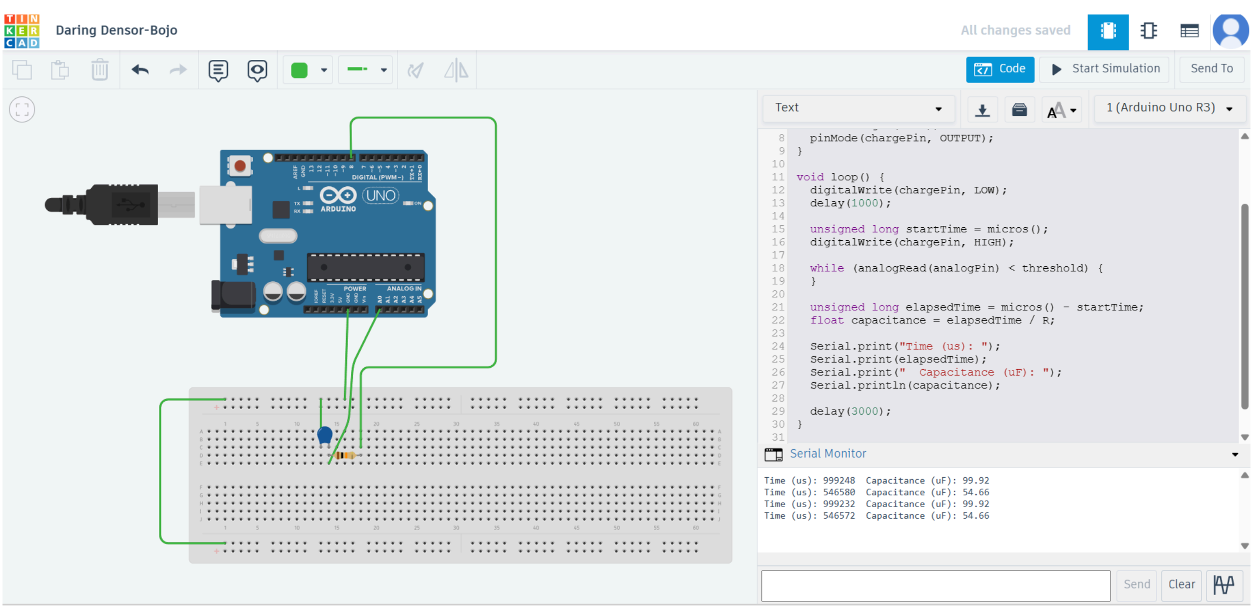
Timing and Software Overhead

Functions such as `analogRead()` introduce small delays, and loop execution time can slightly affect timing accuracy, especially for small capacitance values.

Noise and Simulation Limitations

Minor noise and idealized behavior in simulation (such as no leakage current) can also influence measured result





4.3

Step-by-step calculations

Read ADC value N from Pin A0 (1-1023)

Vout=N*Vcc/1023

I=Vcc/Rref+Rx

Vout=I.Rx=VccRx/Rref+Rx

Rx=RrefVout/Vcc-Vout

Actual vs. measured values

Resistor	**Nominal value (kΩ)**	**Measured resistance (kΩ)**	**Absolute error (kΩ)**	**Percent error (%)**
R1	7.0	7.00	0.00	0.0
R2	1.0	1.00	0.00	0.0
R3	3.0	3.01	0.01	0.3

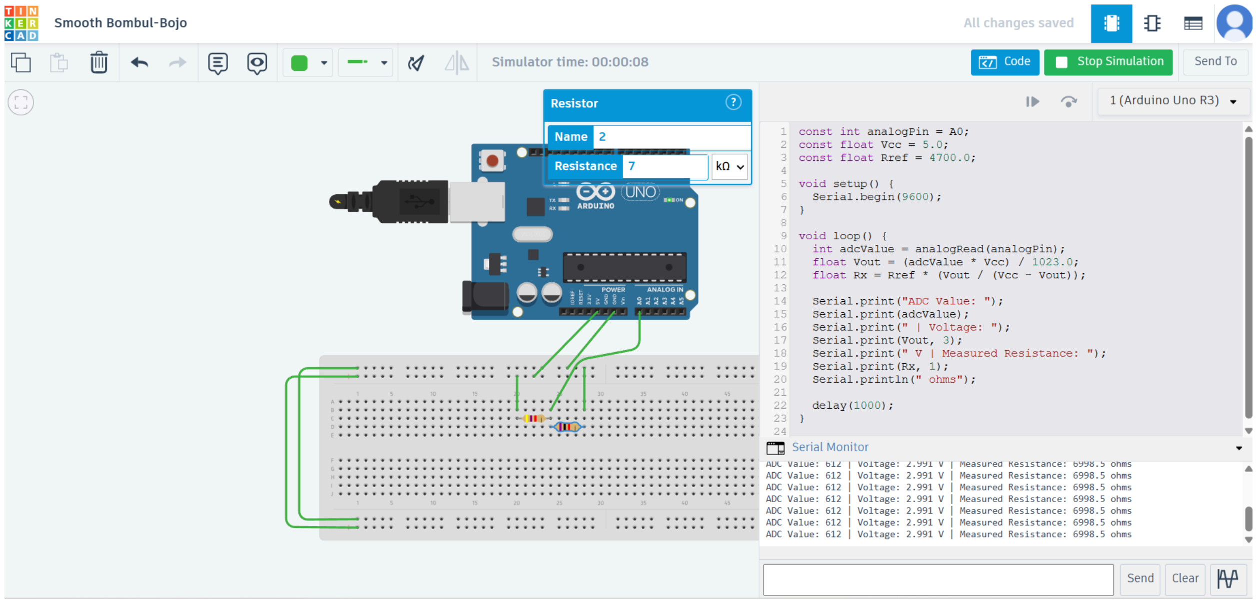
Absolute error $=|\mathrm{Rmeas}-\mathrm{Ractual}|=|R_{\mathrm{meas}}-R_{\mathrm{actual}}|$

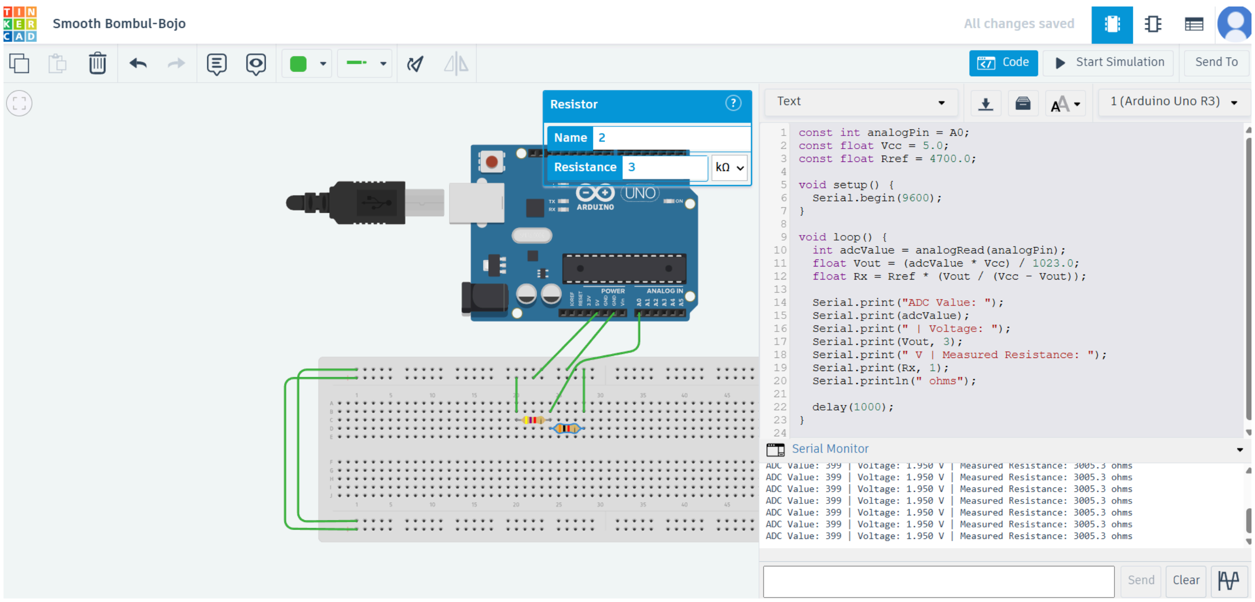
and

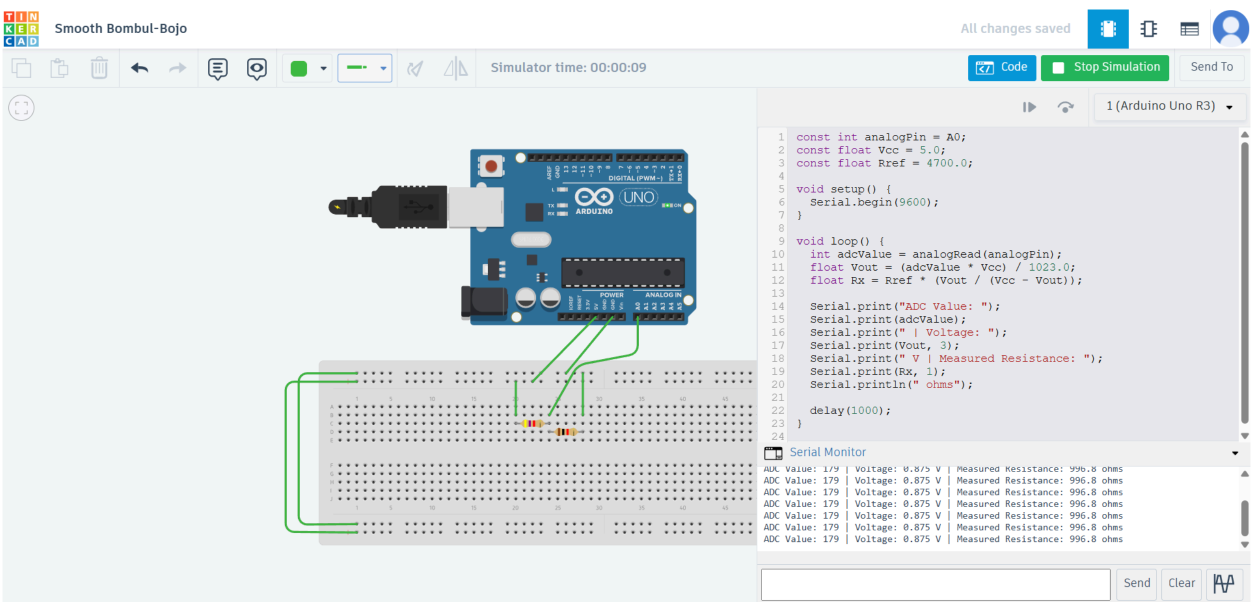
percent error $=\mathrm{absolute\ error}/\mathrm{Ractual}\times 100=R_{\mathrm{actual}}$

Reflection on measurement uncertainty

The ADC has 10-bit resolution, so each step is about 4.9 mV at a 5 V reference, which limits how precisely voltage and resistance can be measured. The actual 5 V supply and 4.7 kΩ reference resistor may deviate from their nominal values, introducing additional error. Breadboard contacts, wiring resistance, and electrical noise can slightly change the voltage seen at A0. Real resistors have tolerance (for example ±5% or ±1%), so their true value can naturally differ from the printed value even with perfect measurement. Overall, the measured resistances are extremely close to the nominal values, so any tiny differences fall within normal tolerance and expected measurement uncertainty.




Smooth Bombul-Bojo
All changes saved
Code
Start Simulation
Send To
Resistor
Name 2
Resistance 3 kΩ
Text
1 (Arduino Uno R3)
const int analogPin = A0;
const float Vcc = 5.0;
const float Rref = 4700.0;
void setup() {
Serial.begin(9600);
}
void loop() {
int adcValue = analogRead(analogPin);
float Vout = (adcValue * Vcc) / 1023.0;
float Rx = Rref * (Vout / (Vcc - Vout));
Serial.print("ADC Value: ");
Serial.print(adcValue);
Serial.print(" | Voltage: ");
Serial.print(Vout, 3);
Serial.print(" V | Measured Resistance: ");
Serial.print(Rx, 1);
Serial.println(" ohms");
delay(1000);
}
Serial Monitor
ADC Value: 399 | Voltage: 1.950 V | Measured Resistance: 3005.3 ohms
ADC Value: 399 | Voltage: 1.950 V | Measured Resistance: 3005.3 ohms
ADC Value: 399 | Voltage: 1.950 V | Measured Resistance: 3005.3 ohms
ADC Value: 399 | Voltage: 1.950 V | Measured Resistance: 3005.3 ohms
ADC Value: 399 | Voltage: 1.950 V | Measured Resistance: 3005.3 ohms
ADC Value: 399 | Voltage: 1.950 V | Measured Resistance: 3005.3 ohms
ADC Value: 399 | Voltage: 1.950 V | Measured Resistance: 3005.3 ohms
Send
Clear


Smooth Bombul-Bojo
All changes saved
Simulator time: 00:00:09
Code
Stop Simulation
Send To
1 (Arduino Uno R3)
const int analogPin = A0;
const float Vcc = 5.0;
const float Rref = 4700.0;
void setup() {
Serial.begin(9600);
}
void loop() {
int adcValue = analogRead(analogPin);
float Vout = (adcValue * Vcc) / 1023.0;
float Rx = Rref * (Vout / (Vcc - Vout));
Serial.print("ADC Value: ");
Serial.print(adcValue);
Serial.print(" | Voltage: ");
Serial.print(Vout, 3);
Serial.print(" V | Measured Resistance: ");
Serial.print(Rx, 1);
Serial.println(" ohms");
delay(1000);
}
Serial Monitor
ADC Value: 179 | Voltage: 0.875 V | Measured Resistance: 996.8 ohms
ADC Value: 179 | Voltage: 0.875 V | Measured Resistance: 996.8 ohms
ADC Value: 179 | Voltage: 0.875 V | Measured Resistance: 996.8 ohms
ADC Value: 179 | Voltage: 0.875 V | Measured Resistance: 996.8 ohms
ADC Value: 179 | Voltage: 0.875 V | Measured Resistance: 996.8 ohms
ADC Value: 179 | Voltage: 0.875 V | Measured Resistance: 996.8 ohms
ADC Value: 179 | Voltage: 0.875 V | Measured Resistance: 996.8 ohms
Send
Clear