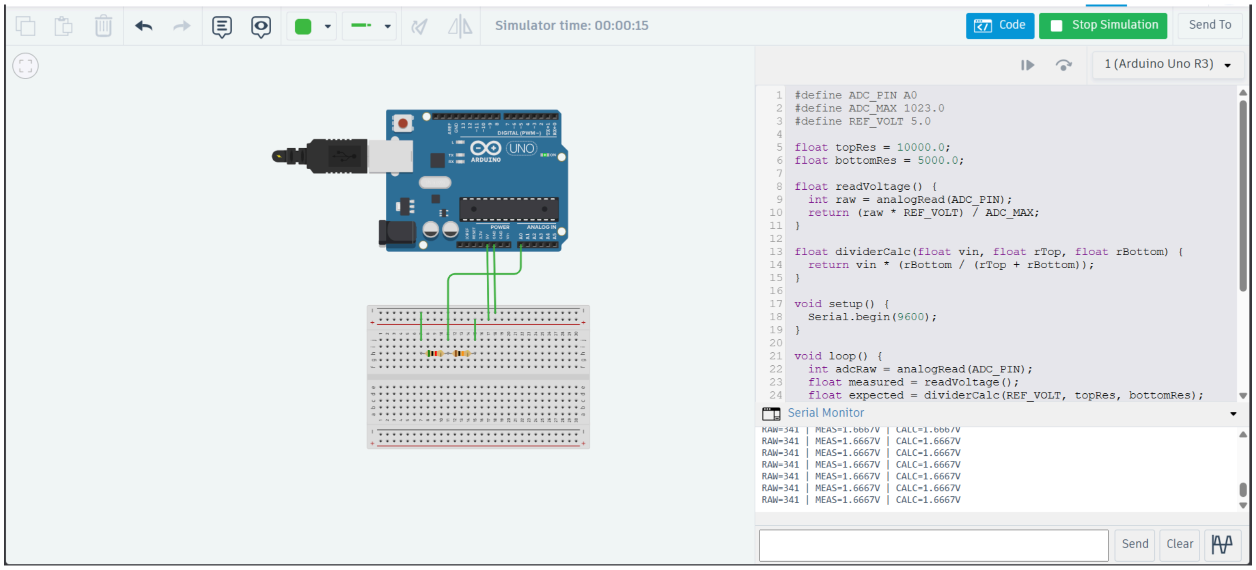
Smart Multimeter Using Microcontroller Systems — Assignment 1 (Ayush Muttepawar 240243)

Task A

1. Purpose:
A voltage divider is used to reduce a high voltage to a lower, safe voltage that a device can measure.
In measurement systems like a digital multimeter or Arduino, the input voltage is divided so it does not damage the ADC pin.

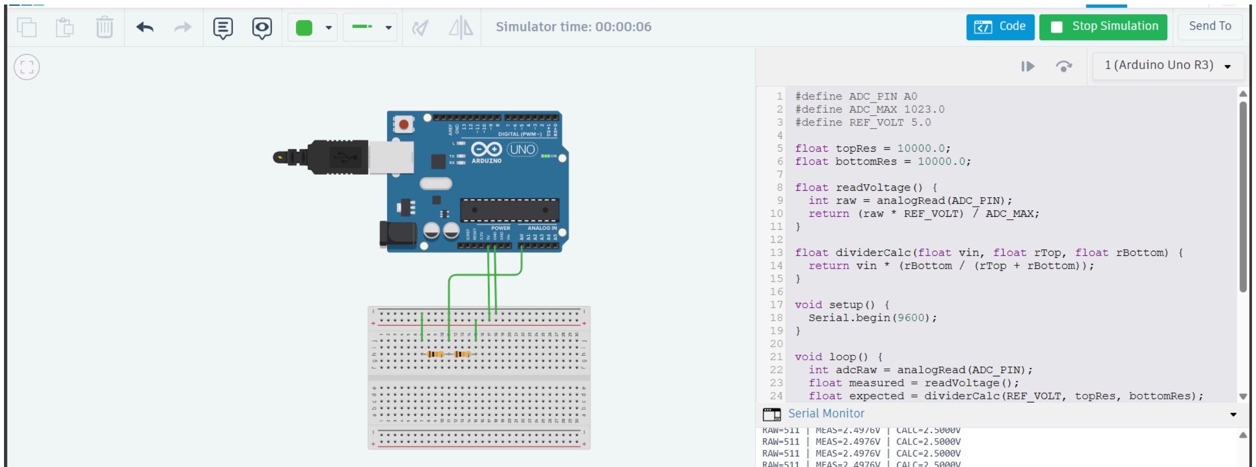
2. ADC reading-to-voltage conversion formula:
Vout = Vin * R2/(R1+R2)
Voltage = ADC Value * Vref/1023

3. Observations: error, noise, or unexpected results:
Measured voltages were close to theoretical values for all resistor combinations and small errors occurred due to ADC resolution and resistor tolerances.



```
#define ADC_PIN A0
#define ADC_MAX 1023.0
#define REF_VOLT 5.0

float topRes = 10000.0;
float bottomRes = 5000.0;

float readVoltage() {
  int raw = analogRead(ADC_PIN);
  return (raw * REF_VOLT) / ADC_MAX;
}

float dividerCalc(float vin, float rTop, float rBottom) {
  return vin * (rBottom / (rTop + rBottom));
}

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

void loop() {
  int adcRaw = analogRead(ADC_PIN);
  float measured = readVoltage();
  float expected = dividerCalc(REF_VOLT, topRes, bottomRes);
```

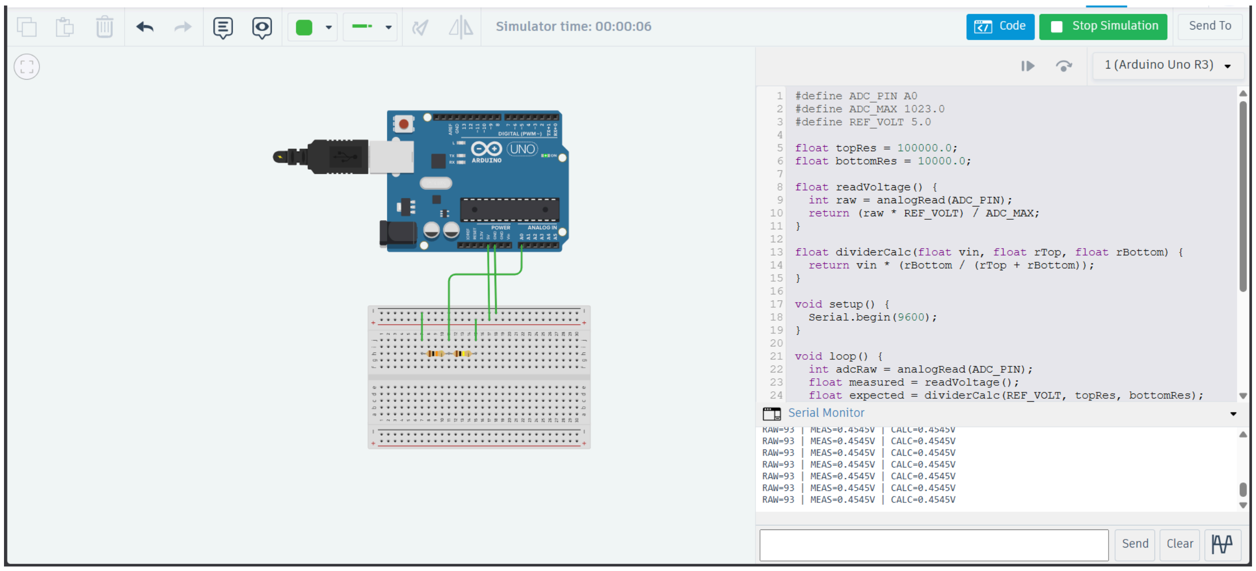
```
RAW=341 | MEAS=1.6667V | CALC=1.6667V
RAW=341 | MEAS=1.6667V | CALC=1.6667V
RAW=341 | MEAS=1.6667V | CALC=1.6667V
RAW=341 | MEAS=1.6667V | CALC=1.6667V
RAW=341 | MEAS=1.6667V | CALC=1.6667V
RAW=341 | MEAS=1.6667V | CALC=1.6667V
RAW=341 | MEAS=1.6667V | CALC=1.6667V
```



```
#define ADC_PIN A0
#define ADC_MAX 1023.0
#define REF_VOLT 5.0

float topRes = 10000.0;
float bottomRes = 10000.0;

float readVoltage() {
  int raw = analogRead(ADC_PIN);
  return (raw * REF_VOLT) / ADC_MAX;
}

float dividerCalc(float vin, float rTop, float rBottom) {
  return vin * (rBottom / (rTop + rBottom));
}

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

void loop() {
  int adcRaw = analogRead(ADC_PIN);
  float measured = readVoltage();
  float expected = dividerCalc(REF_VOLT, topRes, bottomRes);
```

```
RAW=511 | MEAS=2.4976V | CALC=2.5000V
RAW=511 | MEAS=2.4976V | CALC=2.5000V
RAW=511 | MEAS=2.4976V | CALC=2.5000V
RAW=511 | MEAS=2.4976V | CALC=2.5000V
```

Smart Multimeter Using Microcontroller Systems — Assignment 1 (Ayush Muttepawar 240243)



```
#define ADC_PIN A0
#define ADC_MAX 1023.0
#define REF_VOLT 5.0

float topRes = 100000.0;
float bottomRes = 10000.0;

float readVoltage() {
  int raw = analogRead(ADC_PIN);
  return (raw * REF_VOLT) / ADC_MAX;
}

float dividerCalc(float vin, float rTop, float rBottom) {
  return vin * (rBottom / (rTop + rBottom));
}

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

void loop() {
  int adcRaw = analogRead(ADC_PIN);
  float measured = readVoltage();
  float expected = dividerCalc(REF_VOLT, topRes, bottomRes);
```

```
RAW=93 | MEAS=0.4545V | CALC=0.4545V
RAW=93 | MEAS=0.4545V | CALC=0.4545V
RAW=93 | MEAS=0.4545V | CALC=0.4545V
RAW=93 | MEAS=0.4545V | CALC=0.4545V
RAW=93 | MEAS=0.4545V | CALC=0.4545V
RAW=93 | MEAS=0.4545V | CALC=0.4545V
RAW=93 | MEAS=0.4545V | CALC=0.4545V
```

R1	R2	Theoretical Vout	Measured Vout
10k	5k	1.6667	1.6667
10k	10k	2.5000	2.4976
100k	10k	0.4545	0.4545

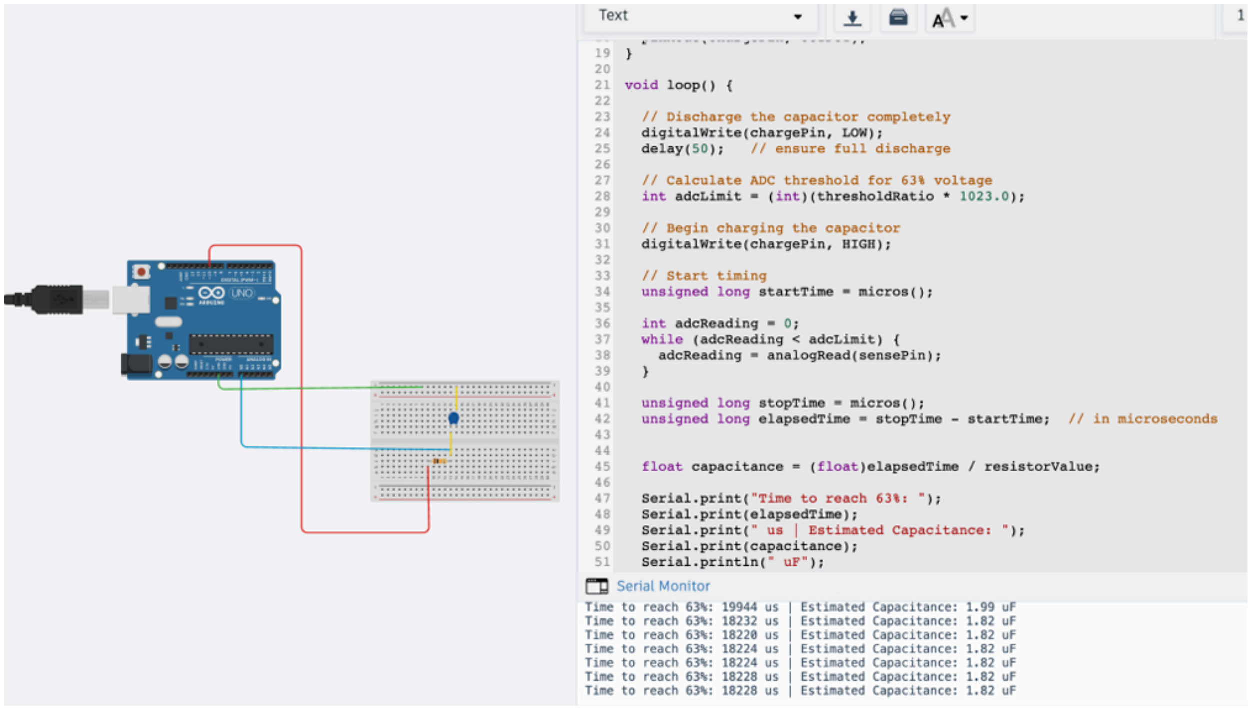
4. Code:

```
#define ADC_PIN A0
#define ADC_MAX 1023.0
#define REF_VOLT 5.0

float topRes = 100000.0;
float bottomRes = 10000.0;

float readVoltage() {
  int raw = analogRead(ADC_PIN);
  return (raw * REF_VOLT) / ADC_MAX;
}

float dividerCalc(float vin, float rTop, float rBottom) {
 return vin * (rBottom / (rTop + rBottom));
}

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

void loop() {
  int adcRaw = analogRead(ADC_PIN);
  float measured = readVoltage();
  float expected = dividerCalc(REF_VOLT, topRes, bottomRes);

  Serial.print("RAW=");
  Serial.print(adcRaw);
  Serial.print(" | MEAS=");
  Serial.print(measured, 4);
  Serial.print("V | CALC=");
  Serial.print(expected, 4);
  Serial.println("V");

  delay(600);
}
```

Smart Multimeter Using Microcontroller Systems — Assignment 1 (Ayush Muttepawar 240243)

Task B

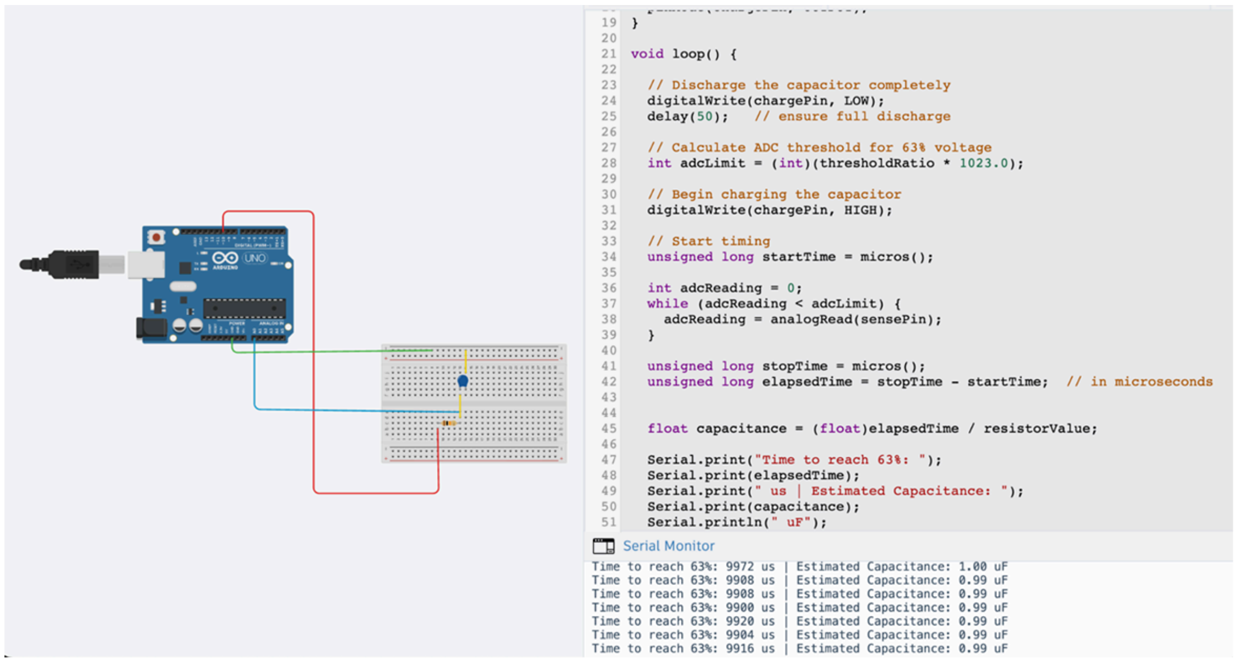
Smart Multimeter Using Microcontroller Systems — Assignment 1 (Ayush Muttepawar 240243)



```
}

void loop() {

  // Discharge the capacitor completely
  digitalWrite(chargePin, LOW);
  delay(50);   // ensure full discharge

  // Calculate ADC threshold for 63% voltage
  int adcLimit = (int)(thresholdRatio * 1023.0);

  // Begin charging the capacitor
  digitalWrite(chargePin, HIGH);

  // Start timing
  unsigned long startTime = micros();

  int adcReading = 0;
  while (adcReading < adcLimit) {
    adcReading = analogRead(sensePin);
  }

  unsigned long stopTime = micros();
  unsigned long elapsedTime = stopTime - startTime;  // in microseconds


  float capacitance = (float)elapsedTime / resistorValue;

  Serial.print("Time to reach 63%: ");
  Serial.print(elapsedTime);
  Serial.print(" us | Estimated Capacitance: ");
  Serial.print(capacitance);
  Serial.println(" uF");
```

Serial Monitor

```
Time to reach 63%: 19944 us | Estimated Capacitance: 1.99 uF
Time to reach 63%: 18232 us | Estimated Capacitance: 1.82 uF
Time to reach 63%: 18220 us | Estimated Capacitance: 1.82 uF
Time to reach 63%: 18224 us | Estimated Capacitance: 1.82 uF
Time to reach 63%: 18224 us | Estimated Capacitance: 1.82 uF
Time to reach 63%: 18228 us | Estimated Capacitance: 1.82 uF
Time to reach 63%: 18228 us | Estimated Capacitance: 1.82 uF
```



```
}

void loop() {

  // Discharge the capacitor completely
  digitalWrite(chargePin, LOW);
  delay(50);   // ensure full discharge

  // Calculate ADC threshold for 63% voltage
  int adcLimit = (int)(thresholdRatio * 1023.0);

  // Begin charging the capacitor
  digitalWrite(chargePin, HIGH);

  // Start timing
  unsigned long startTime = micros();

  int adcReading = 0;
  while (adcReading < adcLimit) {
    adcReading = analogRead(sensePin);
  }

  unsigned long stopTime = micros();
  unsigned long elapsedTime = stopTime - startTime;  // in microseconds


  float capacitance = (float)elapsedTime / resistorValue;

  Serial.print("Time to reach 63%: ");
  Serial.print(elapsedTime);
  Serial.print(" us | Estimated Capacitance: ");
  Serial.print(capacitance);
  Serial.println(" uF");
```

Serial Monitor

```
Time to reach 63%: 9972 us | Estimated Capacitance: 1.00 uF
Time to reach 63%: 9908 us | Estimated Capacitance: 0.99 uF
Time to reach 63%: 9908 us | Estimated Capacitance: 0.99 uF
Time to reach 63%: 9900 us | Estimated Capacitance: 0.99 uF
Time to reach 63%: 9920 us | Estimated Capacitance: 0.99 uF
Time to reach 63%: 9904 us | Estimated Capacitance: 0.99 uF
Time to reach 63%: 9916 us | Estimated Capacitance: 0.99 uF
```

1. RC Time Constant and 63% Significance

The RC time constant (τ) is the time taken by a capacitor to charge to about 63% of the supply voltage through a resistor. This happens because of the exponential charging behavior of capacitors. At one time constant, the voltage across the capacitor reaches 0.63 × V□, which makes it a reliable point for measuring capacitance using time.

Smart Multimeter Using Microcontroller Systems — Assignment 1 (Ayush Muttepawar 240243)

2. Sources of Error

Measurement errors occur due to resistor and capacitor tolerance, Arduino ADC resolution, electrical noise, and timing delays in the microcontroller. Simulation limitations in Tinkercad can also cause small deviations.

Smart Multimeter Using Microcontroller Systems — Assignment 1 (Ayush Muttepawar 240243)

Task C

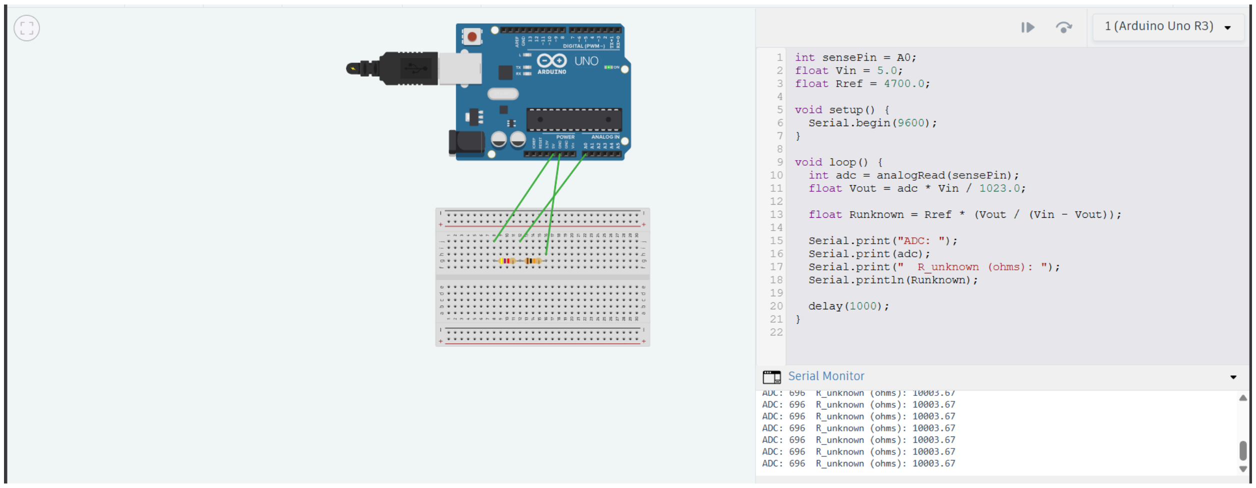
1. Formula:
R unknown = R ref * V out / (Vin - Vout)

2.Calculations:
Arduino reads voltage at the divider midpoint.
ADC value is converted to voltage.
The divider formula is rearranged to calculate unknown resistance.

3.Measurement Uncertainty:
Small errors occur due to resistor tolerance, Arduino ADC resolution, and electrical noise.
Accuracy reduces for very high or very low resistance values.

4. Actual Values:

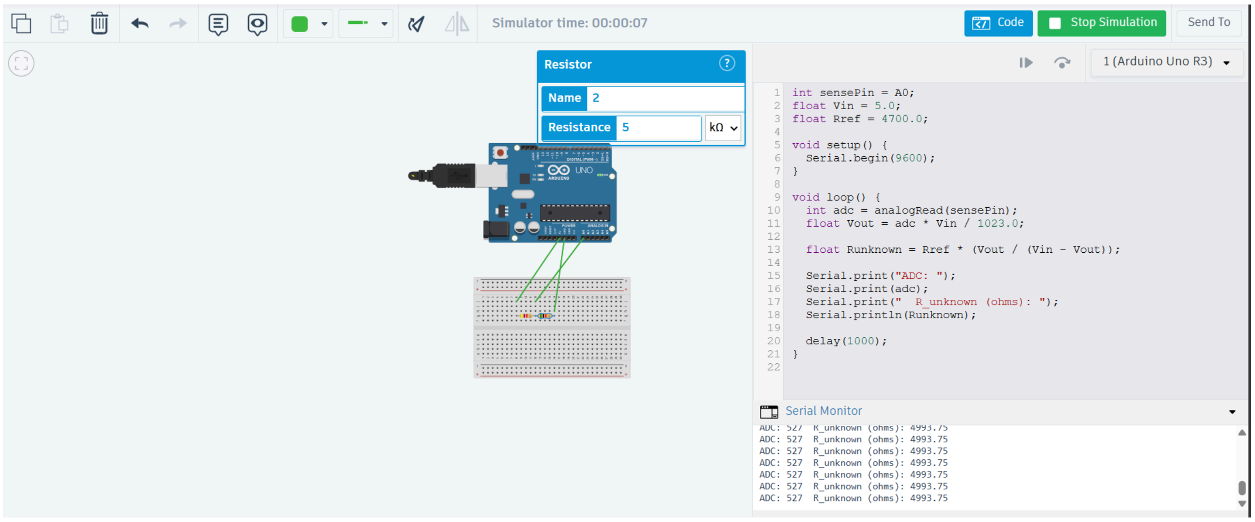
Actual Resistance	Measured Resistance
2 kΩ	1.996 kΩ
5 kΩ	4.994 kΩ
10 kΩ	10.004 kΩ



```
int sensePin = A0;
float Vin = 5.0;
float Rref = 4700.0;

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

void loop() {
  int adc = analogRead(sensePin);
  float Vout = adc * Vin / 1023.0;

  float Runknown = Rref * (Vout / (Vin - Vout));

  Serial.print("ADC: ");
  Serial.print(adc);
  Serial.print("  R_unknown (ohms): ");
  Serial.println(Runknown);

  delay(1000);
}
```

```
ADC: 696  R_unknown (ohms): 10003.67
ADC: 696  R_unknown (ohms): 10003.67
ADC: 696  R_unknown (ohms): 10003.67
ADC: 696  R_unknown (ohms): 10003.67
ADC: 696  R_unknown (ohms): 10003.67
ADC: 696  R_unknown (ohms): 10003.67
ADC: 696  R_unknown (ohms): 10003.67
```

Smart Multimeter Using Microcontroller Systems — Assignment 1 (Ayush Muttepawar 240243)


Simulator time: 00:00:07

Resistor

Name: 2

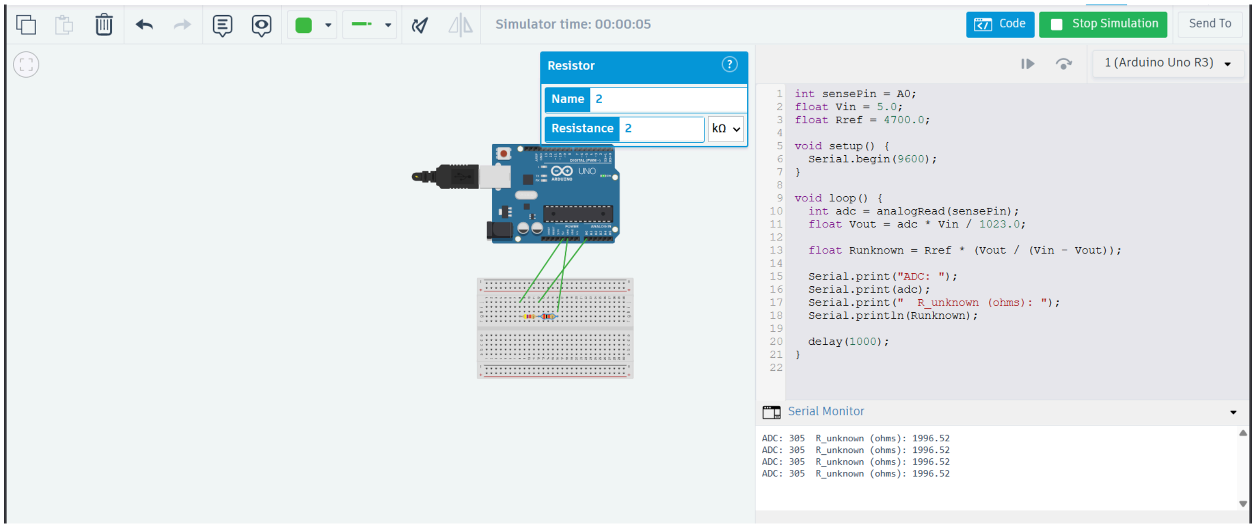
Resistance: 5 kΩ

1 (Arduino Uno R3)

```
int sensePin = A0;
float Vin = 5.0;
float Rref = 4700.0;

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

void loop() {
  int adc = analogRead(sensePin);
  float Vout = adc * Vin / 1023.0;

  float Runknown = Rref * (Vout / (Vin - Vout));

  Serial.print("ADC: ");
  Serial.print(adc);
  Serial.print("  R_unknown (ohms): ");
  Serial.println(Runknown);

  delay(1000);
}
```

Serial Monitor

```
ADC: 527  R_unknown (ohms): 4993.75
ADC: 527  R_unknown (ohms): 4993.75
ADC: 527  R_unknown (ohms): 4993.75
ADC: 527  R_unknown (ohms): 4993.75
ADC: 527  R_unknown (ohms): 4993.75
ADC: 527  R_unknown (ohms): 4993.75
ADC: 527  R_unknown (ohms): 4993.75
```


Simulator time: 00:00:05

Resistor

Name: 2

Resistance: 2 kΩ

1 (Arduino Uno R3)

```
int sensePin = A0;
float Vin = 5.0;
float Rref = 4700.0;

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

void loop() {
  int adc = analogRead(sensePin);
  float Vout = adc * Vin / 1023.0;

  float Runknown = Rref * (Vout / (Vin - Vout));

  Serial.print("ADC: ");
  Serial.print(adc);
  Serial.print("  R_unknown (ohms): ");
  Serial.println(Runknown);

  delay(1000);
}
```

Serial Monitor

```
ADC: 305  R_unknown (ohms): 1996.52
ADC: 305  R_unknown (ohms): 1996.52
ADC: 305  R_unknown (ohms): 1996.52
ADC: 305  R_unknown (ohms): 1996.52
```