Diy projects lab

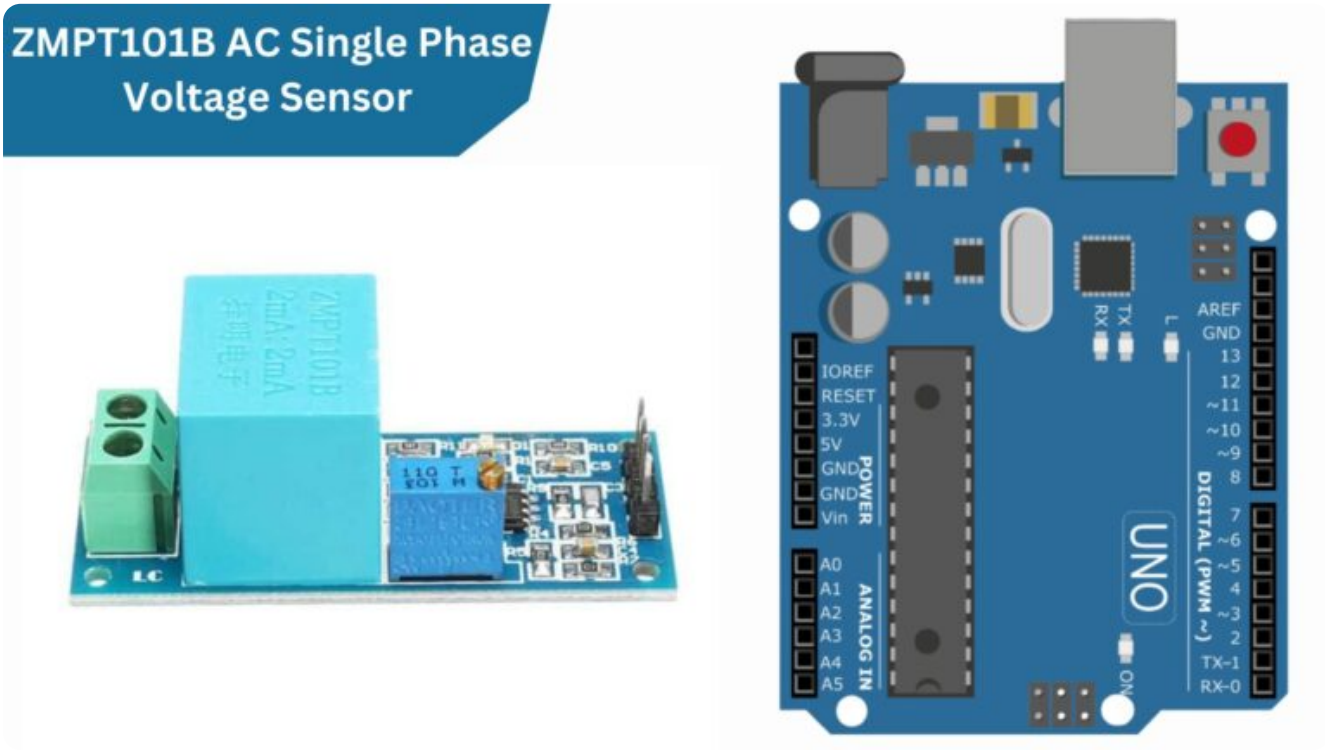
Home » How to use ZMPT101B Voltage Sensor Module with Arduino

ARDUINO PROJECTS

How to use ZMPT101B Voltage Sensor Module with Arduino

BY **ADMIN** — 05/07/2023 — UPDATED: 07/12/2023 NO COMMENTS

Share



In this tutorial, we will be hooking up a ZMPT101B Voltage Sensor Module with Arduino. This module can be used for AC or DC voltage measurements and provides both analog and digital outputs. We will be using the analog output in this tutorial. the module has a range of 0-250V and an accuracy of 1%.

We'll go over how to wire the sensor, how to set up the Arduino sketch, and how to read the sensor data. It can be used to measure the voltage in your home or office. checkout my previous post on **Measure AC Current Using Arduino And SCT-013 Sensor**

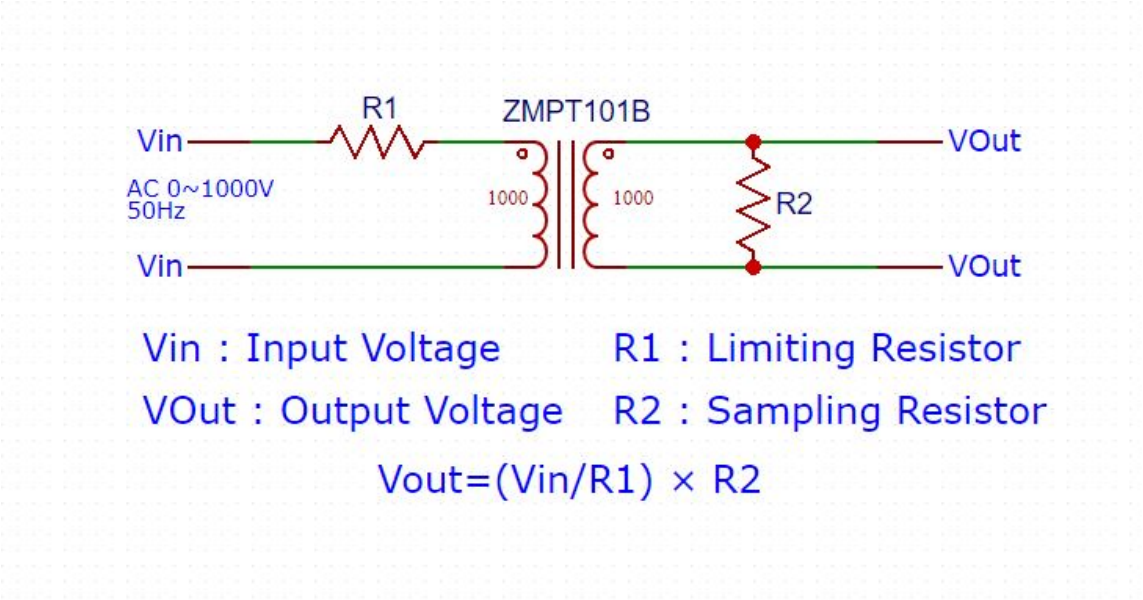
- ZMPT101B Voltage Sensor Module
- Arduino board
- Jumper wires
- Breadboard

ZMPT101B Voltage Transformer

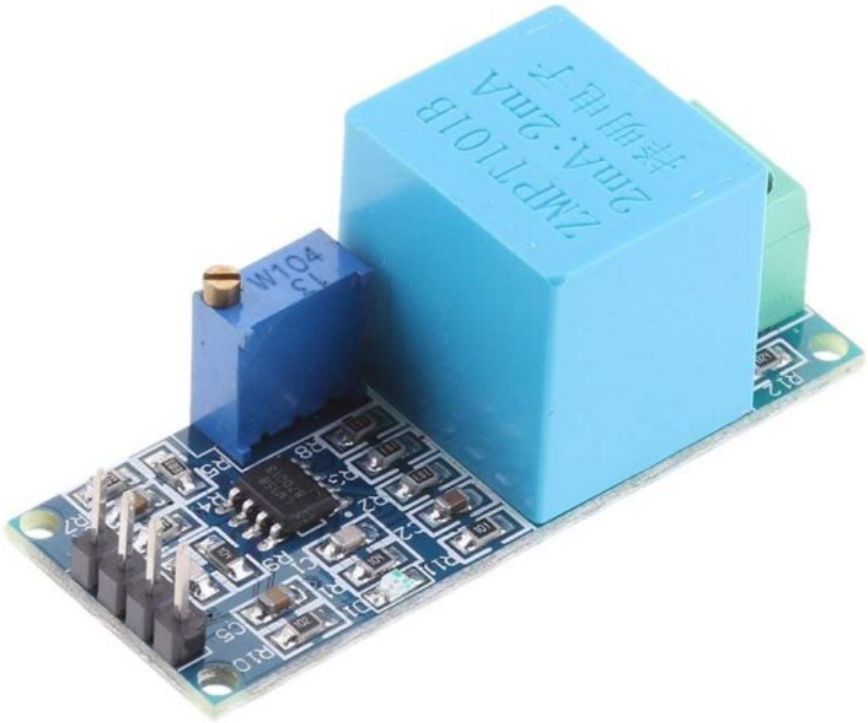
ZMPT 101B is a high-precision voltage transformer that makes it easy to monitor AC mains voltage up to 1**000 volts**.

This transformer holds up to **4kV** per breakdown voltage, the ratio of turns is **1: 1,** but this is a current transformer of **2mA: 2mA**. We feed it a current and remove the current. The input current is simply set by the resistor in series **R1**, and we use a sampling resistor **R2** in parallel to get the output voltage.

Schematic ZMPT101B Voltage Transformer



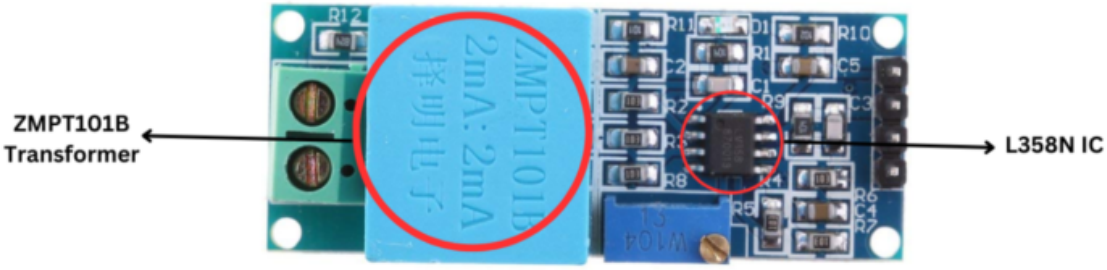
ZMPT101B Voltage Sensor Module



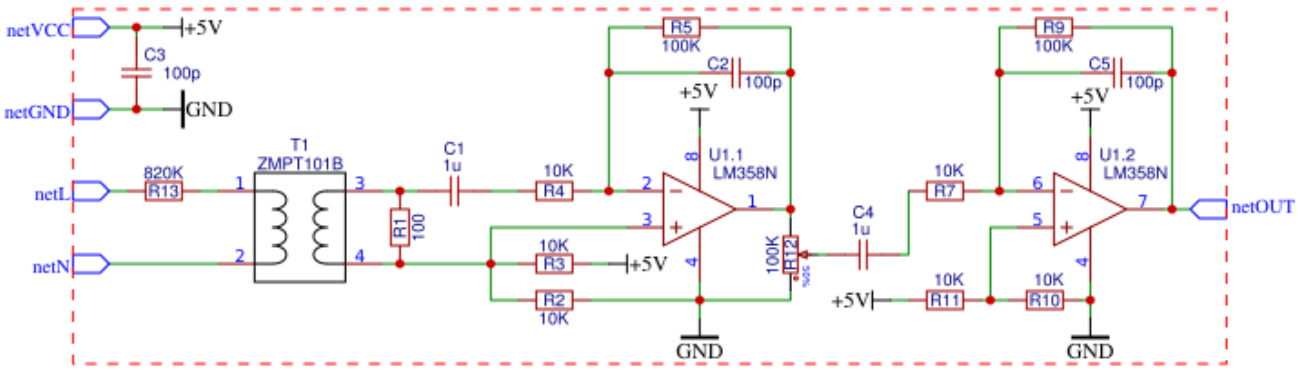
The ZMPT101B Voltage Sensor Module is a small electronic device that allows you to measure **AC voltage** using an Arduino or other microcontroller. It converts the **AC voltage into a signal** that the microcontroller can understand. With this module, you can easily measure AC voltage in your projects, It is commonly used in applications such as power monitoring, home automation, smart energy meter management, and system analysis.

ZMPT101B is the best choice to measure the AC voltage using Arduino/ESP8266/Raspberry Pi. In many electrical projects, the engineer directly deals with measurements with few basic requirements like

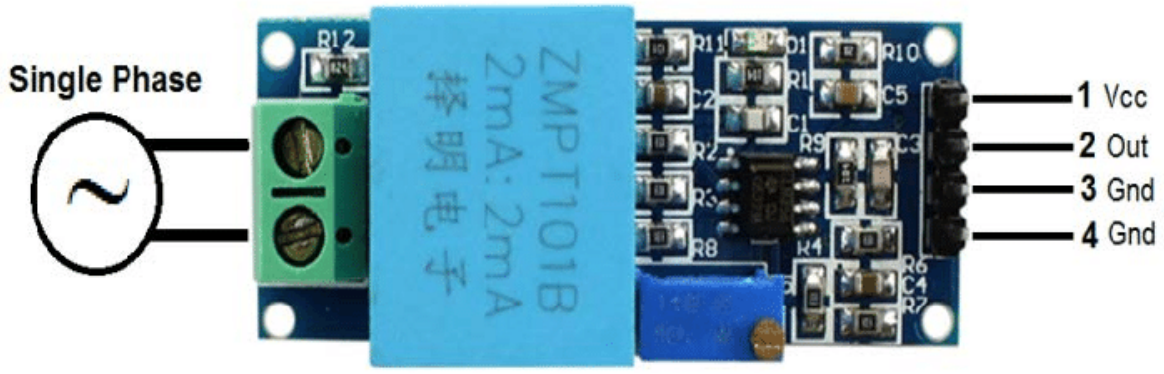
- High galvanic isolation
- Wide Range
- High accuracy
- Good Consistency



The ZMPT101B module is built with an **LM358N IC** chip, a some **resistors**, and few **capacitor** that helps reduce noise, or unwanted electrical signals.



Pinout of ZMPT101B Module



3. **GND**
4. **GND**

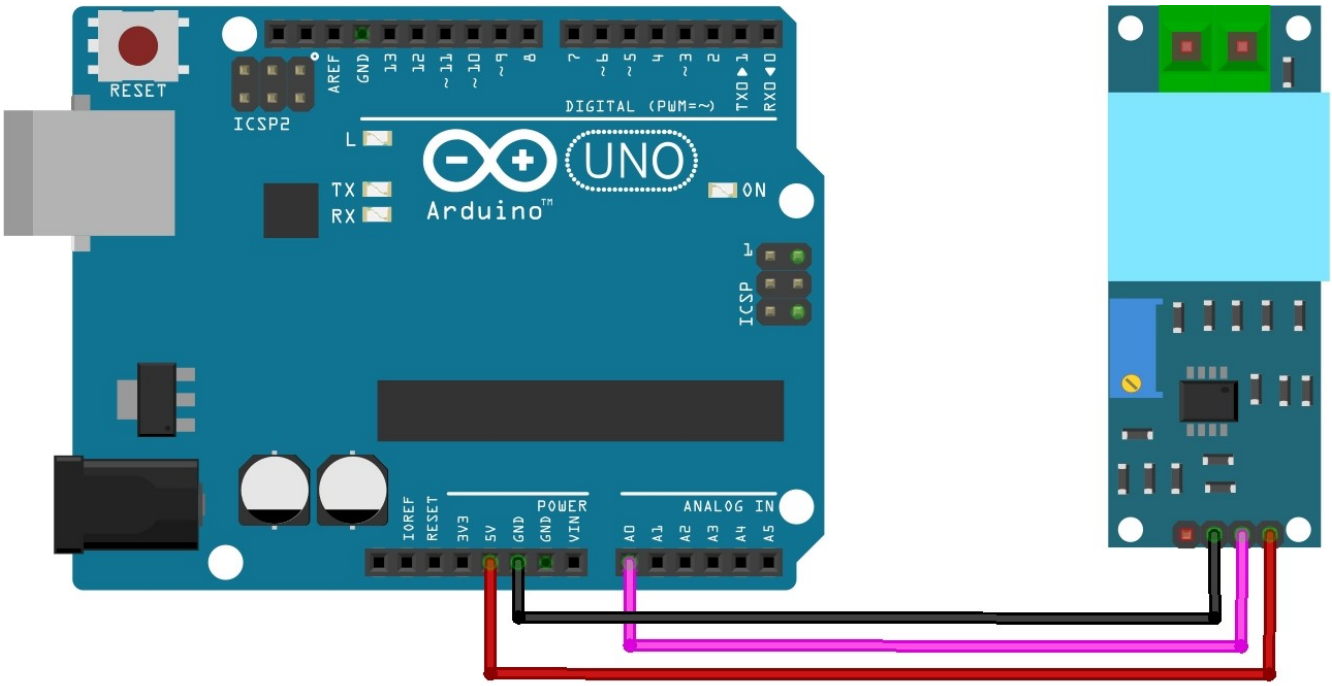
Specifications of ZMPT101B Module

- **Operating Voltage**: 5V
- **Maximum Input Voltage**: 250V (AC)
- **Turn Ratio**: 1:1
- **Input Current Range**: 2mA –10mA
- **Operating temperature** : 40°C ~ + 70°C
- **Output Voltage**: Varies based on input voltage and current
- **Breakdown Voltage**: Up to 4kV
- **Accuracy**: High precision for voltage measurements
- Compatible with Arduino

Applications

- Electrical energy meters
- AC Voltage measurement
- Household electrical equipment
- Industrial apparatuses
- Electrical testing equipment

Here is a connection diagram that can be used to interface the sensor with the Arduino



Connect the VCC, GND,& OUT pin of ZMPT101B to 5V, GND, & A0 of Arduino respectively.

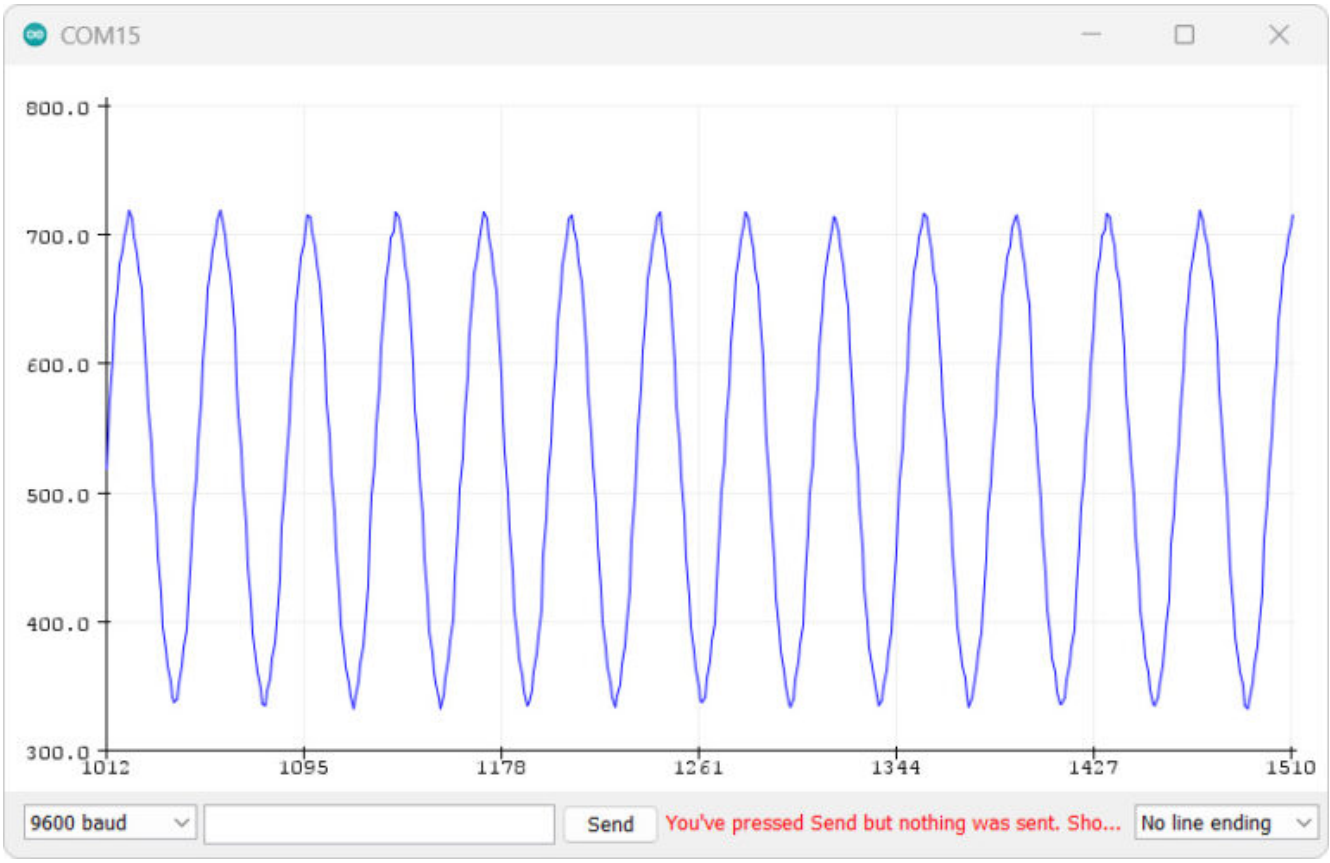
AC Voltage Sensor Sample Program

```
7  Serial.println(analogRead(A0));
8  delay(100);
9  }
```

After uploading the code to your Arduino board, open the Serial Plotter from the Tools menu in the Arduino IDE. When nothing is connected to the module inputs, you should see a value of around 512 (2.5 volts) on the Serial Plotter. If you apply **220V AC** to the input, you should see a sinusoidal voltage diagram on the Serial Plotter.

The Single Phase AC Voltage Module provides an analog output ranging from 0 to 1023, representing the measured voltage.

Turn the potentiometer on the module to get the most perfect sine wave possible, as in the following picture.



Note: Make sure the sine wave on the serial plotter is fully visible, including the

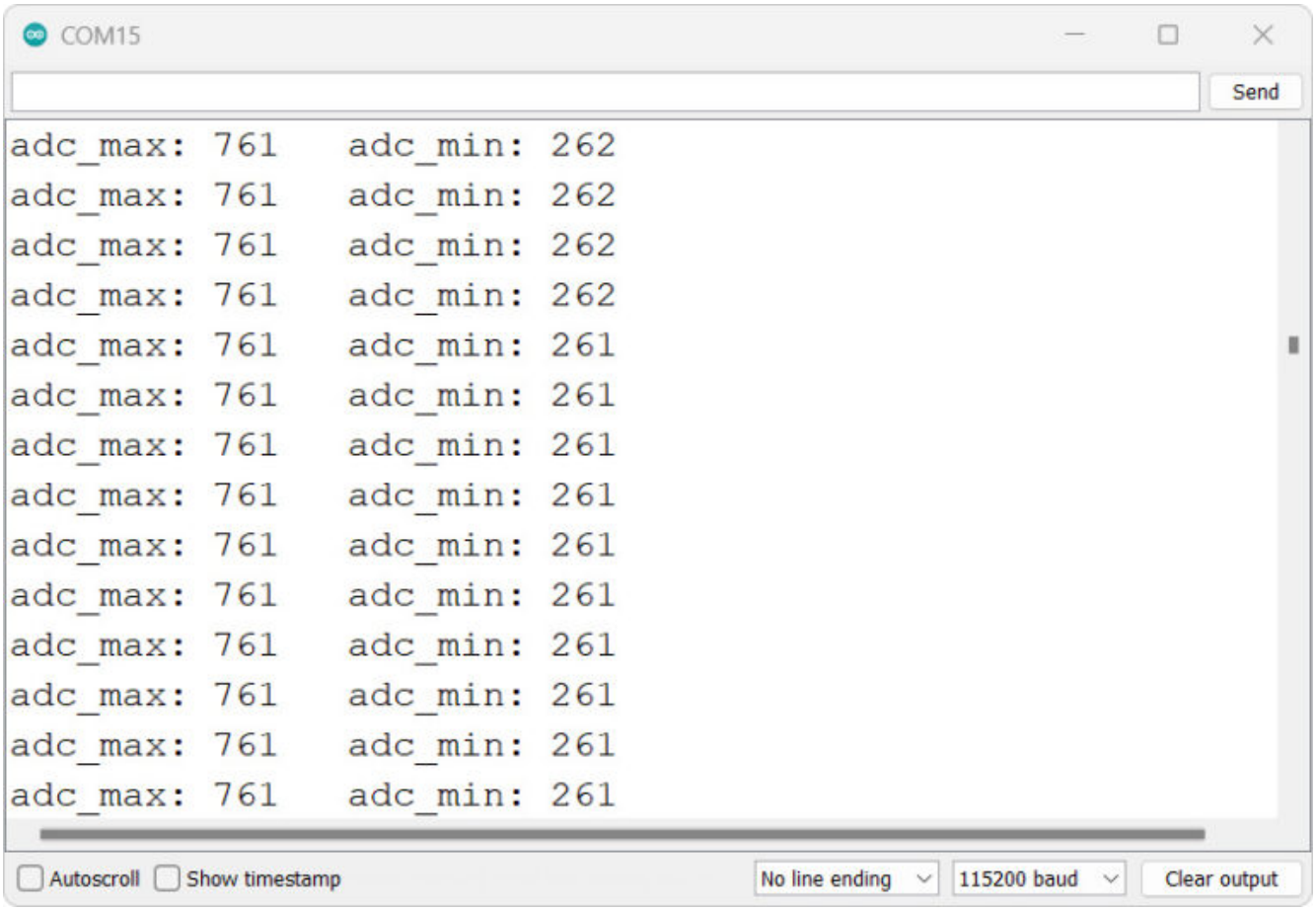
Source Code

Here is a sample code for interfacing the ZMPT101B Voltage Sensor module with Arduino.

Ac_Volt_Cal

```
int adc_max = 0;        // Variable to store the maximum sensor value
int adc_min = 1023;     // Variable to store the minimum sensor value
long tiempo_init;       // Variable to store the initial time

void setup() {
  Serial.begin(115200);   // Initialize the serial communication
  tiempo_init = millis(); // Get the current time in milliseconds
}

void loop() {
  if ((millis() - tiempo_init) > 500) { // Check if 500 milliseconds have passe
    adc_max = 0;          // Reset the maximum sensor value
    adc_min = 1023;       // Reset the minimum sensor value
    tiempo_init = millis(); // Update the initial time
  }

  int sensorValue = analogRead(A0); // Read the analog input from pin A0

  if (sensorValue > adc_max) {
    adc_max = sensorValue; // Update the maximum value if a new maximum is foun
  } else if (sensorValue < adc_min) {
    adc_min = sensorValue; // Update the minimum value if a new minimum is foun
  }
```

Upload the program: **Ac_Volt_Cal**

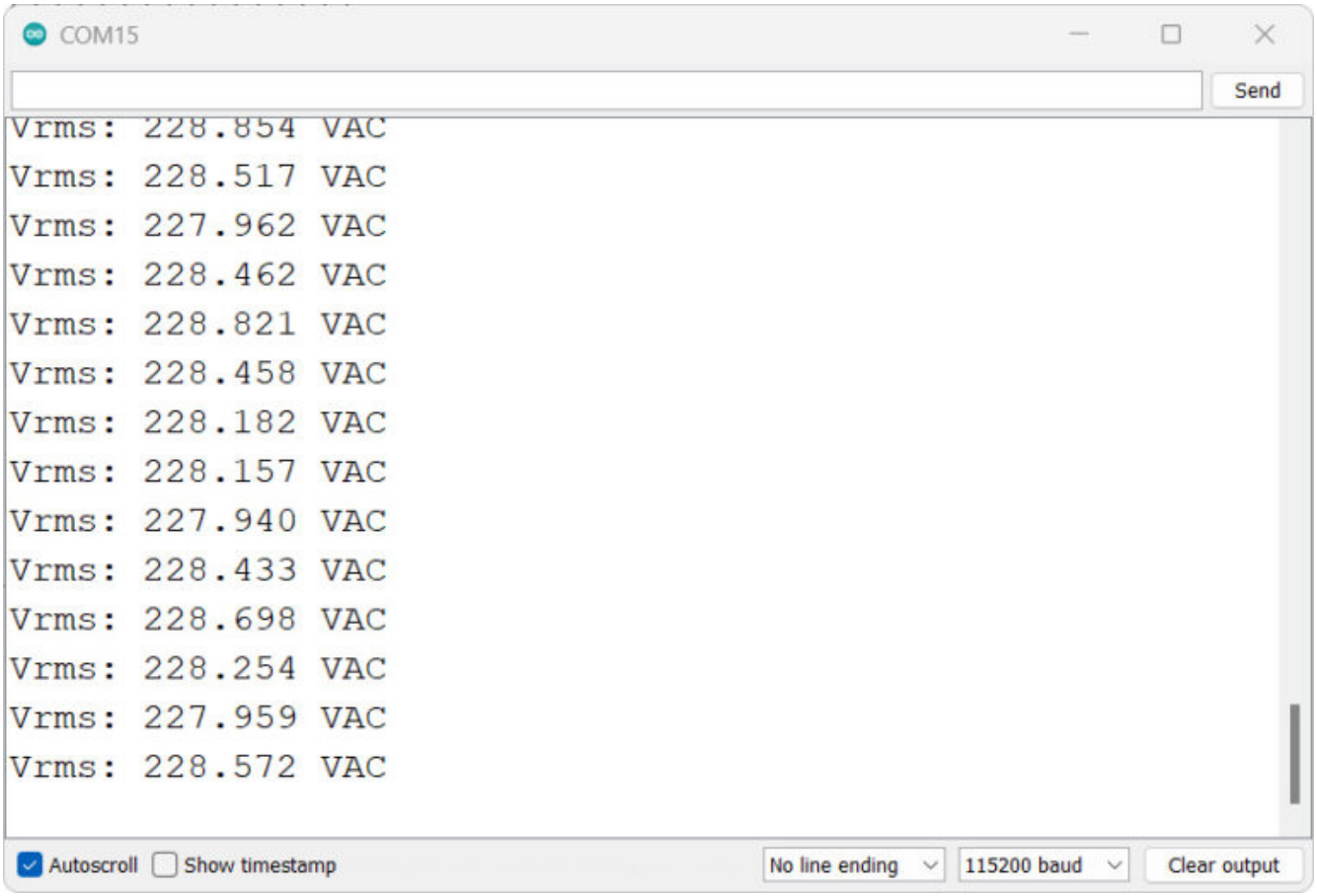


- This program is used to obtain calibration values for the sensor.
- Adjust the potentiometer to set the values of adc_max and adc_min .
- It is recommended to turn the potentiometer counterclockwise until **adc_max** reaches **700+.**
- Take note of the values adc_max and adc_min .
- Use a multimeter to measure the AC voltage and record it.

```
float volt_multi_p;                 // Peak voltage
float volt_multi_n;                 // Negative peak voltage

void setup() {
  Serial.begin(115200);

  volt_multi_p = volt_multi * 1.4142;    // Peak voltage = RMS voltage * 1.4142 (
  volt_multi_n = -volt_multi_p;          // Negative peak voltage
}

void loop() {
  float volt_rms = get_voltage();        // Root Mean Square voltage (V-RMS)

  Serial.print("Vrms: ");
  Serial.print(volt_rms, 3);
  Serial.println(" VAC");

  // Delay for a certain interval if needed
  //delay(100);
}

float get_voltage() {
  float adc_sample;
  float volt_inst = 0;
  float sum = 0;
  float volt;
  long init_time = millis();
  int N = 0;

  while ((millis() - init_time) < 500) {   // Duration of 0.5 seconds (Approxim
    adc_sample = analogRead(A0);             // Sensor voltage
    volt_inst = map(adc_sample, adc_min, adc_max, volt_multi_n, volt_multi_p);
    sum += sq(volt_inst);                    // Sum of Squares
    N++;
    delay(1);
  }

  //Serial.print("N: ");
  //Serial.println(N);

  volt = sqrt(sum / N);                    // RMS equation
  return volt;
}
```

Load the program: **Ac_Volt_inst**

- Set the values of **adc_max** and **adc_min** obtained from the calibration.
- Adjust the **volt_multi** variable to match the measured AC voltage (default is 222VAC).



Please note that this is just a basic overview. I will continue working on ZMPT101B and share my thoughts once I make more progress. I'm also interested to see if anyone else has made any new discoveries with this sensor module. Remember, expertise is for everyone!