

Part 3

lesson

1

Thermometer

Overview

In this lesson, you will use an OLED display to show the temperature.

Component Required:

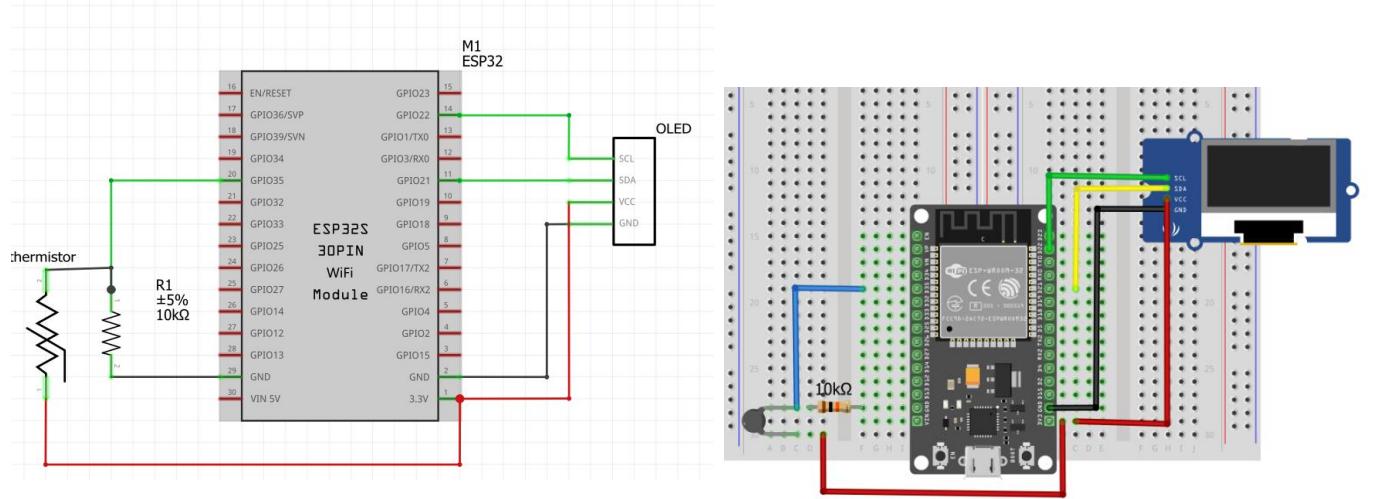
- (1) x Elegoo ESP32
- (1) x OLED Display
- (1) x 10k ohm resistor
- (1) x Thermistor
- (1) x Potentiometer
- (2) x 400 tie-points Breadboard
- (18) x M-M wires (Male to Male jumper wires)

Component Introduction

Thermistor

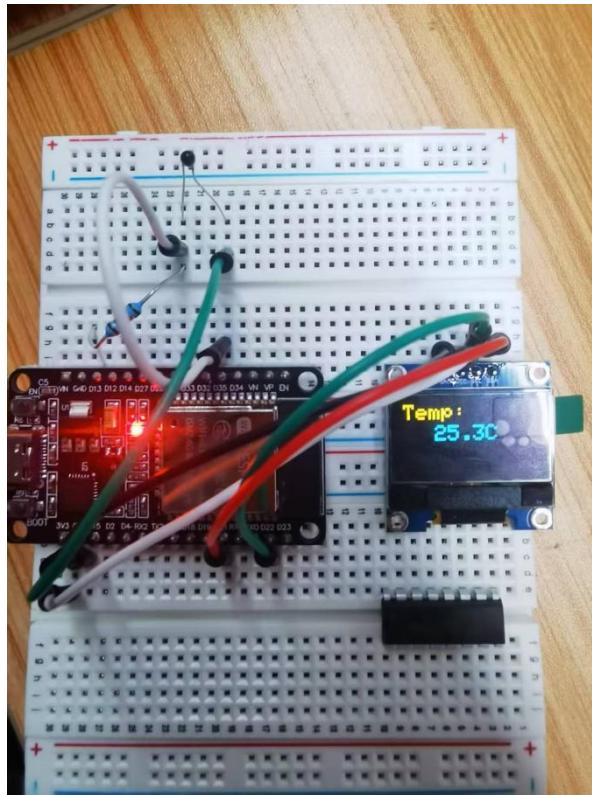
A thermistor is a thermal resistor - a resistor that changes its resistance with temperature. Technically, all resistors are thermistors - their resistance changes slightly with temperature - but the change is usually very small and difficult to measure. Thermistors are made so that the resistance changes drastically with temperature. It can be 100 Ohms or more, per degree of change.

There are two kinds of thermistors, NTC (negative temperature coefficient) and PTC (positive temperature coefficient). In general, you will see NTC sensors used for temperature measurement. PTC's are often used as resettable fuses - an increase in temperature increases the resistance which means that as more current passes through them, they heat up and 'choke back' the current, quite handy for protecting circuits!



Connection Schematic

Wiring diagram



Demo Effect

Code

After wiring, please open the program in the code folder- Thermometer and click UPLOAD to upload the program. See Lesson 5 of part 1 for details about program uploading if there are any errors.

Load it up onto your ESP32 and you should find that warming the temperature sensor by putting your finger on it will increase the temperature reading.

This makes things easier if you decide to change which pins you use.

In the 'loop' function there are now two interesting things going on. Firstly we have to convert the analogue from the temperature sensor into an actual temperature, and secondly we have to work out how to display them.

First of all, let's verify that the OLED is working properly and configure its basic parameters and review some of the built-in OLED functions, as shown in the figure on the right.

It's worth noting that the `setTextColor()` function has no effect on the OLED used in this project: the top row is hard-wired yellow, while the lower three rows are fixed blue.

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

    // Initialize I2C with SDA=21 and SCL=22
    Wire.begin(21, 22);

    // Initialize the OLED display
    if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
        Serial.println(F("SSD1306 allocation failed"));
        while (1);
    }

    delay(2000);
    display.clearDisplay();
    display.setTextSize(1);
    display.setTextColor(WHITE);
    display.setCursor(15, 0);
}
```

- `display.clearDisplay()` – all pixels are off
- `display.drawPixel(x, y, color)` – plot a pixel in the x,y coordinates
- `display.setTextSize(n)` – set the font size, supports sizes from 1 to 8
- `display.setCursor(x, y)` – set the coordinates to start writing text
- `display.print("message")` – print the characters at location x,y
- `display.display()` – call this method for the changes to make effect

Next, we process the temperature-sensor readings. To suppress noise-induced fluctuations, we sample the probe multiple times and average the results for a more stable value.

```
// --- 1. Take multiple ADC samples for noise reduction ---
int tempReading = 0;
for (int i = 0; i < 5; i++) {
    int val = analogRead(tempPin);

    // Constrain ADC reading to avoid extreme outliers (0 or 4095)
    tempReading += constrain(val, 10, 4085);
    delay(3);
}
tempReading /= 5; // Average value
```

The code is responsible for Converts the ESP32's 12-bit ADC reading (0-4095) into the NTC thermistor's resistance (using the voltage divider principle, with a known seriesResistor as the fixed resistor in the circuit).and Ensures the calculated NTC resistance is positive (avoids mathematical errors like log(0) or log(negative)).

```
float ratio = (4095.0 / tempReading) - 1.0;
float ntcResistance = seriesResistor * ratio;

// Safety check: prevent log(0) or log of negative values
if (ntcResistance <= 0) {
    return -1.0; // Error flag
}
```

It applies the Steinhart-Hart equation—an industry-standard model for NTC thermistors—to translate the measured resistance into temperature (Kelvin), converts the result to Celsius, and clamps the value within a sensible window of -10 °C to 80 °C to discard physically impossible readings.

```
double lnR = log(ntcResistance);
double tempK = 1.0 / (
    0.001129148 +
    (0.000234125 + 0.0000000876741 * lnR * lnR) * lnR
);

float tempC = tempK - 273.15; // Convert Kelvin → Celsius

// Limit temperature to a realistic range (-10°C to 80°C)
return constrain(tempC, -10.0, 80.0);
```

Troubleshooting Tips

If the device does not behave as shown in the reference photo after flashing, please verify the following:

Wiring

I²C lines: SDA → GPIO 21, SCL → GPIO 22

Temperature-probe data → GPIO 34

Power

GND and VCC (3.3 V or 5 V as required) firmly connected and not reversed

If parts of the text appear “missing” when you position it, don’t worry—this is just an optical illusion caused by the OLED’s fixed yellow/blue color boundary. Simply shift the text one line up or down and the distortion will disappear.

