

Part 15

-

1-Wire DS18B20

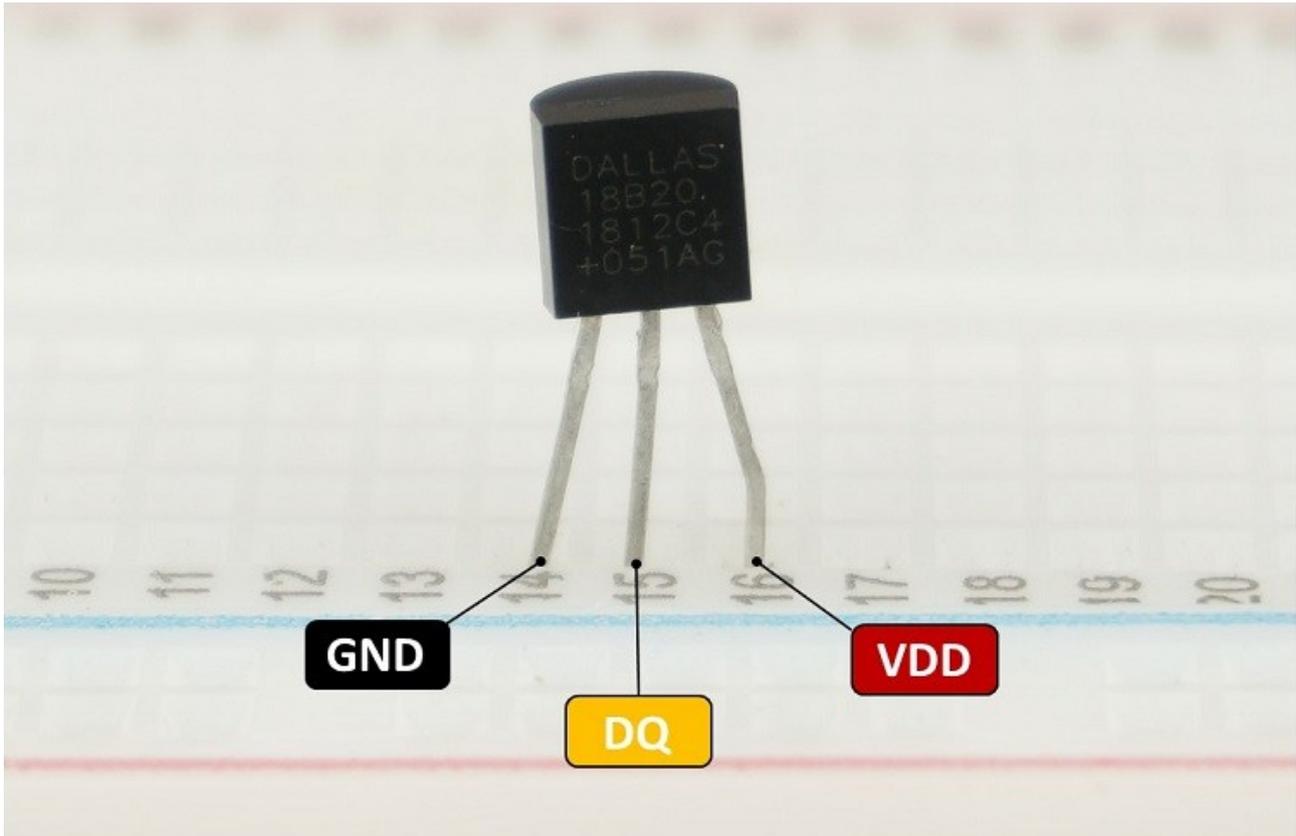
ESP32 DS18B20 Temperature Sensor

This is an in-depth guide for the DS18B20 temperature sensor with ESP32 using Arduino IDE. We'll show you how to wire the sensor, install the required libraries, and write the code to get the sensor readings from one and multiple sensors. Finally, we'll build a simple web server to display the sensor readings.

Introducing DS18B20 Temperature Sensor

The DS18B20 temperature sensor is a one-wire digital temperature sensor. This means that it just requires one data line (and GND) to communicate with your ESP32.

It can be powered by an external power supply or it can derive power from the data line (called "parasite mode"), which eliminates the need for an external power supply.



Each DS18B20 temperature sensor has a unique 64-bit serial code. This allows you to wire multiple sensors to the same data wire. So, you can get temperature from multiple sensors using just one GPIO.

The DS18B20 temperature sensor is also available in waterproof version.

Here's a summary of the most relevant specs of the DS18B20 temperature sensor

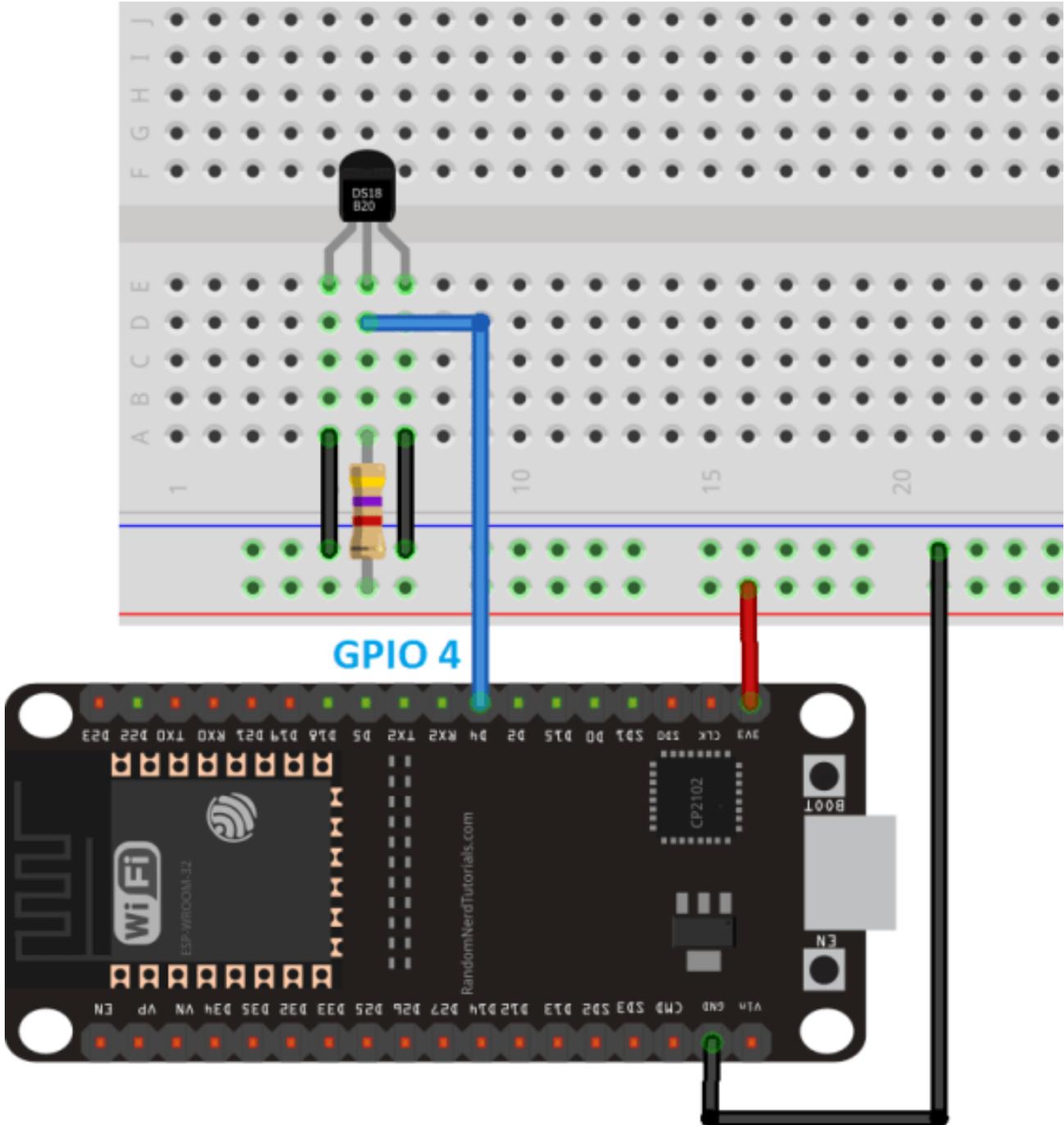
- Communicates over one-wire bus communication
- Power supply range: 3.0V to 5.5V
- Operating temperature range: -55°C to +125°C
- Accuracy +/-0.5 °C (between the range -10°C to 85°C)

You can use the preceding links or go directly to [MakerAdvisor.com/tools](https://www.makeradvisor.com/tools) to find all the parts for your projects at the best price!

How to connect

As mentioned previously, the DS18B20 temperature sensor can be powered through the VDD pin (normal mode), or it can derive its power from the data line (parasite mode). You can choose either modes. If you're using an ESP32 follow one of these two schematic diagrams. Our preference goes to normal mode.

Parasite Mode



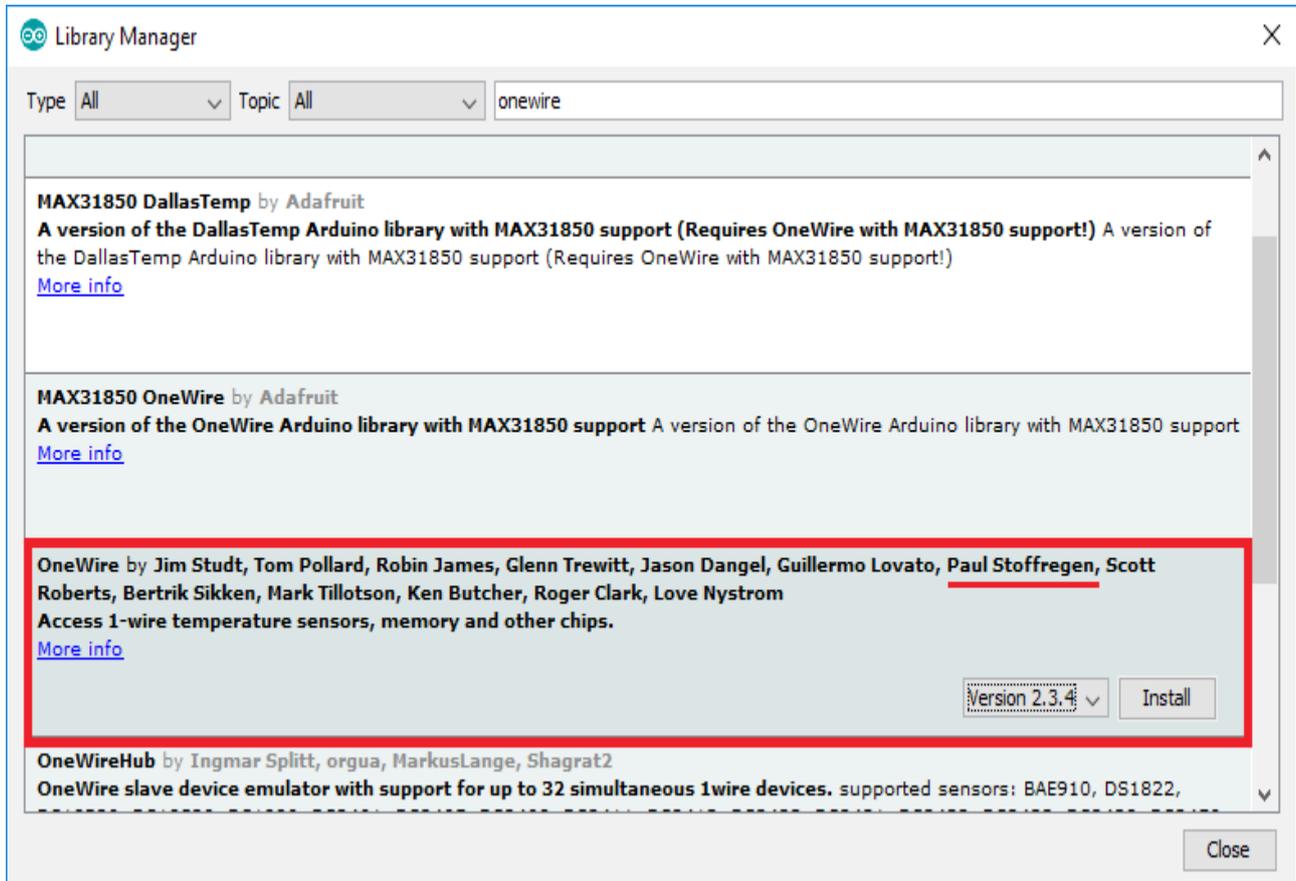
DS18B20 using Arduino IDE

Installing Libraries

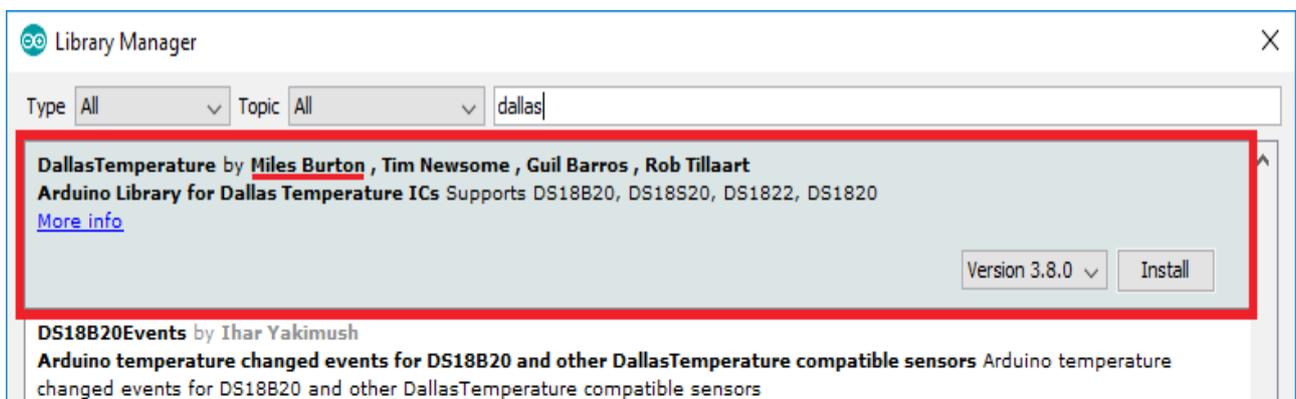
To interface with the DS18B20 temperature sensor, you need to install the One-Wire library and the Dallas Temperature library.

Open your Arduino IDE and go to Sketch > Include Library > Manage Libraries. The Library Manager should open.

Type "onewire" in the search box and install OneWire library.



Then, search for "Dallas" and install DallasTemperature library by Miles Burton.



After installing the libraries, restart your Arduino IDE.

Getting Temperature from single DS18B20 Temperature Sensor

Code

You can upload the code below to the ESP32. The following code reads temperature from the DS18B20 temperature sensor and displays the readings on the Arduino IDE Serial Monitor.

```
#include <OneWire.h>
#include <DallasTemperature.h>

// GPIO where the DS18B20 is connected to
const int oneWireBus = 4;

// Setup a oneWire instance to communicate with any OneWire devices
OneWire oneWire(oneWireBus);

// Pass our oneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);

void setup() {
  // Start the Serial Monitor
  Serial.begin(115200);
  // Start the DS18B20 sensor
  sensors.begin();
}

void loop() {
  sensors.requestTemperatures();
  float temperatureC = sensors.getTempCByIndex(0);
  float temperatureF = sensors.getTempFByIndex(0);
  Serial.print(temperatureC);
  Serial.println("°C");
  Serial.print(temperatureF);
  Serial.println("°F");
  delay(5000);
}
```

There are many different ways to get the temperature from DS18B20 temperature sensors. However, if you're using just one single sensor, this is one of the easiest and simplest ways.

How the Code Works

Start by including the `OneWire` and the `DallasTemperature` libraries.

```
#include <OneWire.h>
#include <DallasTemperature.h>
```

Create the instances needed for the temperature sensor. The temperature sensor is connected to GPIO 4.

```
// GPIO where the DS18B20 is connected to
const int oneWireBus = 4;

// Setup a oneWire instance to communicate with any OneWire devices
OneWire oneWire(oneWireBus);

// Pass our oneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
```

In the `setup()`, initialize the Serial Monitor at a baud rate of 115200.

```
erial.begin(115200);
```

Initialize the DS18B20 temperature sensor:

```
sensors.begin();
```

Before actually getting the temperature, you need to call the `requestTemperatures()` method.

```
sensors.requestTemperatures();
```

Then, get the temperature in Celsius by using the `getTempCByIndex()` method as shown below:

```
float temperatureC = sensors.getTempCByIndex(0);
```

Or use the `getTempFByIndex()` to get the temperature in Fahrenheit.

```
float temperatureF = sensors.getTempFByIndex(0);
```

The `getTempCByIndex()` and the `getTempFByIndex()` methods accept the index of the temperature sensor. Because we're using just one sensor its index is 0. If you want to read more than one sensor, you use index 0 for one sensor, index 1 for other sensor and so on. Finally, print the results in the Serial Monitor.

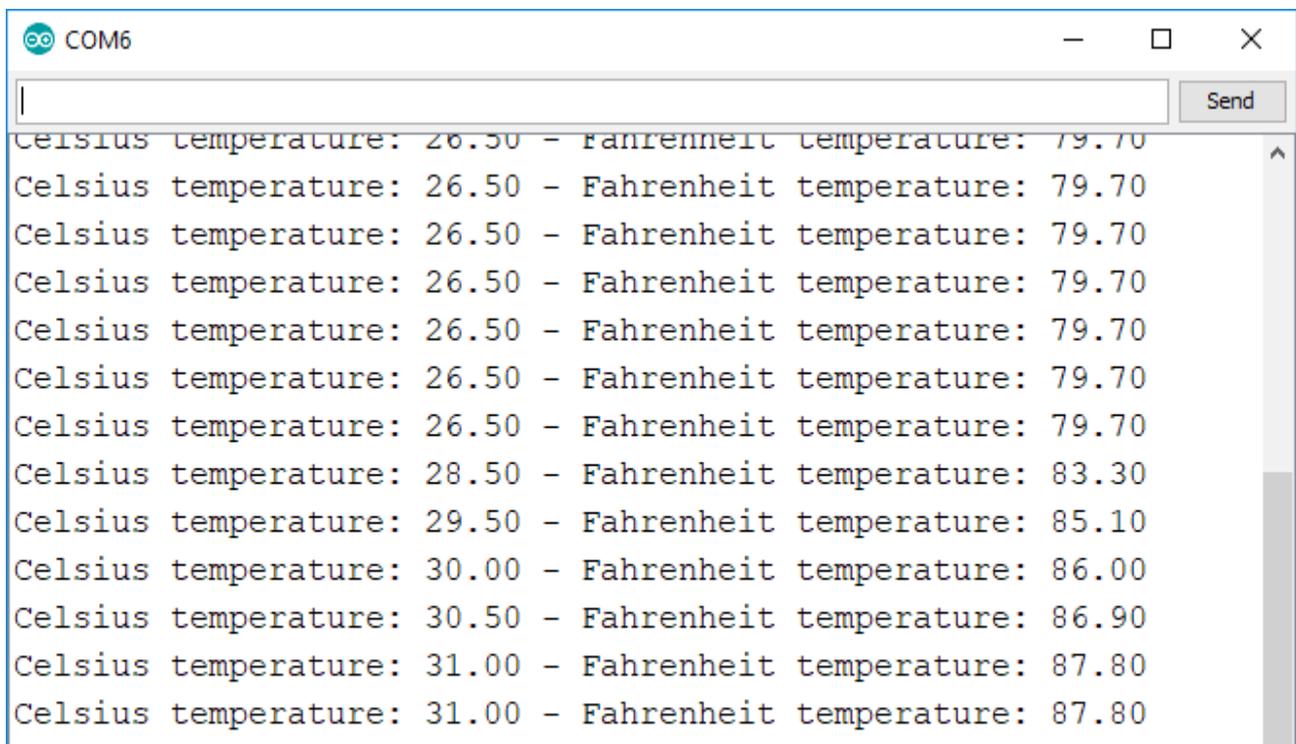
```
Serial.print(temperatureC);  
Serial.println("°C");  
Serial.print(temperatureF);  
Serial.println("°F");
```

New temperature readings are requested every 5 seconds.

```
Delay(5000);
```

Testing

After uploading the code, you should get your sensor readings displayed in the Serial Monitor

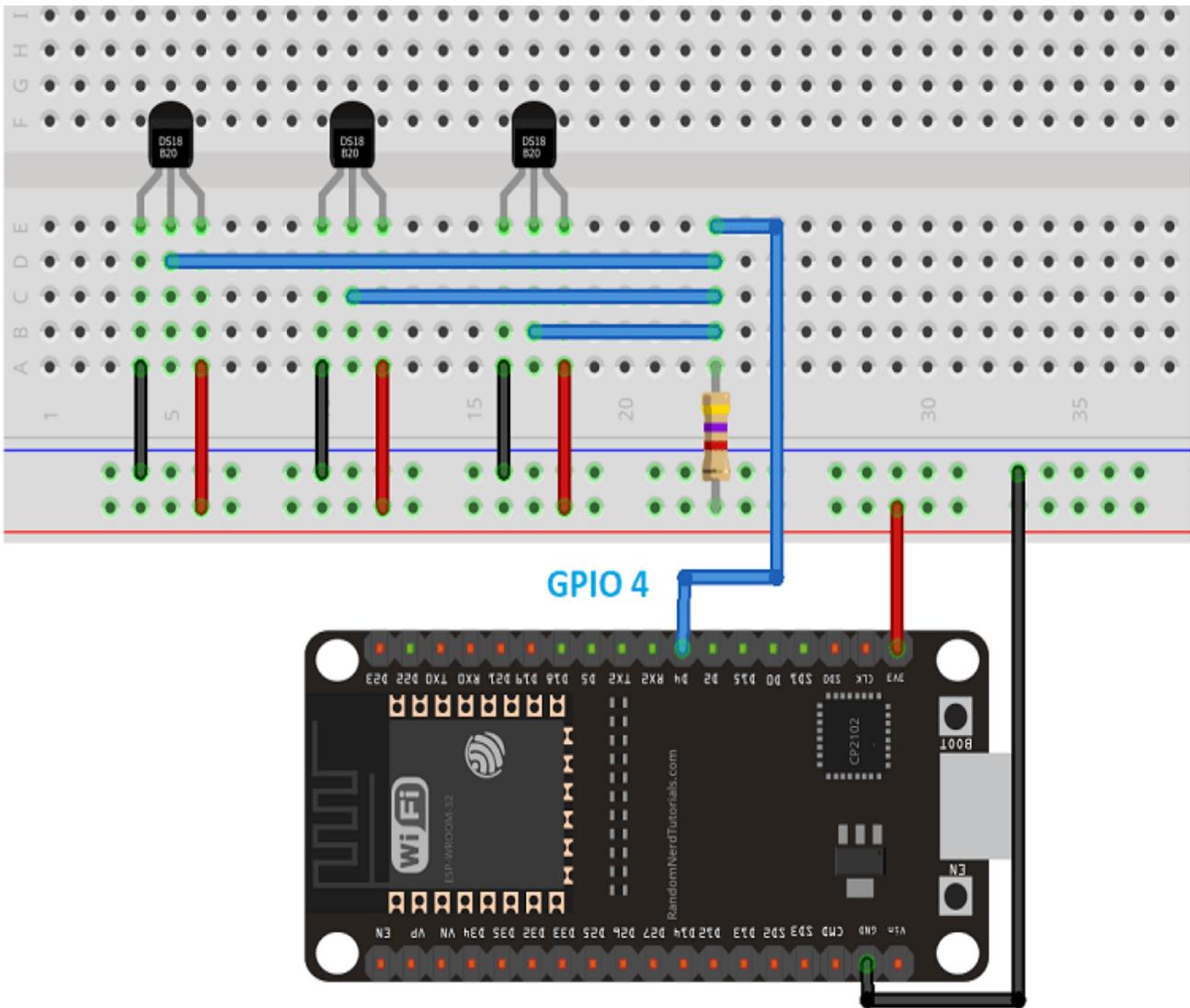


The screenshot shows the Serial Monitor window for COM6. The output displays temperature readings in Celsius and Fahrenheit, with a 'Send' button in the top right corner. The readings are as follows:

Celsius temperature	Fahrenheit temperature
26.50	79.70
26.50	79.70
26.50	79.70
26.50	79.70
26.50	79.70
26.50	79.70
26.50	79.70
28.50	83.30
29.50	85.10
30.00	86.00
30.50	86.90
31.00	87.80
31.00	87.80

Getting Temperature from Multiple DS18B20 Temperature Sensors

The DS18B20 temperature sensor communicates using one-wire protocol and each sensor has a unique 64-bit serial code, so you can read the temperature from multiple sensors using just one single GPIO. You just need to wire all data lines together as shown in the following schematic diagram:



Code

Then, upload the following code. It scans for all devices on GPIO 4 and prints the temperature for each one. This sketch is based on an example provided by the DallasTemperature library.

```
#include <OneWire.h>
#include <DallasTemperature.h>

// Data wire is plugged TO GPIO 4
#define ONE_WIRE_BUS 4

// Setup a oneWire instance to communicate with any OneWire devices
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

// Number of temperature devices found
int numberOfDevices;

// We'll use this variable to store a found device address
DeviceAddress tempDeviceAddress;

void setup()
{
  // start serial port
  Serial.begin(115200);

  // Start up the library
  sensors.begin();

  // Grab a count of devices on the wire
  numberOfDevices = sensors.getDeviceCount();

  // locate devices on the bus
  Serial.print("Locating devices...");
  Serial.print("Found ");
  Serial.print(numberOfDevices, DEC);
  Serial.println(" devices.");

  // Loop through each device, print out address
  for(int i=0;i<numberOfDevices; i++)
  {
    // Search the wire for address
    if(sensors.getAddress(tempDeviceAddress, i)){
      Serial.print("Found device ");
      Serial.print(i, DEC);
      Serial.print(" with address: ");
      printAddress(tempDeviceAddress);
      Serial.println();
    }
    else
    {
      Serial.print("Found ghost device at ");
      Serial.print(i, DEC);
      Serial.print(" but could not detect address. Check power and cabling");
    }
  }
}
```

```

void loop()
{
  sensors.requestTemperatures(); // Send the command to get temperatures

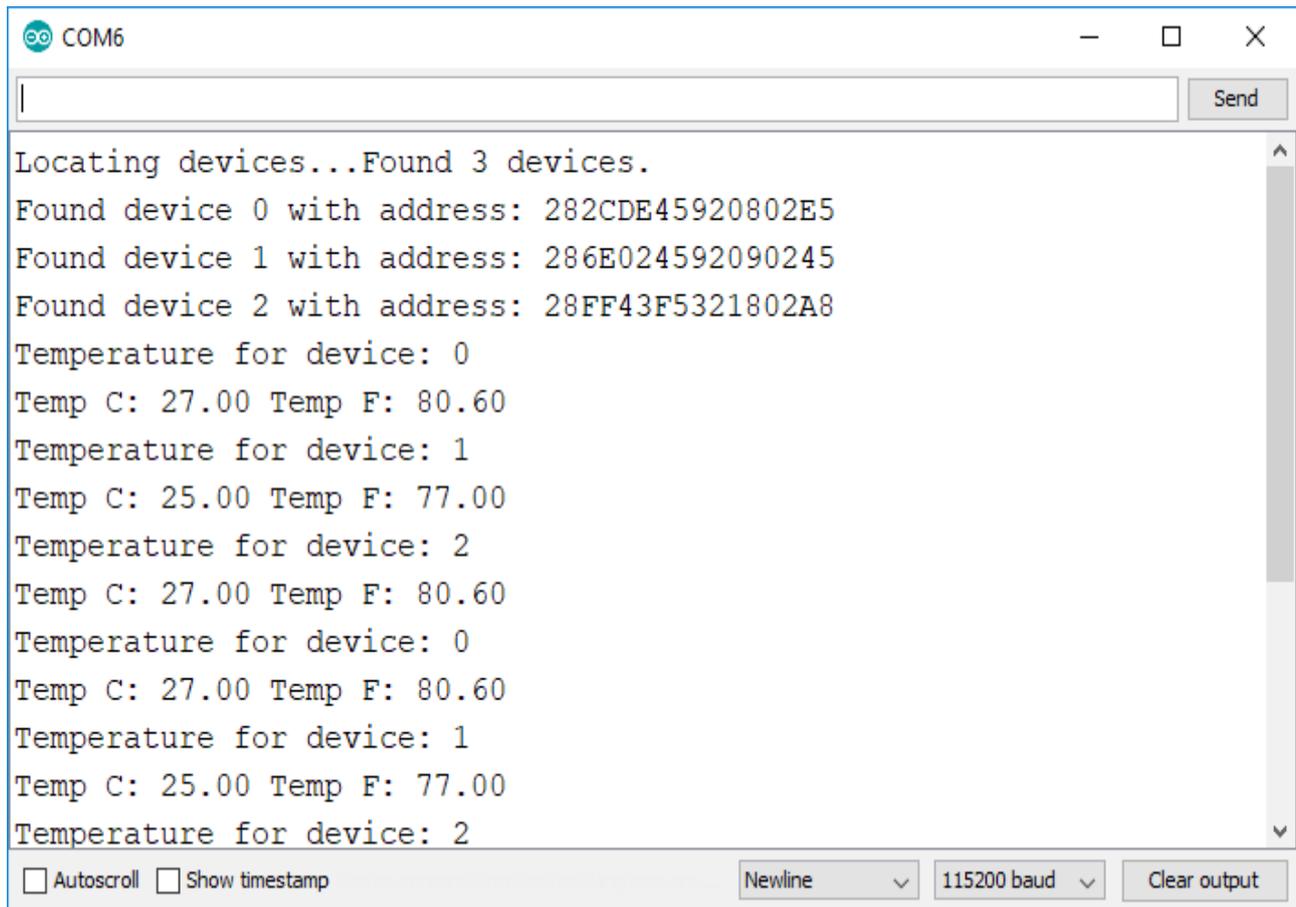
  // Loop through each device, print out temperature data
  for(int i=0;i<numberOfDevices; i++)
  {
    // Search the wire for address
    if(sensors.getAddress(tempDeviceAddress, i))
    {
      // Output the device ID
      Serial.print("Temperature for device: ");
      Serial.println(i,DEC);
      // Print the data
      float tempC = sensors.getTempC(tempDeviceAddress);
      Serial.print("Temp C: ");
      Serial.print(tempC);
      Serial.print(" Temp F: ");
      // Converts tempC to Fahrenheit
      Serial.println(DallasTemperature::toFahrenheit(tempC));
    }
  }
  delay(5000);
}

// function to print a device address
void printAddress(DeviceAddress deviceAddress)
{
  for (uint8_t i = 0; i < 8; i++)
  {
    if (deviceAddress[i] < 16)
    {
      Serial.print("0");
    }
    else
    {
      Serial.print(deviceAddress[i], HEX);
    }
  }
}

```

Testing

In this example, we're using three DS18B20 temperature sensors. This is what we get on the Arduino IDE Serial Monitor.



The screenshot shows the Arduino IDE Serial Monitor window for COM6. The output text is as follows:

```
Locating devices...Found 3 devices.  
Found device 0 with address: 282CDE45920802E5  
Found device 1 with address: 286E024592090245  
Found device 2 with address: 28FF43F5321802A8  
Temperature for device: 0  
Temp C: 27.00 Temp F: 80.60  
Temperature for device: 1  
Temp C: 25.00 Temp F: 77.00  
Temperature for device: 2  
Temp C: 27.00 Temp F: 80.60  
Temperature for device: 0  
Temp C: 27.00 Temp F: 80.60  
Temperature for device: 1  
Temp C: 25.00 Temp F: 77.00  
Temperature for device: 2
```

The window also features a 'Send' button at the top right, a scroll bar on the right side, and a control bar at the bottom with the following options: Autoscroll, Show timestamp, a 'Newline' dropdown menu, a '115200 baud' dropdown menu, and a 'Clear output' button.