# Arduino analog input and output

Difficult Level: 

## A. Introduction

The world is analog. And any inputs we can perceive are analog. For example, sounds are analog signals; they are continuous time and continuous value. Our ears listen to analog signals and we speak with analog signals. Images, pictures, and video are all analog at the source and our eyes are analog sensors.

## B. The analog in this Kit

An analog input pin can read a voltage level that ranges from the supply voltage of your board 3.3 v (3.3V) down to 0 v (GND). The value returned to your program from reading the pin is not the actual voltage value but a number that is between 0 and 1023. This is because the microcontroller on the board is a digital system and must convert the analog signal at an input pin to a digital number. The value representing the input voltage that is returned is relative to the actual voltage level present at the pin.

Reading an analog input pin is quite simple. There’s just one block to does it which returns a number that represents the input voltage level. The raindrop sensor is an analog quantity in the kit. Now let’s read the raindrop sensor.

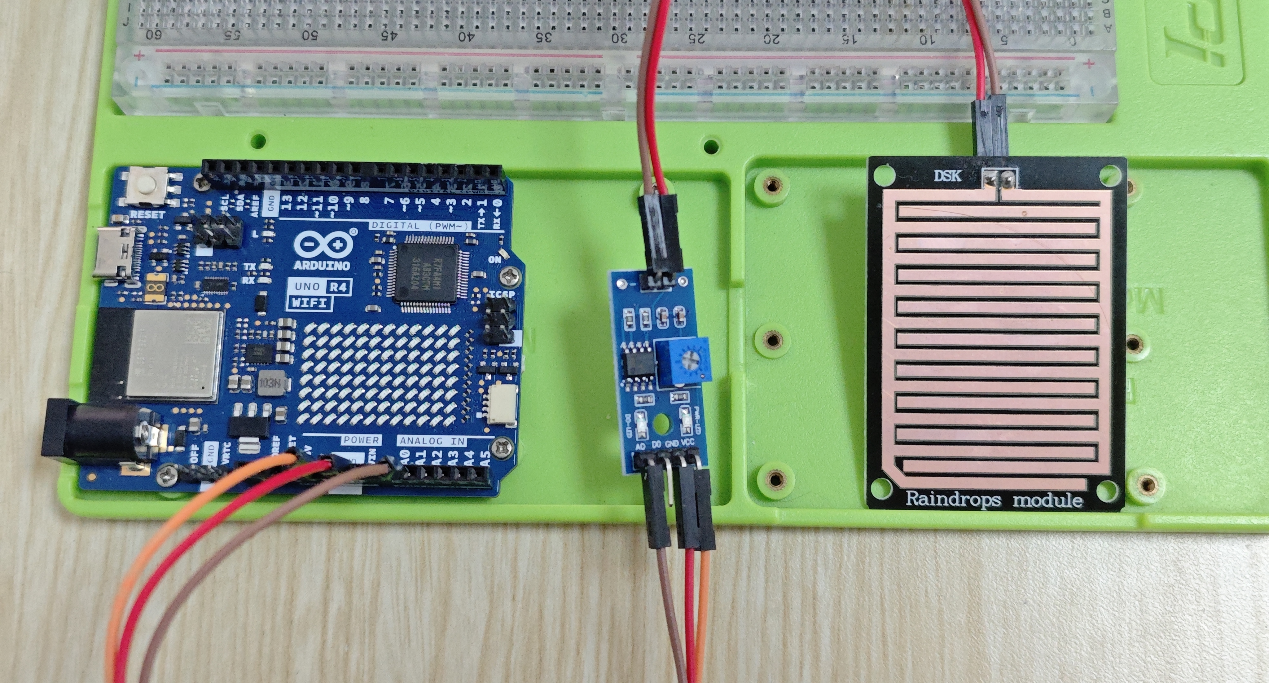
### What you need?

* Arduino IDE software
* 1 x Arduino UNO R4 Wi-Fi
* 1 x USB 2.0 cable Type C
* 1 x Raindrop Sensor Module
* 3 x Female-Male Jumper wires

### How to wiring circuit diagram?

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | Raindrop Sensor Module |
| 3.3V | VCC |
| GND | GND |
| NC | DO |
| A0 | AO |

You can look at the circuit connection diagram below to connect our own devices.



Then connect the computer with the Arduino UNO R4 WIFI, and open the Arduino IDE. Then copy the code and upload it. It’s easy to know we need only read from the A0 pin.

**#define AO\_PIN A0 // Arduino's pin connected to AO pin of the rain sensor**

**void setup() {**

**// initialize serial communication**

**Serial.begin(9600);**

**}**

**void loop() {**

**//read analog from A0 pin**

**int rainValue = analogRead(AO\_PIN);**

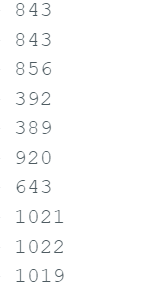
**Serial.println(rainValue); // print out the analog value**

**delay(1000); // pause for 1 sec to avoid reading sensors frequently to prolong the sensor lifetime**

**}**

After upload finished we can get our answer from the serial monitor. (By put some water drops on the sensor and dry it slowly, its value gradually increases)

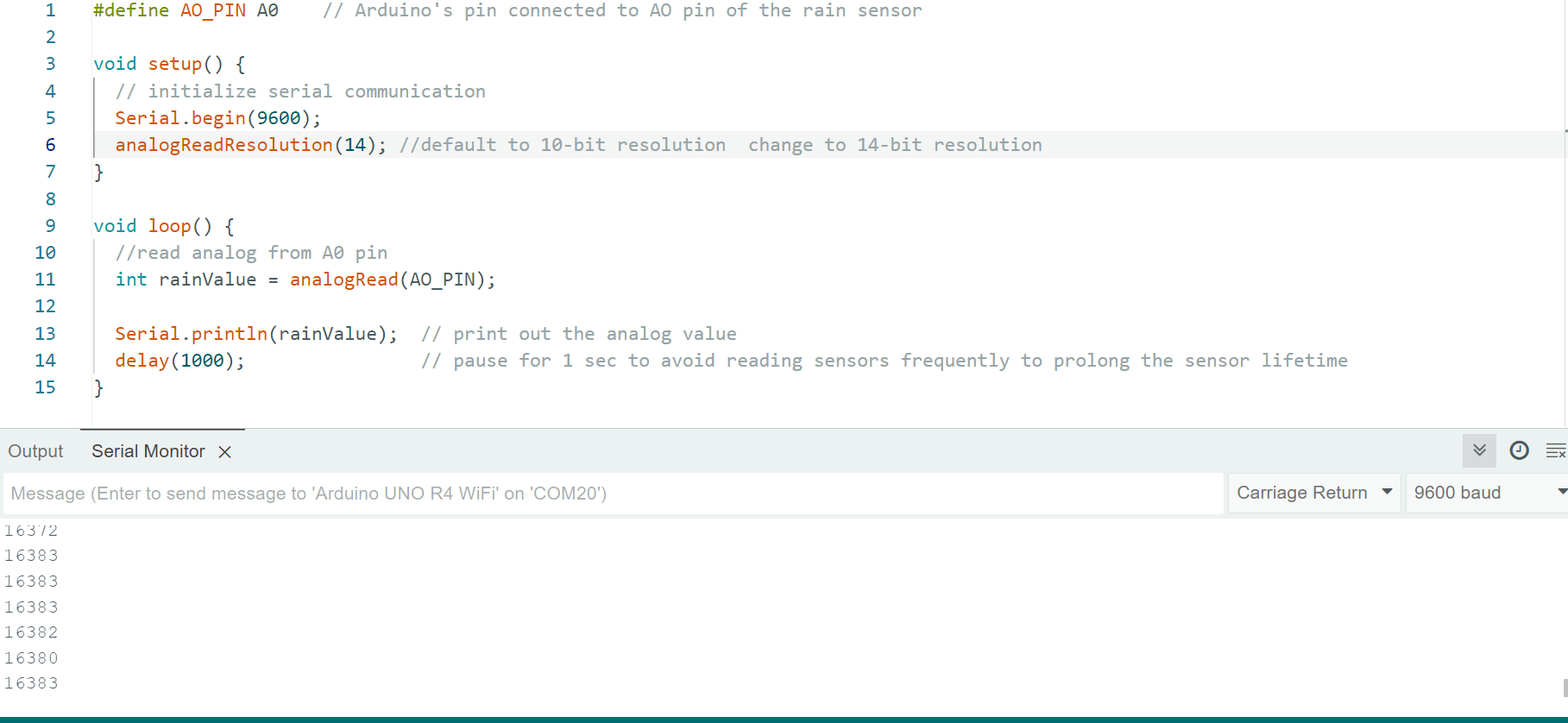




An analog-to-digital converter (ADC) transforms an analog signal to a digital one. In the previous Arduino Uno board, the Arduino Uno R4 Wi-Fi updates the standard resolution. You can use **analogReadResolution()** to update the standard resolution to 12-bit or 14-bit resolutions. The standard resolution on Arduino boards is typically set to 10-bit (0-1023).

This means that it will map input voltages between 0 and the operating voltage (5V or 3.3V) into integer values between 0 and 1023 (0-4095 for 12 bits, 0-16383 for 14 bits). At 10-bit resolution, for example, this yields a resolution between readings of: 5 volts / 1024 units or, 0.0049 volts (4.9 mV) per unit.

You can modify your code to read the data more accurately based on the information below.



## C. Moisture Sensor

The work principle of a moisture sensor's analog output is based on the variation of conductivity or resistance in the sensor's probes as they come into contact with the moisture content in the soil or other material being measured. Here's how it generally works:

* **Sensor Construction:** A moisture sensor typically consists of two or more metal probes or electrodes embedded in a material that can absorb moisture, such as soil. These probes are connected to the sensor circuitry.
* **Conductivity Variation:** When the sensor probes are inserted into the soil, they come into contact with the moisture content. Moisture in the soil enhances its conductivity, allowing electric current to flow more easily between the probes.
* **Analog Output:** The sensor measures the conductivity between its probes and generates an analog voltage or current signal proportional to the moisture content. Higher moisture levels result in higher conductivity and thus a higher analog output signal, while lower moisture levels result in lower conductivity and a lower analog output signal.
* **Calibration:** To convert the analog output signal to a meaningful moisture level, the sensor may need to be calibrated. This calibration process typically involves correlating the analog output signal with known moisture levels in the material being measured. This calibration can be done empirically by testing the sensor in different moisture conditions and recording the corresponding analog output values.
* **Signal Processing:** Once calibrated, the analog output signal can be processed by a microcontroller or other control system to determine the moisture level. This processing may involve scaling the analog signal, applying calibration factors, and possibly filtering out noise or other unwanted signals.
* **Display or Action:** Finally, the moisture level information derived from the analog output signal can be displayed on a screen, used to trigger actions such as activating irrigation systems, or transmitted wirelessly to other devices for further processing or monitoring.

Overall, the analog output of a moisture sensor provides a convenient and relatively simple way to measure and monitor the moisture content in soil or other materials, making it useful for applications such as agriculture, gardening, and environmental monitoring.

### Demo code

Open a new sketch on Arduino IDE, and typing following code, or write your own.

and use the same way to upload the code to Arduino board.

**#define MOISTURE\_SENSOR A1   // Arduino's pin connected to AO pin of the moisture sensor**

**void setup() {**

**// initialize serial communication**

**Serial.begin(9600);**

**analogReadResolution(14); //default to 10-bit resolution**

**}**

**void loop() {**

**//read analog from A1 pin**

**int moisture\_value = analogRead(MOISTURE\_SENSOR);**

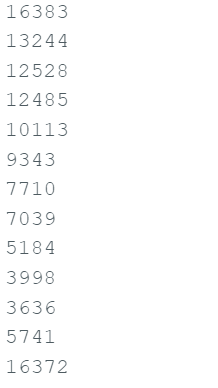
**Serial.println(moisture\_value);  //print out value**

**delay(1000);   // pause for 1 sec to avoid reading sensors**

**}**

To test the moisture sensor, first ensure that the code has been uploaded to the microcontroller and the sensor is correctly wired according to the provided instructions. Once everything is set up, place the sensor probes into the soil or other material you wish to test. Then, observe the readings either through the serial monitor or any output method specified in the code. If the sensor is functioning properly, you should see readings corresponding to the moisture level of the soil. Additionally, you can simulate changes in moisture levels by adding or removing water from the soil and observing how the sensor readings respond. This will help verify that the code accurately reflects changes in moisture content.

When you open serial monitor, you will see following figure, put moisture sensor into the soil and watering the soil, observe the data changes.



Congratulations! By successfully testing the demo code, you've taken the first step towards exploring the exciting world of sensor technology. Now, I encourage you to take your learning journey a step further by diving into the code and experimenting with it. Building your own code based on the demo code is a fantastic way to deepen your understanding and creativity in electronics and programming. Don't hesitate to modify parameters, add new features, or integrate additional sensors to tailor the code to your specific needs or project ideas. Embrace the opportunity to innovate and create something truly unique. Happy coding!