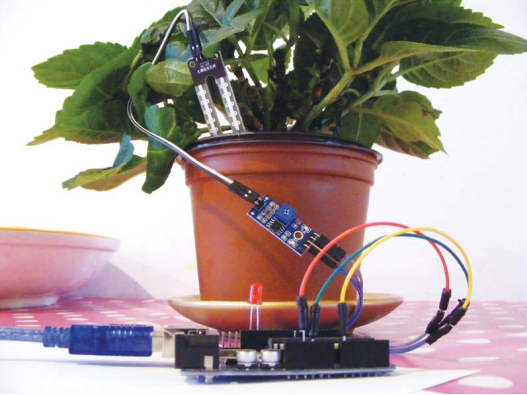
**Reg No:21RP01304**

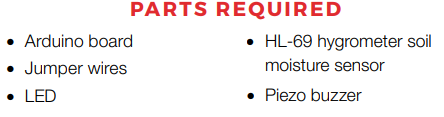
**Project Plant Monitor**

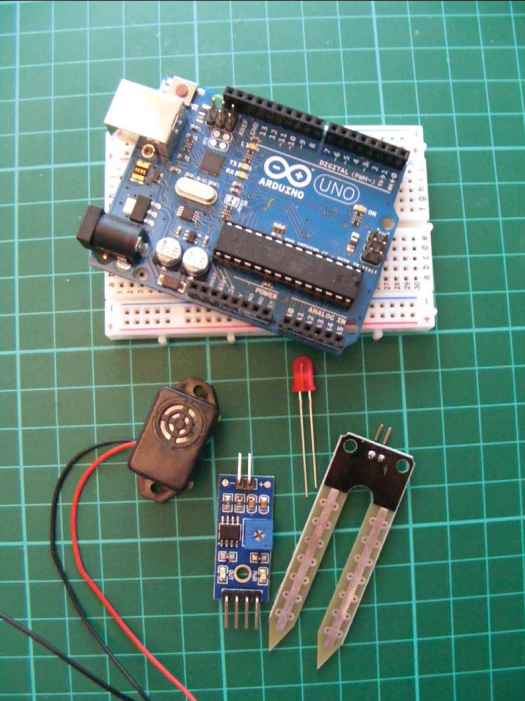
**In this project I’ll introduce a new type of analog sensor that detects moisture levels.**

**You’ll set up a light and sound alarm system to tell you when your plant**

**needs watering**.







**How It Works**

You’ll use an HL-69 moisture sensor, readily available online for a few

dollars or from some of the retailers . The prongs

of the sensor detect the moisture level in the surrounding soil by pass\_ing current through the soil and measuring the resistance. Damp soil

conducts electricity easily, so it provides lower resistance, while dry

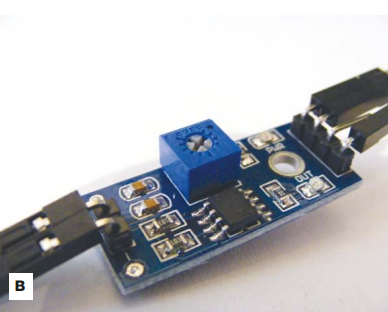
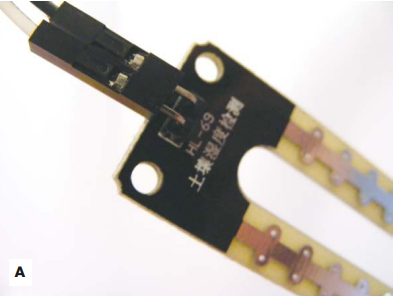
soil conducts poorly and has a higher resistance.

The sensor consists of two parts:

actual prong sensor (a) and the controller (b). The two pins on the

sensor need to connect to the two separate pins on the controller

(connecting wires are usually supplied). The other side of the con\_troller has four pins, three of which connect to arduino.



The four pins are, from left to right, AO (analog out), DO (digital

out), GND, and VCC (see Figure above). You can read the values from

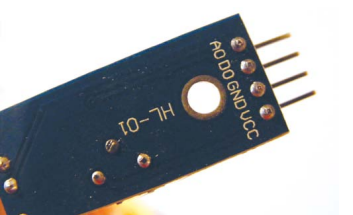
the controller through the IDE when it’s connected to your computer.

The HL-69 moisture sensor

prong (a) and controller (b)

This project doesn’t use a breadboard, so the connections are all

made directly to the Arduino.



Lower readings indicate that more moisture is being detected, and

higher readings indicate dryness. If your reading is above 900, your

plant is seriously thirsty. If your plant gets too thirsty, the LED will light

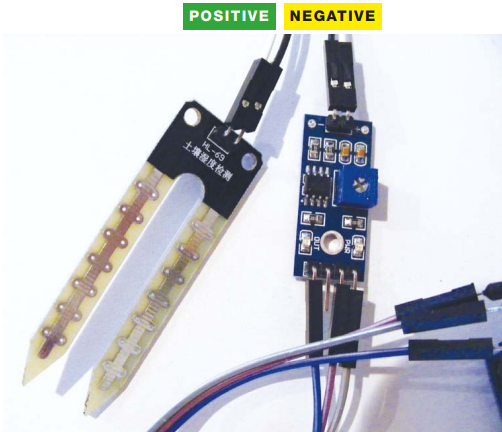
and the piezo buzzer will sound.

**The Build**

1. Connect the sensor’s two pins to the + and – pins on the

controller using the provided connecting wires, as shown in

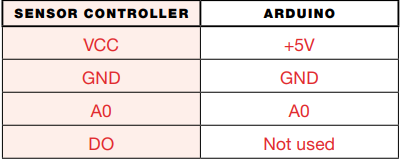
Figure below:



2. Connect the three prongs from the controller to +5V, GND, and

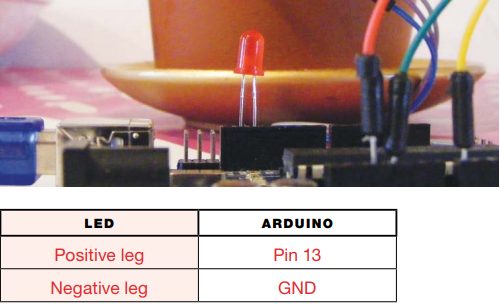
Arduino A0 directly on the Arduino, as shown in the following

table. The DO pin is not used.



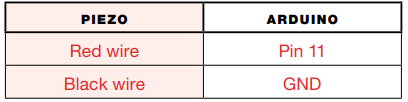
3. Connect an LED directly to the Arduino with the shorter, nega\_tive leg in GND and the longer, positive leg in Arduino pin 13, as

shown in Figure below:



4. Connect the piezo buzzer’s black wire to GND and its red wire to

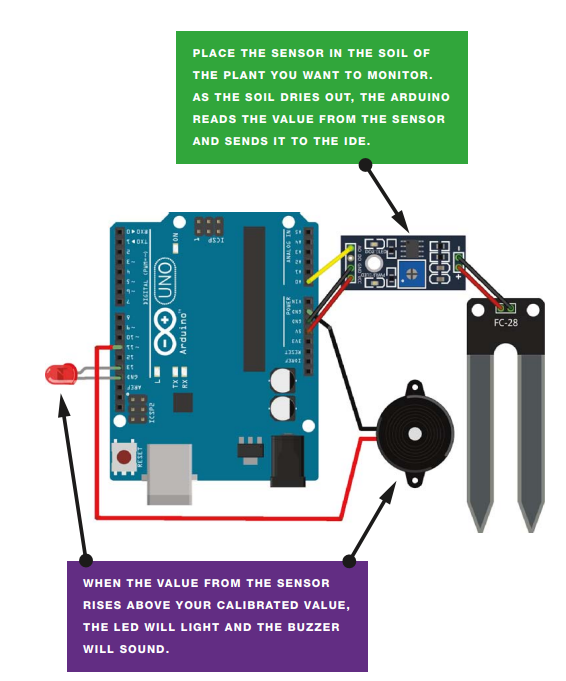
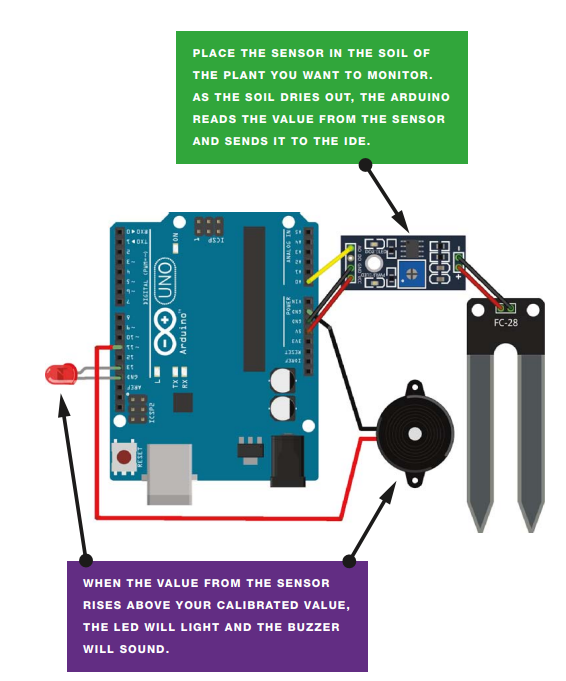
Arduino pin 11



5. Check that your setup matches that of Figure , and then

upload the code in “The Sketch”

**CIRCUIT DIAGRAM**

****

6. Connect the Arduino to your computer using the USB cable.

Open the Serial Monitor in your IDE to see the values from the

sensor—this will also help you to calibrate your plant monitor.

The IDE will display the value of the sensor’s reading. My value was 1000 with the sensor dry and not inserted in the soil, so I know this is the highest, and driest, value. To calibrate this value, turn the potentiometer on the controller clockwise to increase the resistance and counterclockwise to decrease it (see Figure Above)

When the sensor is inserted into moist soil, the value will

drop to about 400. As the soil dries out, the sensor value rises;

when it reaches 900, the LED will light and the buzzer will sound.



**The Sketch/Arduino code**

The sketch first defines Arduino pin A0 so that it reads the moisture

sensor value. It then defines Arduino pin 11 as output for the buzzer,

and pin 13 as output for the LED. Use the Serial.Println() func\_tion to send the reading from the sensor to the IDE, in order to see

the value on the screen.

Change the value in the line

if(analogRead(0) > 900){

depending on the reading from the sensor when it is dry (here it’s

900). When the soil is moist, this value will be below 900, so the LED

and buzzer will remain off. When the value rises above 900, it means

the soil is drying out, and the buzzer and LED will alert you to water

your plant.

const int moistureAO = 0;

int AO = 0; // Pin connected to A0 on the controller

int tmp = 0; // Value of the analog pin

int buzzPin = 11; // Pin connected to the piezo buzzer

int LED = 13; // Pin connected to the LED

void setup () {

Serial.begin(9600); // Send Arduino reading to IDE

Serial.println("Soil moisture sensor");

pinMode(moistureAO, INPUT);

pinMode(buzzPin, OUTPUT); // Set pin as output

pinMode(LED, OUTPUT); // Set pin as output

}

**Benefits of a plant efficiency monitoring system in manufacturing**

* Improved productivity. The system can help identify and address inefficiencies in the production process, resulting in increased productivity and output.
* Reduced downtime.
* Better decision-making.
* Improved quality control.
* Reduced costs.

.