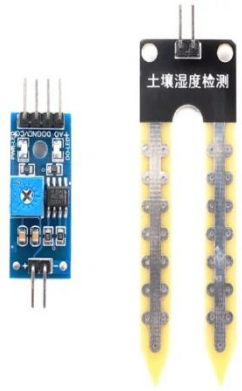
YL-69 Soil Moisture Sensor Module (Analog Mode + Digital Mode)

Used to measure soil moisture content, useful for plant watering automation.

Pin	***Description***
VCC	*Power Supply (3.3V–5V)*
GND	*Ground*
A0	*Analog Output (connect to ESP32 analog pin, e.g., GPIO 34)*
D0	*Digital Output (connect to digital pin if needed; optional)*



Sensor Probes connect to the 2-pin screw terminal on the board.

Formulas to be Used (Analog Mode):

- *The voltage from A0 (due to resistance in soil):*

$$\mathbf{V_{OUT} = V_{CC} \times \frac{R_{soil}}{R_{soil} + R_{POT}}}$$

- *This voltage becomes input to ESP32 12-bit ADC:*

$$\mathbf{ADC\ (Value) = 4095 \times \frac{V_{in}}{V_{CC}}}$$

- *More moisture → lower resistance → lower ADC reading*
  Dry soil → higher resistance → higher ADC reading

How It Works:

- *The sensor has two conductive probes.*
- *When inserted into soil:*
  - *Moisture allows current to flow → lower resistance*
  - *Dryness reduces flow → higher resistance*

- *The voltage divider converts this into analog voltage on A0.*
- *D0 gives HIGH/LOW based on threshold (adjustable via onboard potentiometer).*

Points to Remember:

- *Use analogRead() on ESP32 to get 0–4095 from A0.*
- *For digital mode, connect D0 to ESP32 digital pin and read via digitalRead().*
- *Adjust threshold using onboard potentiometer for D0 logic level.*
- *Works with 3.3V or 5V; ESP32 safe.*

Use Cases:

- *Automatic plant watering systems*
- *Smart garden or greenhouse monitoring*
- *Soil condition logger for data analytics*
- *Alert or buzzer when plants need water*
- *IoT-based agriculture solutions*

Code:

```
#define YL69_d 4
#define YL69_a 32
#define LED 2
void setup() {
  Serial.begin(115200);
  pinMode(LED, OUTPUT);
  pinMode(YL69_a, INPUT);
  pinMode(YL69_d, INPUT);
}
void loop() {
  int digitalValue = digitalRead(YL69_d);
  int rawAnalogValue = analogRead(YL69_a);
  Serial.print("digitalValue: ");
  Serial.println(digitalValue);
  Serial.print("analogValue: ");
  Serial.println(rawAnalogValue);
  float MoistureLevel = ((4095-rawAnalogValue)/4095.0)*100.0;
  Serial.print(MoistureLevel);
  Serial.println("%");
  delay(500);
}
```

RC522 RFID Reader Module (MFRC522 – SPI Interface)

The RC522 is an RFID reader/writer module for 13.56 MHz passive RFID tags and cards using the MFRC522 chip. It uses SPI communication and is widely used in access control, attendance and identification systems.

Pinout of RC522 Module (SPI Mode)

Pin	ESP32 Pins
VCC	3.3V
GND	GND
RST	GPIO 27
IRQ	Leave unconnected
MISO	GPIO 19
MOSI	GPIO 23
SCK	GPIO 18
SDA (SS)	GPIO 5

How It Works:

- The RC522 uses SPI to communicate with the ESP32.
- When an RFID tag/card is placed near the antenna:
  - The reader detects the tag using electromagnetic induction.
  - It reads the UID (Unique ID) and can access data stored in MIFARE 1K tags.
- The UID is usually 4 bytes/32-bits long and sent to the ESP32.
- Useful for authentication, tracking, or identification.

Points to Remember:

- Use MFRC522 library (by Github user miguelbalboa) for ESP32.
- Initialize using SPI.begin() and mfrc522.PCD_Init();
- Use PICC_IsNewCardPresent() and PICC_ReadCardSerial() to detect and read UID.
- Tags/cards must be within 2–5 cm of the antenna for detection.

UID Reading Formula:

- If UID is 4 bytes:

```
for (byte i = 0; i < mfrc522.uid.size; i++) {
    uid += String(mfrc522.uid.uidByte[i], HEX);
}
```

- This gives a hexadecimal UID string (e.g., a3b4c159)

Example Use Cases:

- Attendance systems for school or office
- Smart door locks or entry gates
- Library management using RFID cards
- Cashless payment with stored-value cards
- Pet or object tracking with RFID tags

CODE:

```
#include<SPI.h>
#include<MFRC522.h>
#define SS 5
#define RST 27
MFRC522 rfid(SS, RST);
void setup() {
  Serial.begin(115200);
  SPI.begin();
  rfid.PCD_Init();
}
void loop() {
  // Serial.println("Waiting for RFID Card!");
  if(rfid.PICC_IsNewCardPresent() && rfid.PICC_ReadCardSerial()){
    Serial.print("Card Detected! UID: ");
    for(byte i = 0; i<rfid.uid.size; i++){
      Serial.print(rfid.uid.uidByte[i] < 10 ? "0" : " ");
//10(HEXA) is 16(Decimal) i.e. as upto A there is single letter/number
      Serial.print(rfid.uid.uidByte[i], HEX);
    }
    Serial.println();
    rfid.PICC_HaltA();
  }
  delay(1000);
}
```