센서 3

릴레이/조이스틱/배터리/
토양수분센서/빗 방울 센서/
GPS센서/블루투스/컬러 센서


Keyes_SR1y

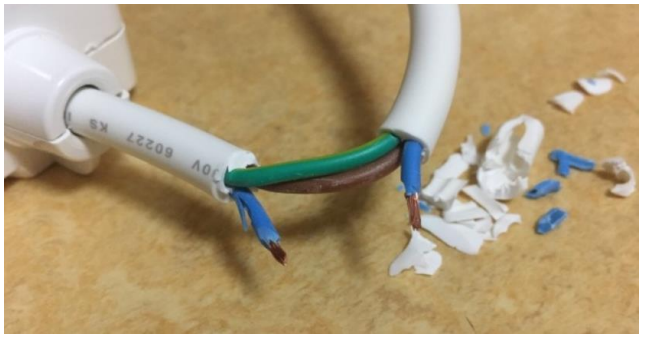
AA Battery
AA Battery
Keyes_SR1y
10A 250VAC 10A 125VAC
10A 30VDC 10A 28VDC
SRD-05VDC-SL-C
Arduino
UNO

```
int relay = 9;
void setup ()
{
  pinMode (relay, OUTPUT);
}
void loop ()
{
  digitalWrite (relay, HIGH);
  delay (1000);
  digitalWrite (relay, LOW);
  delay (1000);
}
```

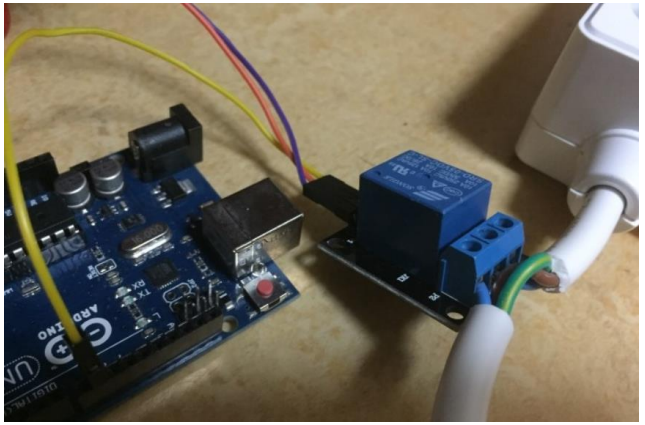
릴레이 연결의 기초 예제!

- 위 이미지를 기준으로 전원연결측 단자(왼쪽)과 아두이노 연결 단자(오른쪽)으로 나눔
- 아두이노 연결단자는 모듈에 적혀있는대로 Vcc는 5V로 연결하고 Gnd 는 Gnd로, S는 신호를 받는 단자로 소스코드와 같은 핀에 연결
- 외부전원 연결단자는 가운데 붉은 단자는 공통 단자 이며 ON일때 연결하려면 위쪽 단자에 OFF일때 연결하려면 아래쪽단자에 연결하면 됨
- 다시말해서 위쪽단자와 함께 연결할 경우 아두이노에서 신호를 줄때(HIGH) LED가 켜지게 됨

- 릴레이는 낮은전압으로 높은 전압의 기기를 컨트롤할 때 사용하는 전자부품
- 예를 들어서, 아두이노의 5V전압으로 220V의 헤어드라이기나 전등, 선풍기 등 전제제품을 제어 할 수 있음

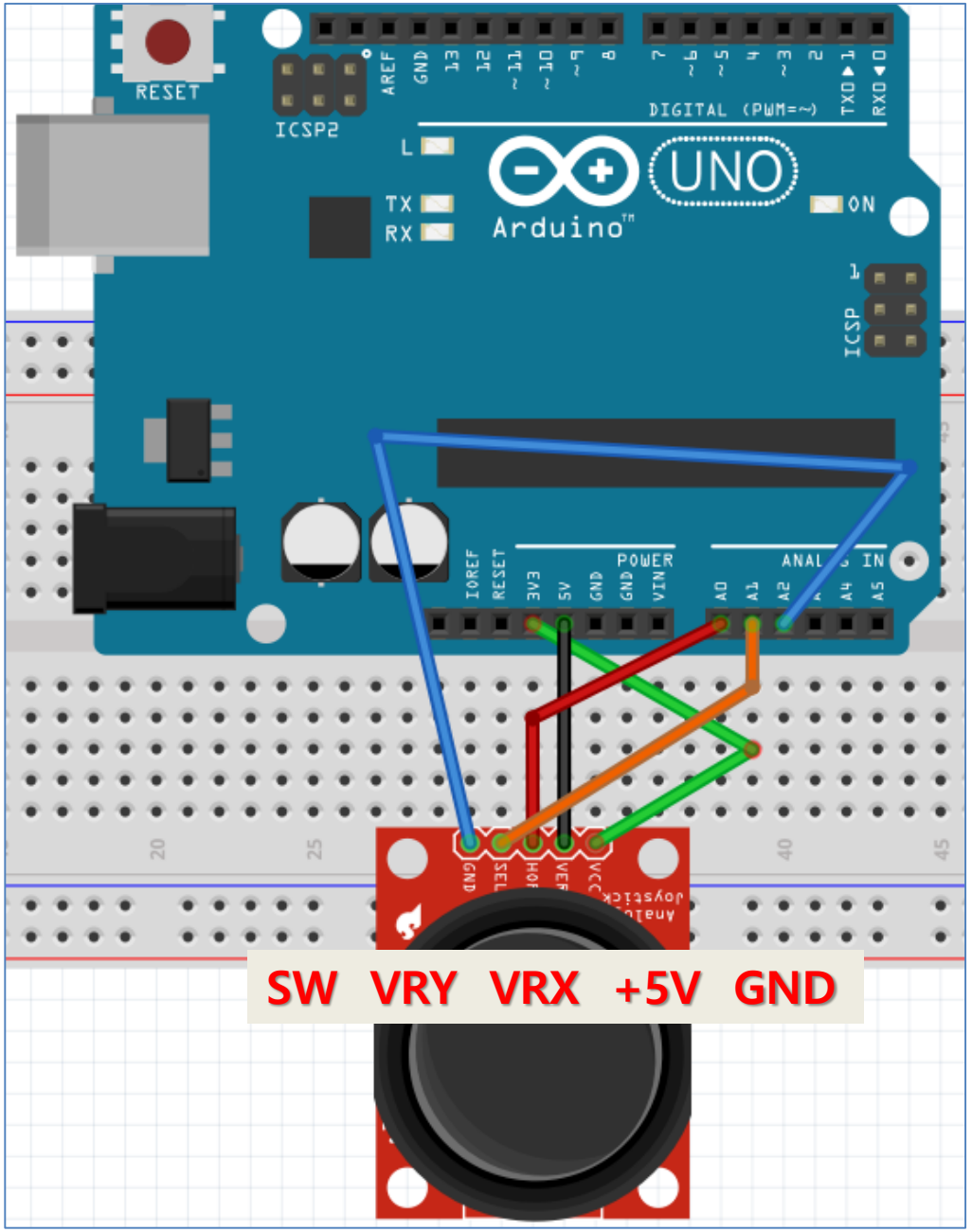


아두이노 1채널 릴레이 모듈 5V용 E4-P26



https://blog.naver.com/3demp/221057404554

조이스틱


RESET
ICSP2
DIGITAL (PWM=~)
UNO
Arduino
ON
ICSP
POWER
SW VRY VRX +5V GND

JovStic

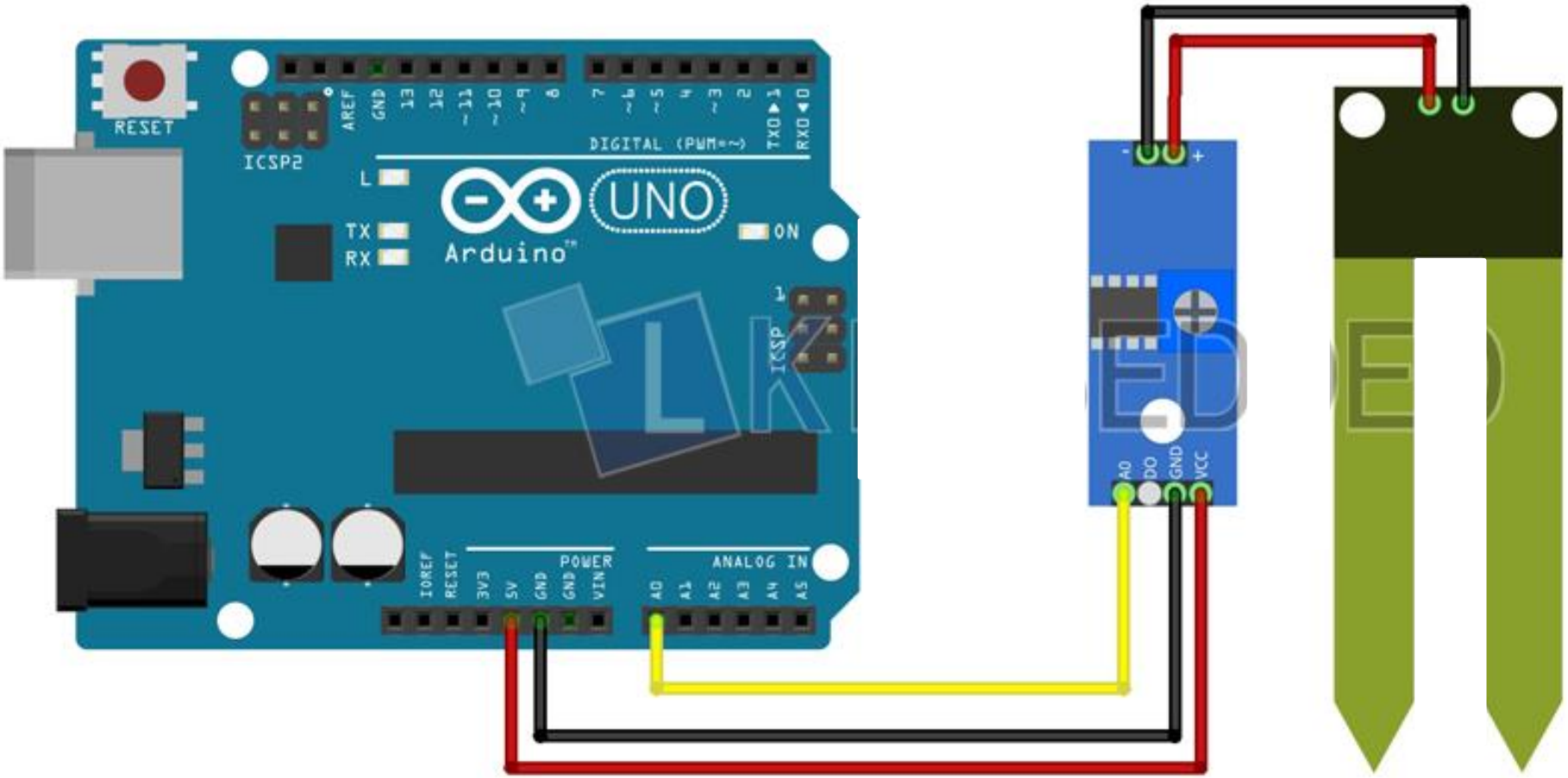
```
int JovStic_X = A0;
int JoyStic_Y = A1;
int pin_SW = A2;

void setup() {
  pinMode(pin_SW, INPUT_PULLUP); //스위치가 Pullup상태에서 해제될때 인식
  Serial.begin(9600);
}

void loop() {
  int x, y, z;
  x=analogRead(JovStic_X);
  y=analogRead(JoyStic_Y);

  Serial.print(x,DEC);
  Serial.print(",");
  Serial.print(y,DEC);
  Serial.print(",");

  if(digitalRead( pin_SW ))
  {
    Serial.println("OFF");
  }else {
    Serial.println("ON");
  }
  delay(100);
}
```

COM3 (Arduino/Genuino Uno)

```
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,488,OFF
517,489,OFF
517,488,OFF
```

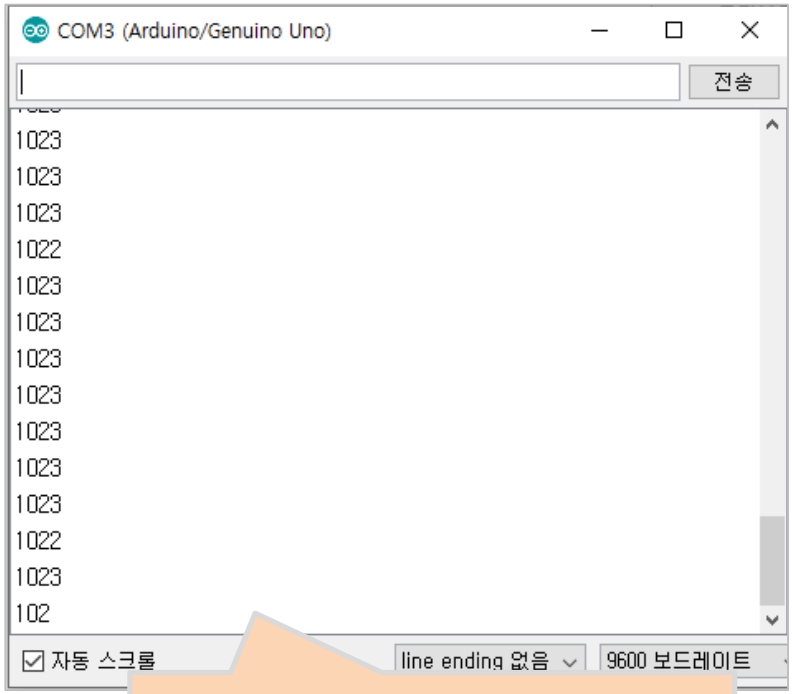
☑ 자동 스크롤

토양 수분 센서


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino
ON
ICSP
POWER
IOREF
RESET
3V3
5V
GND
GND
VIN
ANALOG IN
A0
A1
A2
A3
A4
A5
A0
DO
GND
VCC

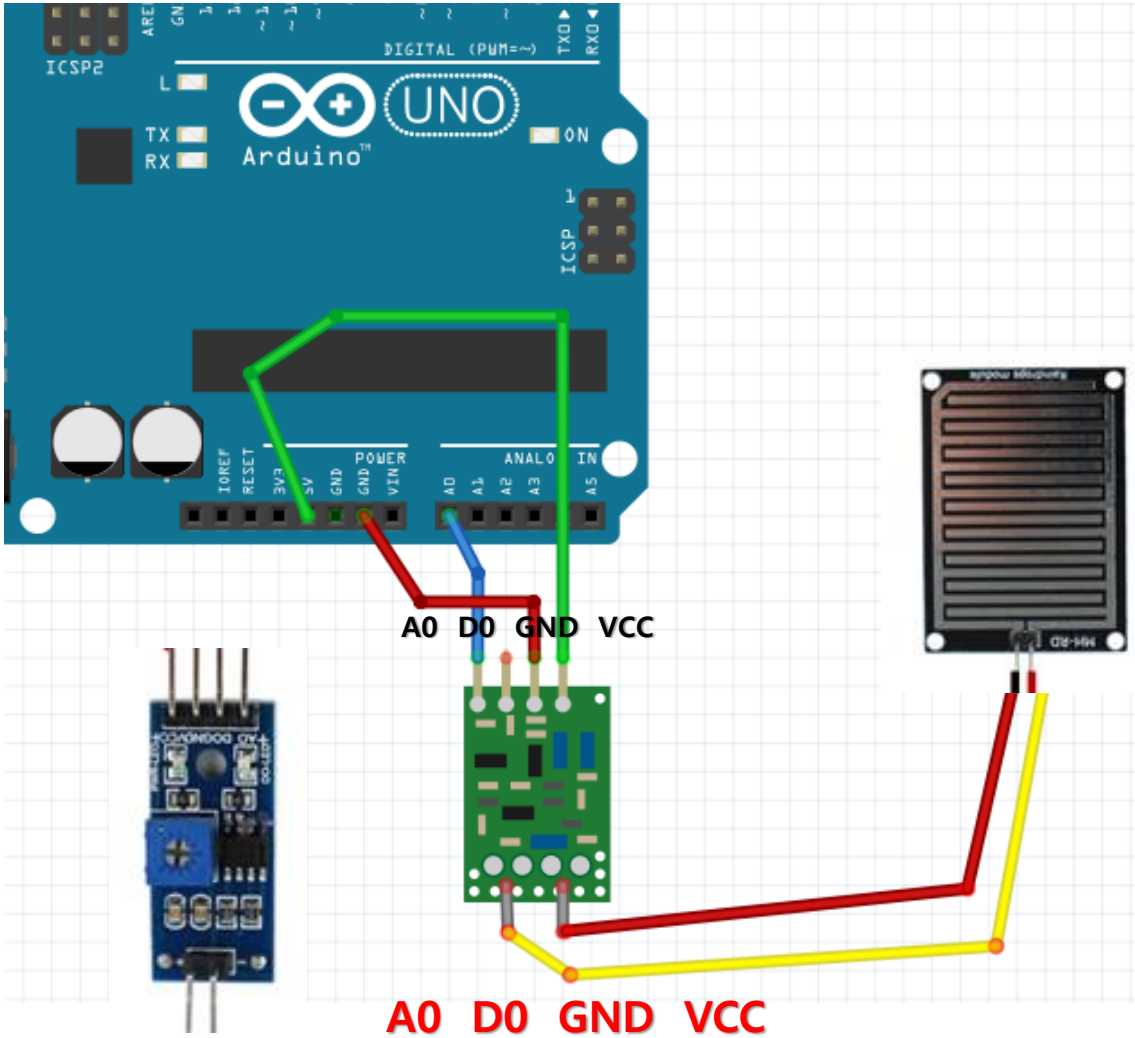
토양 수분 센서

```
unsigned int val;

void setup() {
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop() {
  val = analogRead(A0);
  Serial.println(val);
  delay(100);
}
  /*0 ~ 300 : 높은 습도 상태
    300 ~ 700 : 보통 습도 상태
    700 ~ 950 : 건조 상태*/
```



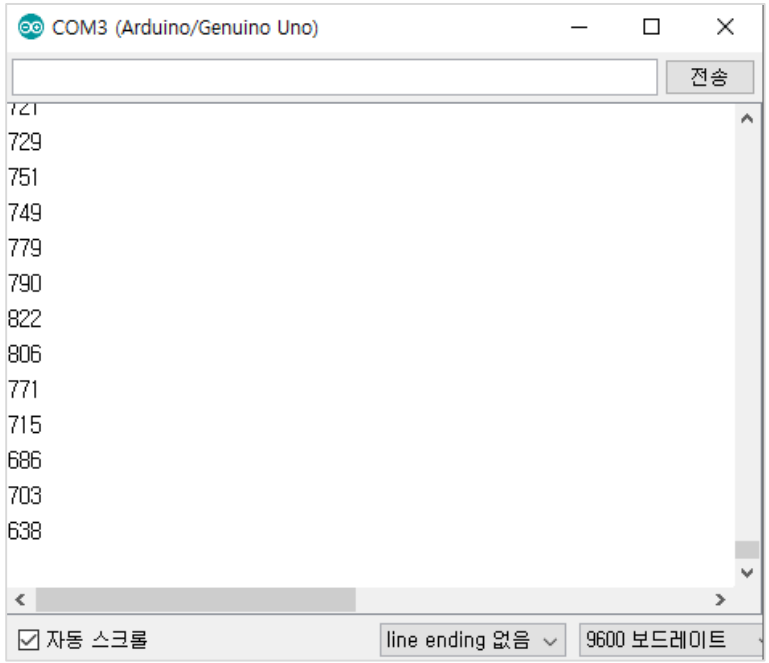
0~300 : 높은 습도
300~700 : 보통 습도
700 ~ 1024 : 건조

빗방울 센서

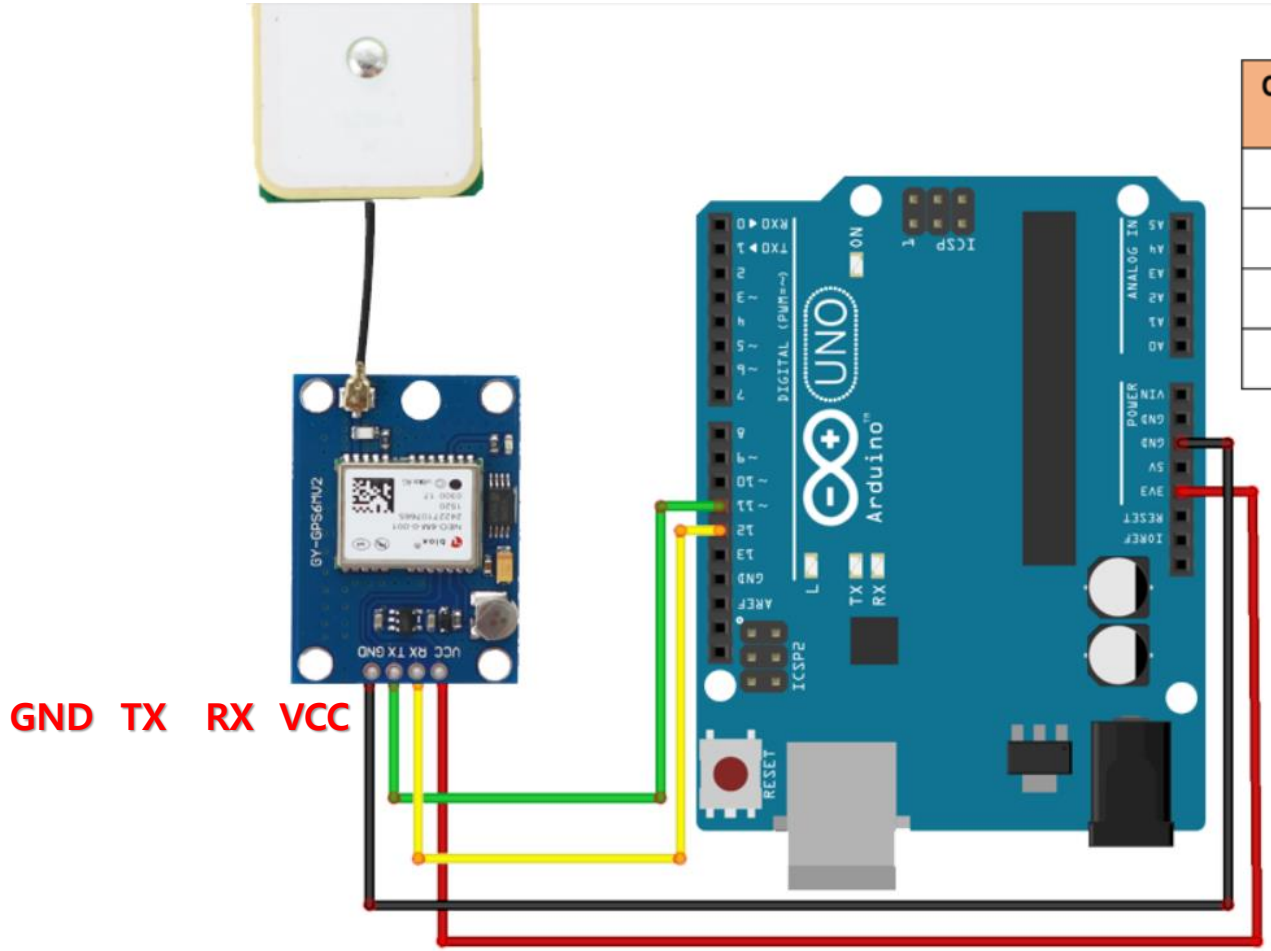

ICSP2
DIGITAL (PWM=~)
UNO
Arduino™
TX
RX
ON
ICSP
POWER
ANALOG IN
A0 D0 GND VCC
A0 D0 GND VCC

빗방울 센서

```
int Raindrops_pin = A0;

void setup() {
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop() {
  int val = analogRead(Raindrops_pin);
  Serial.println(val);
  delay(100);
}
```



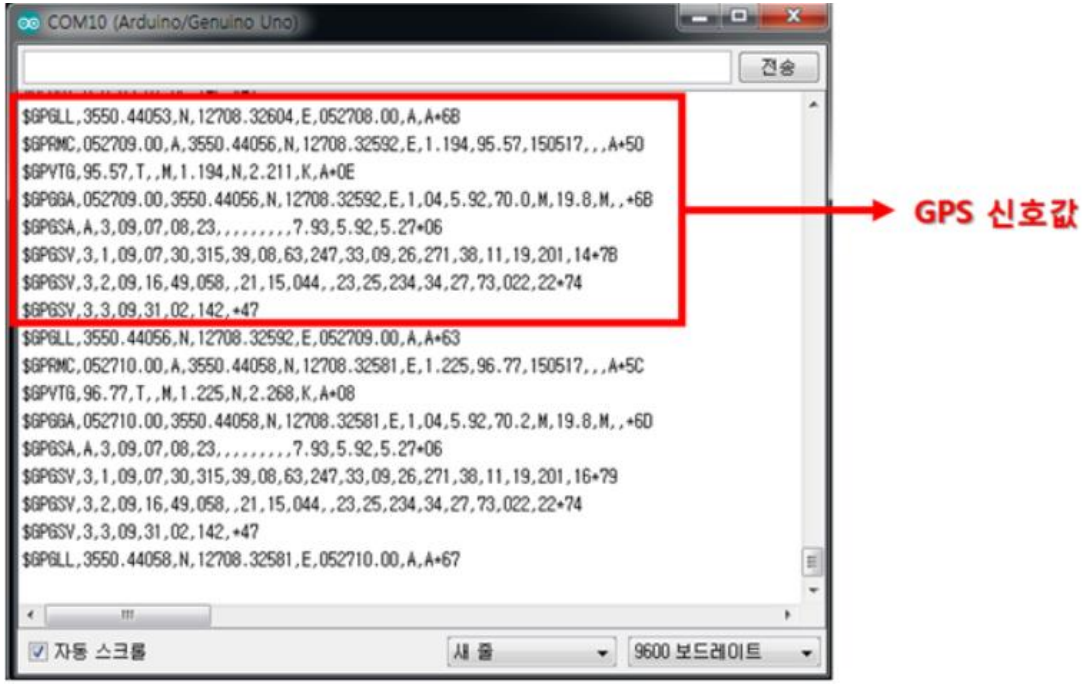
GPS 모듈



아두이노 우노 (Uno)	GPS 수신 모듈
5V	VCC
GND	GND
D12	RX
D11	TX

GPS 모듈

```
#include <SoftwareSerial.h>
SoftwareSerial gps(11,12);

void setup() {
  Serial.begin(9600);
  gps.begin(96000);
}

void loop() {
  if(gps.available()){
    Serial.write(gps.read());
  }
}
```



$GPBOD	Bearing, origin to destination
$GPBWC	Bearing and distance to waypoint, great circle
o $GPGGA	Global Positioning System Fix Data
o $GPGLL	Geographic position, latitude / longitude
o $GPGSA	GPS DOP and active satellites
o $GPGSV	GPS Satellites in view
$GPHDT	Heading, True
$GPROO	List of waypoints in currently active route
$GPRMA	Recommended minimum specific Loran-C data
$GPRMB	Recommended minimum navigation info
o $GPRMC	Recommended minimum specific GPS / Transit data
$GPRTE	Routes
$GPTRF	Transit Fix Data
$GPSTN	Multiple Data ID
$GPVBW	Dual Ground / Water Speed
o $GPVTG	Track made good and ground speed
$GPWPL	Waypoint location
$GPXTE	Cross-track error, Measured
$GPZDA	Date & Time

· 시리얼 모니터를 통해 확인 할 수 있는 신호들은 다음과 같습니다.

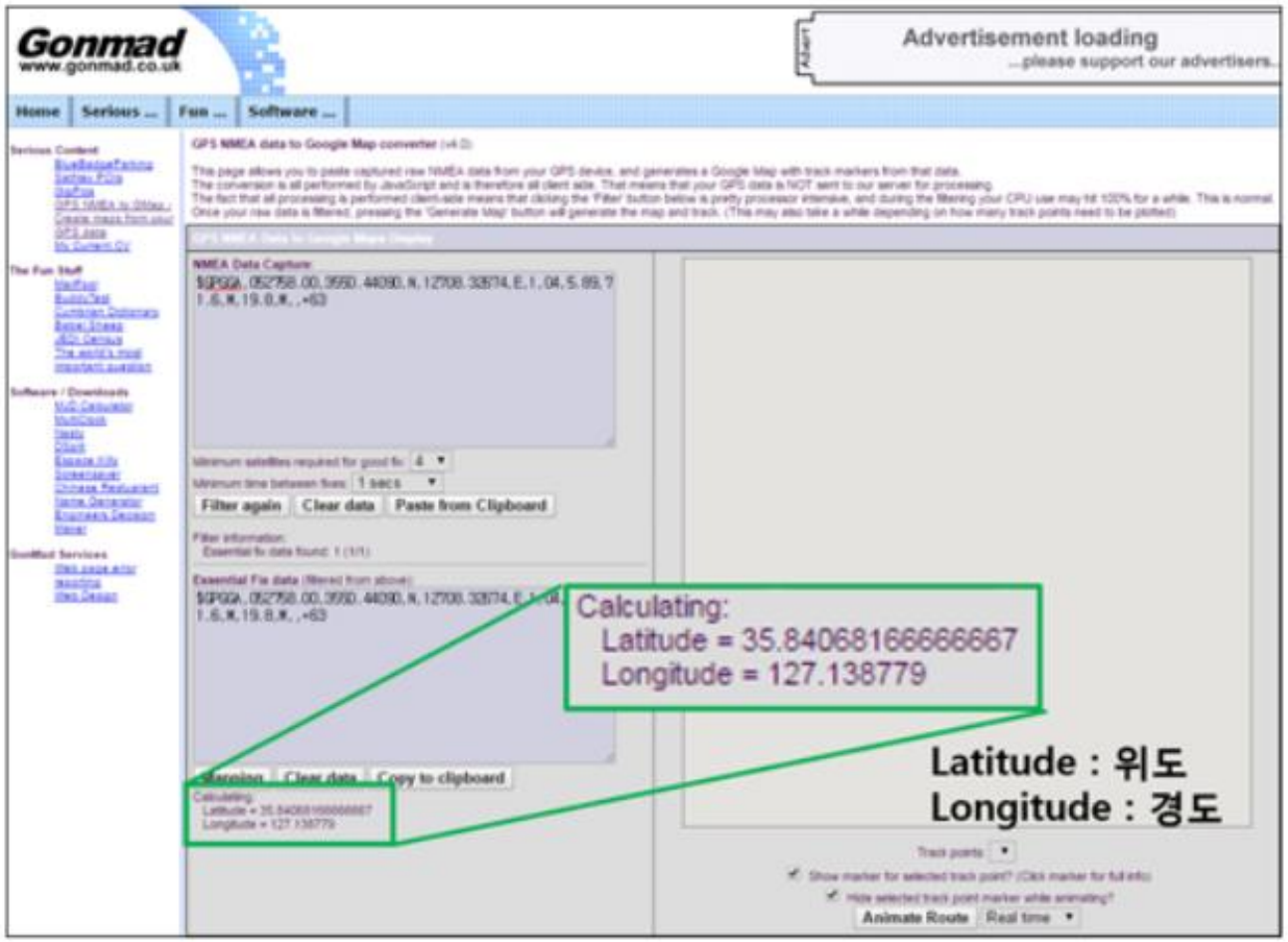
$GPGGA	GPS 의 고정값(기준값)
$GPGLL	지리적 위치의 위도, 경도
$GPGSA	GPS 시스템 소프트웨어와 통신위성
$GPGSV	GPS 위성이 본 경관
$GPRMC	//권장되는 최소한의 GPS / 데이터 교환
$GPVTG	지구 표면상의 항공 궤도와 대지 속도

2) 위도와 경도 확인하기

→ $GPGGA는 GPS 신호의 위치, 시간, 속도 등의 다양한 정보를 축약하여 담고 있습니다.
그렇기 때문에 $GPGGA의 값을 그대로 지도에서 읽어내려 한다면 제대로 된 위치가 나오지 않습니다.
아래 사이트를 통해 $GPGGA값을 우리도 쉽게 알아 볼 수 있는 위도와 경도값으로 바꿔보겠습니다.

사이트 - http://www.gonmad.co.uk/nmea.php

→ 각각의 노란 박스가 있는 곳에 $GPGGA를 붙여넣기 하고 Filter data를 클릭합니다.
'Fix data값을 찾았다'는 표시가 뜬다면 Generate Map을 클릭합니다.

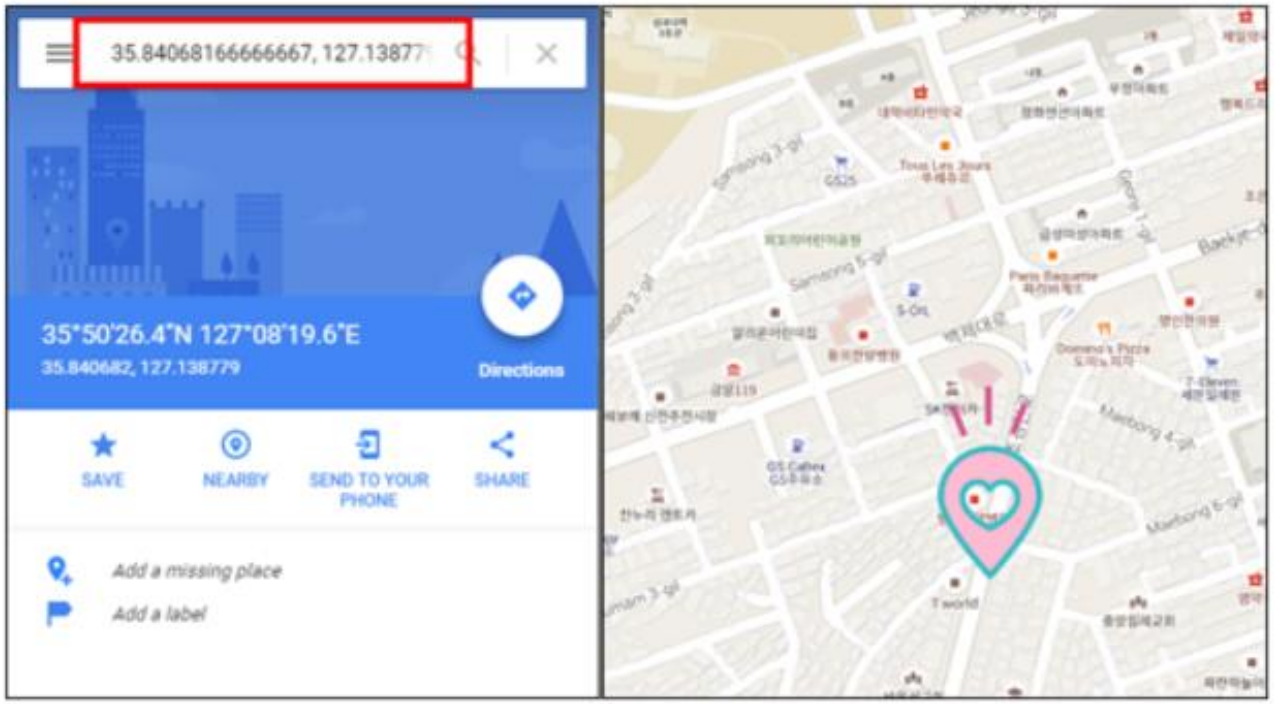


3) 구글맵에서 위치 검색하기

→ 위도와 경도를 구글맵에 붙여넣기하여 현재 위치를 확인합니다.

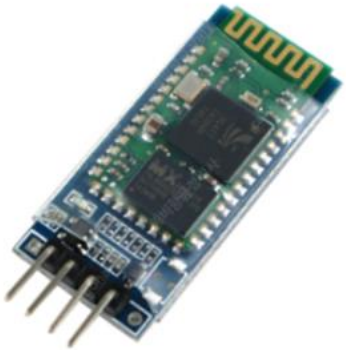
→ 실내에서 테스트했음에도 1~2m에 불과한 오차를 보입니다.

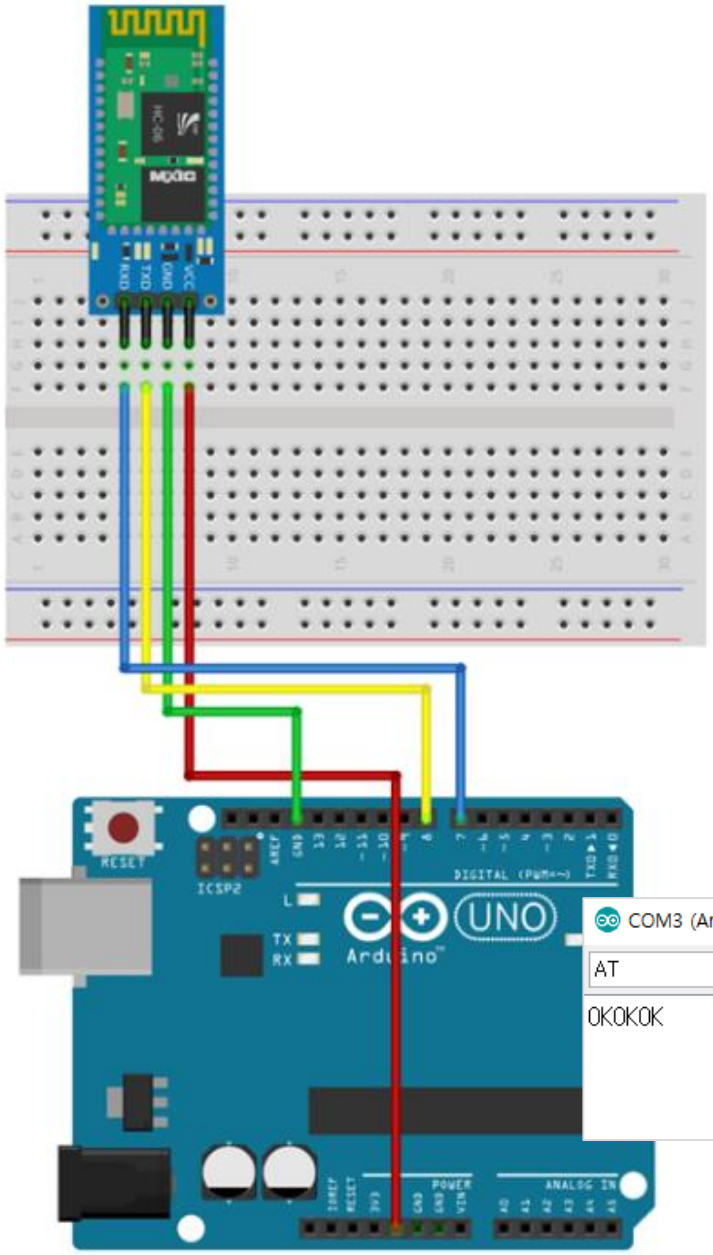
(GPS는 일반적으로 실외보다 실내에서의 오차 범위가 크게 나타납니다.)



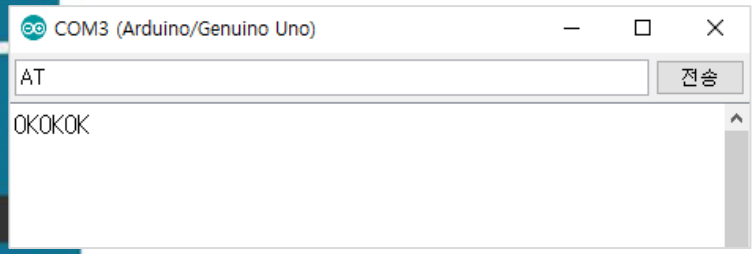
참고 사이트 : http://eduino.kr/product/detail.html?product_no=261&cate_no=55&display_group=1

블루투스

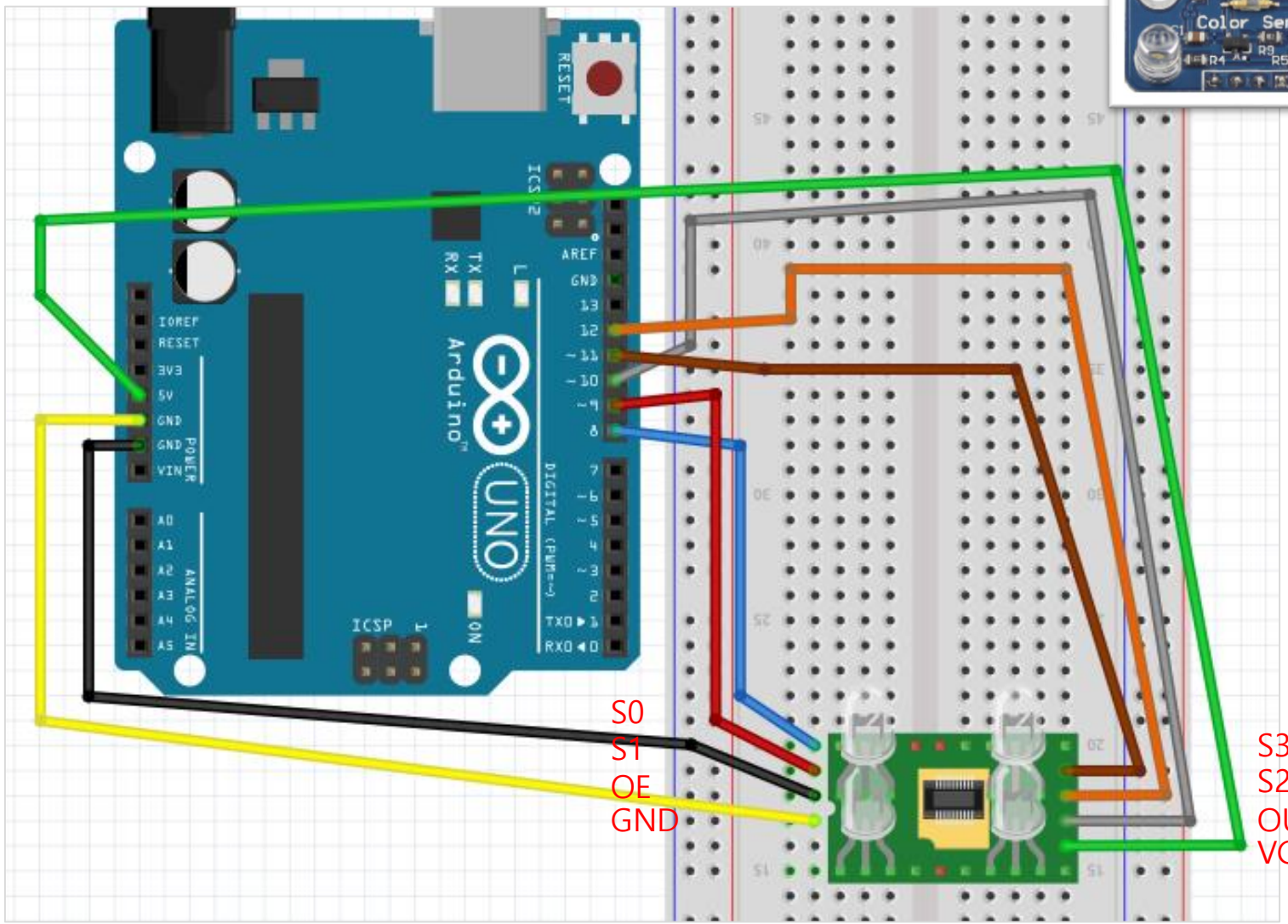




아두이노 우노보드	HC-06
5V	VCC
GND	GND
D8	TXD
D7	RXD



컬러 센서

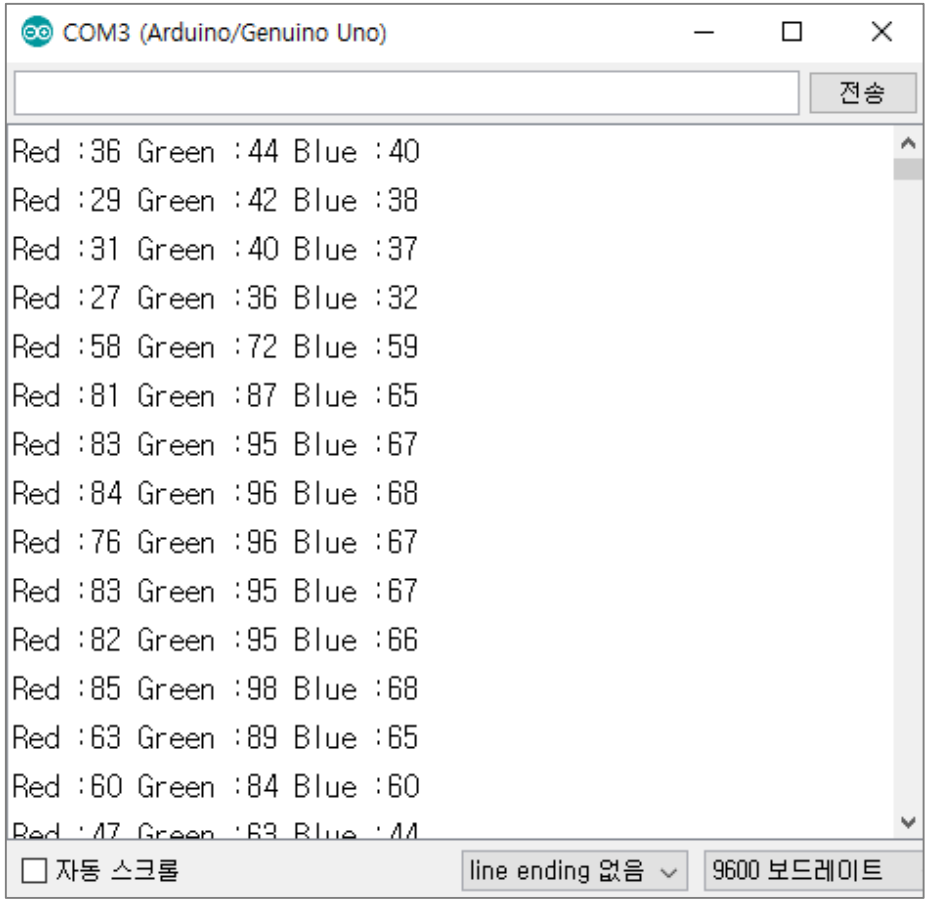


컬러센서

```
const int s0 = 8;
const int s1 = 9;
const int s2 = 12;
const int s3 = 11;
const int out = 10;

int red = 0;
int green = 0;
int blue = 0;

void setup()
{
  pinMode(s0, OUTPUT);
  pinMode(s1, OUTPUT);
  pinMode(s2, OUTPUT);
  pinMode(s3, OUTPUT);
  pinMode(out, INPUT);

  Serial.begin(9600);
  digitalWrite(s0, HIGH);
  digitalWrite(s1, HIGH);
}
```

```

void loop()
{
  color(); //Chama a rotina que le as cores
  Serial.print("Red :");
  Serial.print(red, DEC);
  Serial.print(" Green :");
  Serial.print(green, DEC);
  Serial.print(" Blue :");
  Serial.print(blue, DEC);
  Serial.println();
  delay(2000);
 }
void color()
{
  digitalWrite(s2, LOW);
  digitalWrite(s3, LOW);
  red = pulseIn(out, digitalRead(out) == HIGH ? LOW : HIGH);
  digitalWrite(s3, HIGH);
  blue = pulseIn(out, digitalRead(out) == HIGH ? LOW : HIGH);
  digitalWrite(s2, HIGH);
  green = pulseIn(out, digitalRead(out) == HIGH ? LOW : HIGH);
}
```

S0	S1	S2	S3	
HIGH	HIGH	LOW	LOW	→ RED
HIGH	HIGH	LOW	HIGH	→ BLUE
HIGH	HIGH	HIGH	HIGH	→ GREEN


COM3 (Arduino/Genuino Uno)
전송
Red :36 Green :44 Blue :40
Red :29 Green :42 Blue :38
Red :31 Green :40 Blue :37
Red :27 Green :36 Blue :32
Red :58 Green :72 Blue :59
Red :81 Green :87 Blue :65
Red :83 Green :95 Blue :67
Red :84 Green :96 Blue :68
Red :76 Green :96 Blue :67
Red :83 Green :95 Blue :67
Red :82 Green :95 Blue :66
Red :85 Green :98 Blue :68
Red :63 Green :89 Blue :65
Red :60 Green :84 Blue :60
자동 스크롤
line ending 없음
9600 보드레이트