아두이노 Learning Kit 매뉴얼

(실속/초급자/중급자키트 포함)



Last Update : 19/03/13

By : 키트플러스

시작하며.

본 매뉴얼은 아두이노 Learing Kit 키트에 관한 사용자 메뉴얼입니다.

아두이노 Learning 키트는 코딩을 처음 시작하시는 분, 아두이노를 처음 접하시는 분, 그리고 첫걸음/입문자 키트를 마치신 분 들에게 매우 적합한 키트입니다.

키트플러스의 아두이노 키트는 납땜 작업 없이 점퍼 작업만으로 아두이노의 기본적인 동작들을 공부 할 수 있습니다. Learing 키트로 배울 수 있는 내용들은 아두이노 IDE 사용법, 프로그램 작성 방법, 컴파일 및 업로드, 시리얼 모니터를 활용한 디버깅 방법과 브래드보드를 활용한 여러 디바이스를 이해하고 제어하실 수 있습니다.

Learning 키트에서 기본 제공되는 디바이스들은 캐릭터 LCD, LM35 온도센서, 회로 내장부저, 무회로부저, 스탭모터, 서보모터, 7 세그먼트, IR 송수신, 8x8 Dot matrix, 불꽃센서, 조도센서, 기울기센서, 74HC595, XY 조이스틱, DHT11 온습도센서, 수위센서, 소리센서, 등이 있습니다..

본 문서가 아두이노 코딩교육이 필요하신 분들에게 많은 도움이 되었으면 합니다.

"나는 나의 의지대로 된다."

-쇼펜하우어

목 차

아두이노란 **5**

하드웨어 구성 **6**

아두이노 우노 R3 보드 6

Learning 키트 구성 8

브래드보드 11

아두이노 보드 PC 연결 12

소프트웨어 시작하기 13

아두이노 IDE 다운로드 13

아두이노 IDE 설치 14

아두이노 IDE 및 소프트웨어 구조 18

아두이노 라이브러리 추가 19

아두이노 보드 선택 20

컴파일 및 업로드 21

시리얼 모니터 사용법 23

아두이노 코딩절차 24

예제 학습하기 **25**

LED blink 25

RGB LED 28

버튼입력 32

회로내장 부저 35

무회로 부저 37

틸트스위치 39

서보모터 41

4x4 키패드 45

DHT11 온습도센서 48

조이스틱 52
IR 적외선 수신 54
IR 적외선 송신 57
수위센서 60
Clock 모듈 62
소리감지센서 69
1602 LCD 71
74HC595 74
아날로그 입력(포텐션미터) 78
조도센서 81
7 세그먼트 84
4 자리 7 세그먼트 89
릴레이 93
스탭모터 97
불꽃감지센서 102
LM35 온도센서 105
NTC 온도센서 108
10 BAR LED 세그먼트 113
TCRT5000 근접센서 116
도트매트릭스 119
아두이노 프로토 쉴드 123
초음파센서 125
참고자료 132

What is ARDUINO ?



아두이노는 2005년 이탈리아에서 개발된 오픈 소스 기반을 한 단일 보드를 뜻합니다.

아두이노가 세상에 나오기 전에는 마이크로컨트롤러라는 것을 비전공자가 공부하기 어려웠습니다. 일반인이 자료를 구하는 것도 쉽지 않았고, 개발환경을 구축하는 것도 복잡한 과정이 필요했습니다. 그리고 하드웨어를 구입해야 하는 비용도 저렴하지 않았습니다.

하지만 아두이노는 저렴한 보드와 다양한 부품 그리고 강력한 소프트웨어의 지원으로 하드웨어에 익숙지 않는 학생에게 코딩학습을 도와 줍니다.

즉, 아두이노는 임베디드시스템이라는 것을 누구나 쉽게 개발할 수 있는 환경을 제공합니다.

현재도 아두이노와 관련된 수많은 센서와 장치들이 새롭게 등장하고 있습니다.

하드웨어 장치와 더불어 누구나 사용이 가능한 수많은 라이브러리도 계속 업데이트 되고 있습니다.

아두이노는 오픈하드웨어/오픈소스 기반으로 하고 있기 때문에 웹에서 기술 자료를 찾는 것이 어렵지 않습니다. 그

다양한 아두이노 보드가 있지만 본 매뉴얼에서는 "아두이노 우노 R3"를 사용한 Learning 키트 사용법에 대해서 설명을 드리겠습니다.

□ 하드웨어 구성

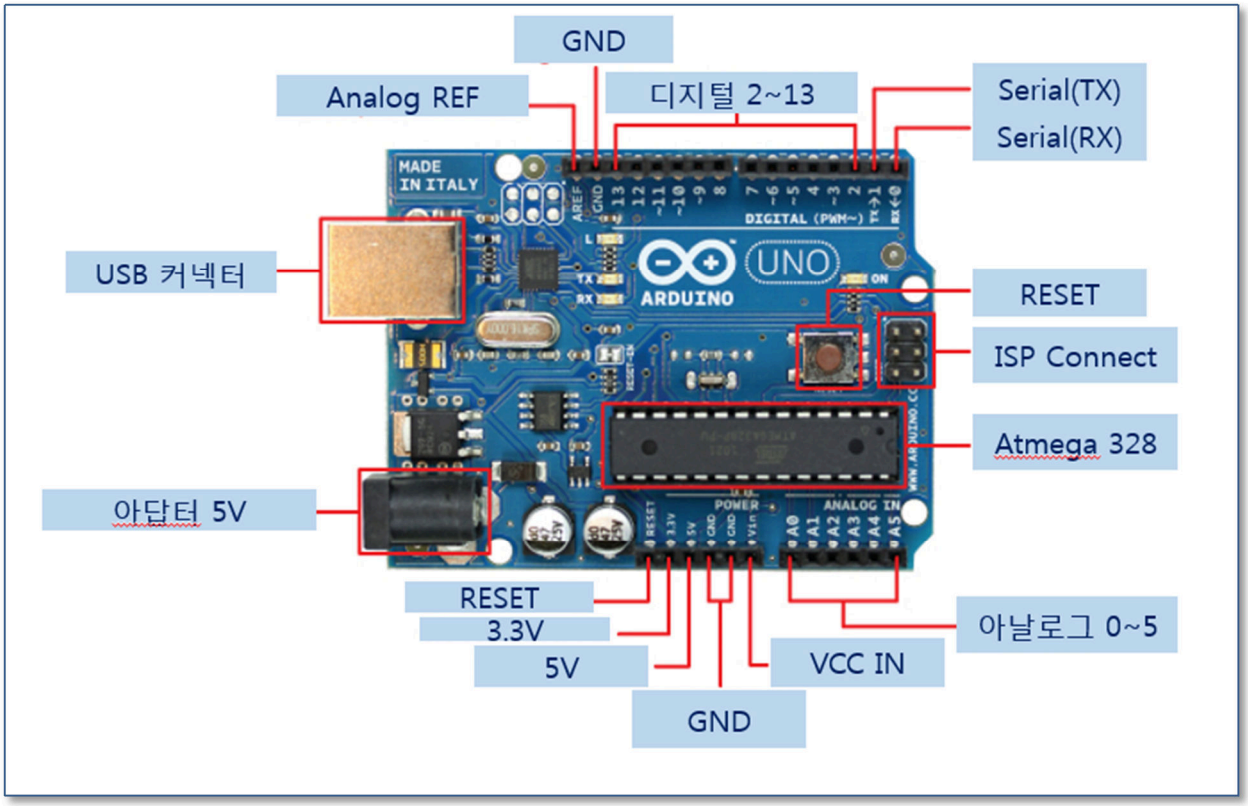
○ 아두이노 우노 R3보드

아두이노 우노 보드는 ATmega328 기반의 마이크로 컨트롤러 보드입니다.

14개의 디지털 I/O핀(14개 중 6개의 핀은 PWM 출력으로 사용 가능)과 6개의 아날로그 입력, 16MHz ceramic resonator로 구성되며, 외부 USB 커넥터, 파워 커넥터, ISP 헤더, 리셋버튼이 내장되어있습니다.

쉽게 컴퓨터와 USB 케이블로 연결할 수 해서 사용할 수 있습니다. 전력소모가 많은 디바이스(ex. 모터)를 사용할 경우에는 추가로 아답터가 필요할 수도 있습니다.

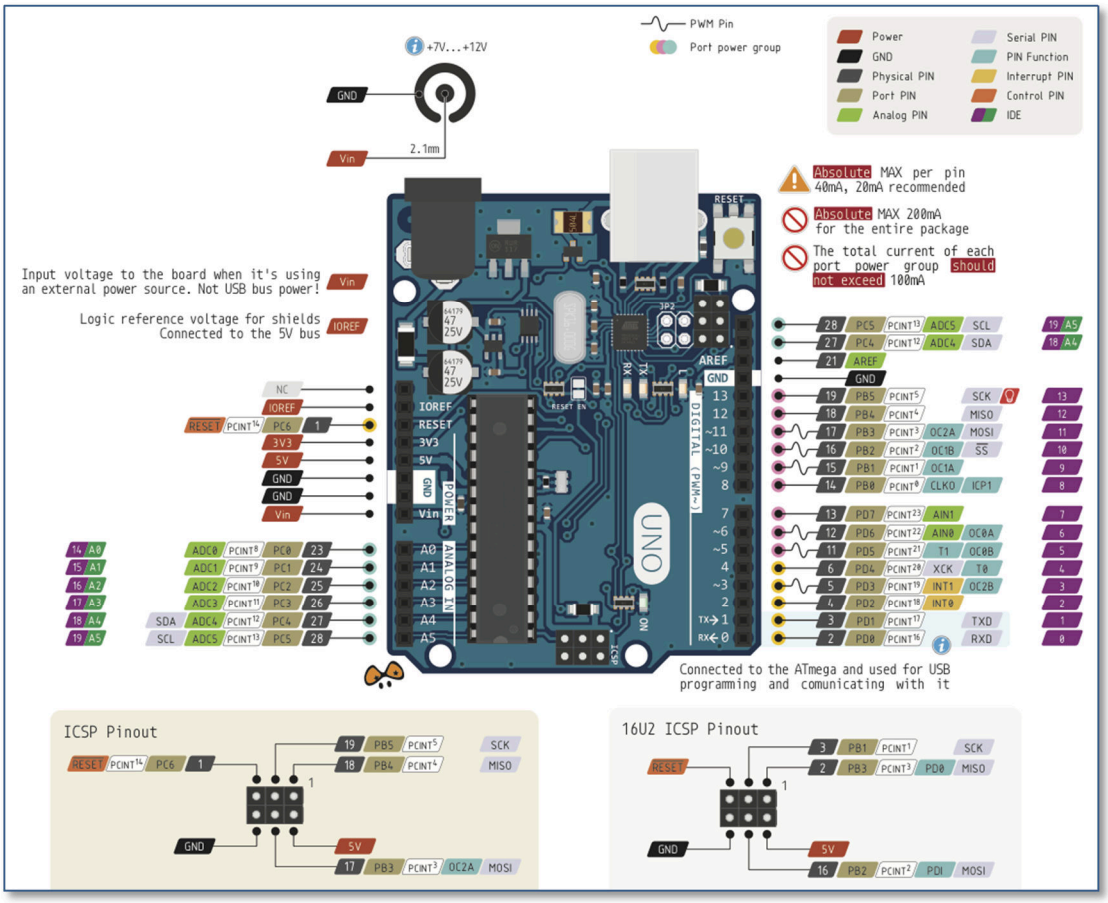
아두이노 우노 정품보드는 USB-to-Serial는 Atmega16U2로구성되어있고, 호환보드는 FT232 또는 CH340으로 구성되어있습니다. 만약 CH340 호환보드를 사용하실 경우에는 별도의 드라이버를 설치하셔야 합니다. 외형적인 핀맵 및 기본 구성은 정품/호환 보드 모두 동일합니다.



[정품 아두이노 우노 구성도]

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by boot loader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

[아두이노 우노 R3 주요 사양]



(출처 : http://forum.arduino.cc/index.php?topic=146315.0)

[아두이노 우노 핀맵]

○ Learning 키트 구성

Learning Kit 는 초/중/상급자 키트로 구분이 되어있습니다.

- 초급자 키트 구성품

No	Part	한글명
1	UNO R3 Board	아두이노 우노 호환보드
2	USB Cable	USB 케이블
3	LCD1602 Character LCD	16x2 캐릭터 LCD
4	830 Hole breadboard	830 홀 브레드보드
5	LM35 Temperature Sensor	LM35 온도센서
6	Active Buzzer	액티브 부저
7	Passive Buzzer	피에조 부저
8	Stepper Motor	스텝모터
9	Stepper motor Driver	스텝모터 드라이버 모듈
10	9G Servo Motor	서보모터
11	7 Segment Display	FND 7 세그먼트
12	4 segment Segment display	4 자리 FND 7 세그먼트
13	IR Mini Remote	적외선 리모콘
14	8*8 LED Matrix	도트메트릭스(1 또는 2 color)
15	74HC595	74HC595
16	Photoresistor (5516 5mm)	조도센서
17	10 kΩ Variable resistor pots	가변저항(10k 옴)
18	Flame sensor	불꽃감지센서
19	Ball Switch	기울기센서
20	IR Receiver	IR 수신 센서
21	9V Battery snap	9V 배터리 커넥터
22	Red LED	빨강 LED
23	Green LED	녹색 LED
24	Yellow LED	노랑 LED
25	Resistor Kit	3 종 저항세트(220,1K,10K)
26	Push Buttons	택트스위치
27	Push Buttons cap	택트스위치 캡
28	Plastic Box	키트보관박스

29	Jumper Wires solderless Male to Male	핀점퍼 세트
30	Female to male dupond line	무지개 점퍼
31	Single Row 1*40mm Pin Header Male	핀헤더
32	UNO Proto shield	확장보드
33	9V Battery	9V 배터리
34	Jumper cap	부품보관함
35	IR LED	적외선 LED
36	Variable resistor cap	가변저항 캡
37	RGB LED	RGB LED
38	10K NCT thermistor	NTC 온도센서
39	Ultrasonic sensor	초음파센서

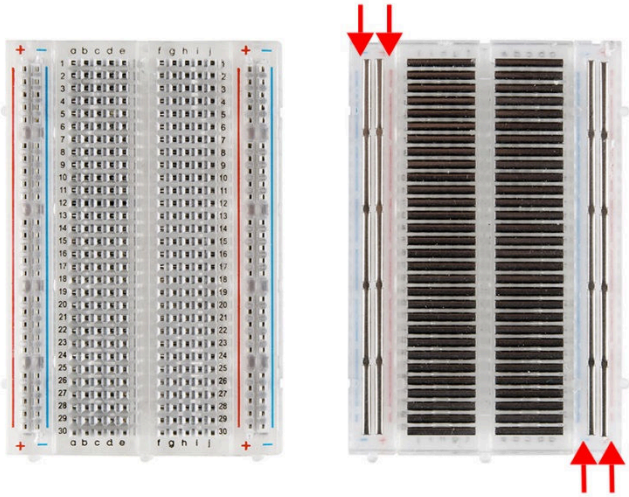
- 중급자 키트 구성품

No	Part	한글명
1	UNO R3 Board	아두이노 우노 호환보드
2	USB Cable	USB 케이블
3	LCD1602 Character LCD	16x2 캐릭터 LCD
4	830 Hole breadboard	830 홀 브레드보드
5	LM35 Temperature Sensor	LM35 온도센서
6	Active Buzzer	액티브 부저
7	Passive Buzzer	피에조 부저
8	Stepper Motor	스텝모터
9	Stepper motor Driver	스텝모터 드라이버
10	9G Servo Motor	서보모터
11	7 Segment Display	FND 7 세그먼트
12	4 segment Segment display	4 자리 FND 7 세그먼트
13	IR Mini Remote	적외선 리모콘
14	8*8 LED Matrix	도트메트릭스(1 또는 2 color)
15	74HC595	74HC595
16	Photoresistor (5516 5mm)	조도센서
17	10 kΩ Variable resistor pots	가변저항(10k 옴)
18	Flame sensor	불꽃감지센서
19	Ball Switch	기울기센서

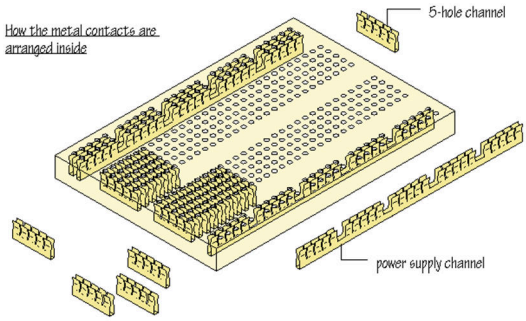
20	IR Receiver	IR 수신 센서
21	9V Battery snap	9V 배터리 커넥터
22	Red LED	빨강 LED
23	Green LED	녹색 LED
24	Yellow LED	노랑 LED
25	220 ohms Resistor	3 종 저항세트(220,1K,10K)
26	Push Buttons	택트스위치
27	Push Buttons cap	택트스위치 캡
28	Plastic Box	키트보관박스
29	Jumper Wires solderless Male to Male	핀점퍼 세트
30	Female to male dupond line	무지게 점퍼 20cm
31	Pin header 2.54	핀헤더
32	XY joystick module	조이스틱 모듈
33	Temperature module	DHT11 온습도센서
34	Water test module	수분센서
35	Sr501	인체감지센서
36	RGB LED	3 색 LED
37	TTP223B	터치센서모듈
38	Sound Module	소리감지센서
39	Relay Module	릴레이 모듈
40	Clock Module	RTC 모듈
41	4*4 Key board	4X4 키패드
42	Mini bread board	미니 브래드보드
43	10K NCT thermistor	NTC 온도센서
44	Bar LED segment	바 타입 LED 세그먼트
45	TCRT5000	적외선 근접센서
46	Variable resistor cap	가변저항 캡
47	Ultrasonic sensor	초음파센서
48	IR LED	적외선 LED
49	RGB 3 color module	RGB LED

○ 브래드보드

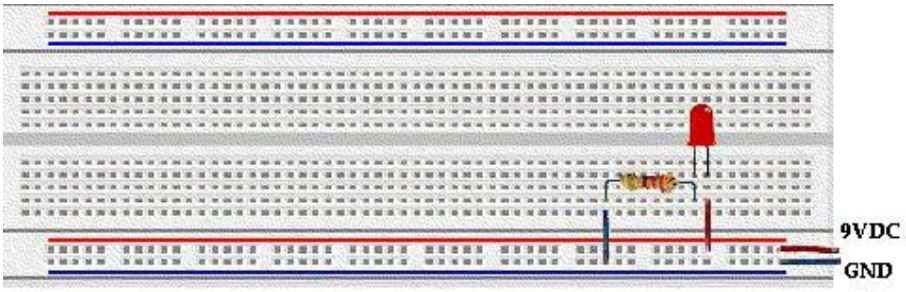
브래드보드란 납땜 작업 없이 전자부품들을 서로 빠르고 쉽게 연결 할 수 있는 보드입니다. 기본 형태는 여러 개의 홀(구멍)이 촘촘히 배치되어있습니다. 홀 내부는 "a,b,c,d,e" 가 서로 연결이 되어있고, +,- 의 전원선도 수직으로 연결 되어있는 구조입니다.



출처 : SparkFun Electronics



출처 http://secte1.elec.uow.edu.au

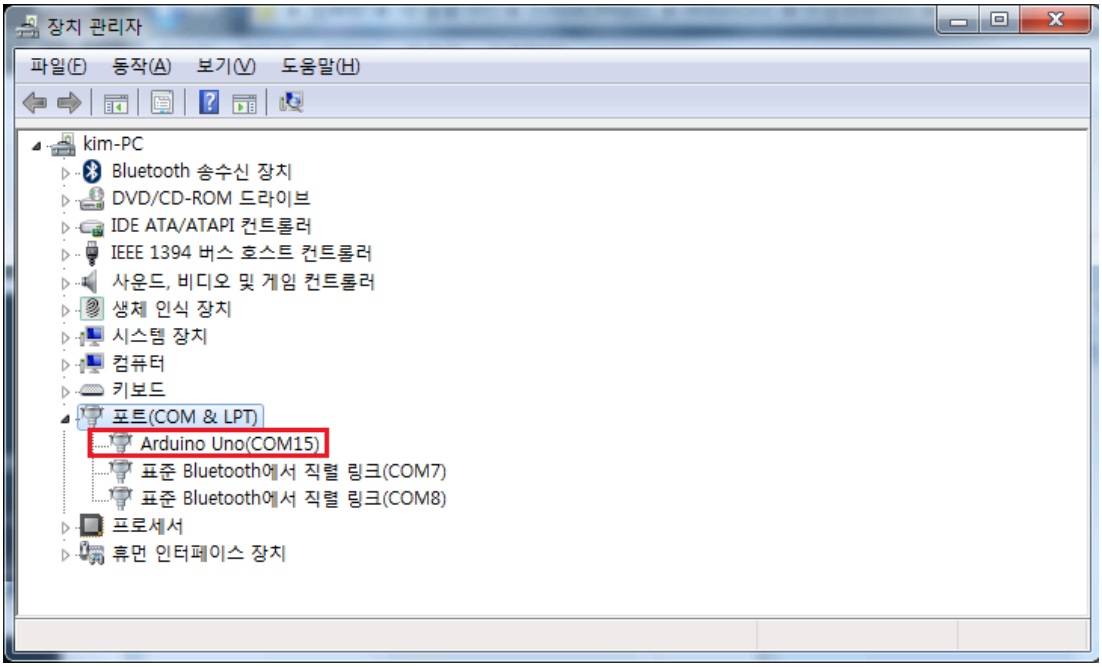


[브래드보드 내부 구조]

○ 아두이노 보드 PC 연결

USB 케이블을 아두이노와 PC 에 연결하고, 장치관리자에서 COM 포트를 확인합니다.





□ 소프트웨어 시작하기

○ 아두이노 IDE 다운로드

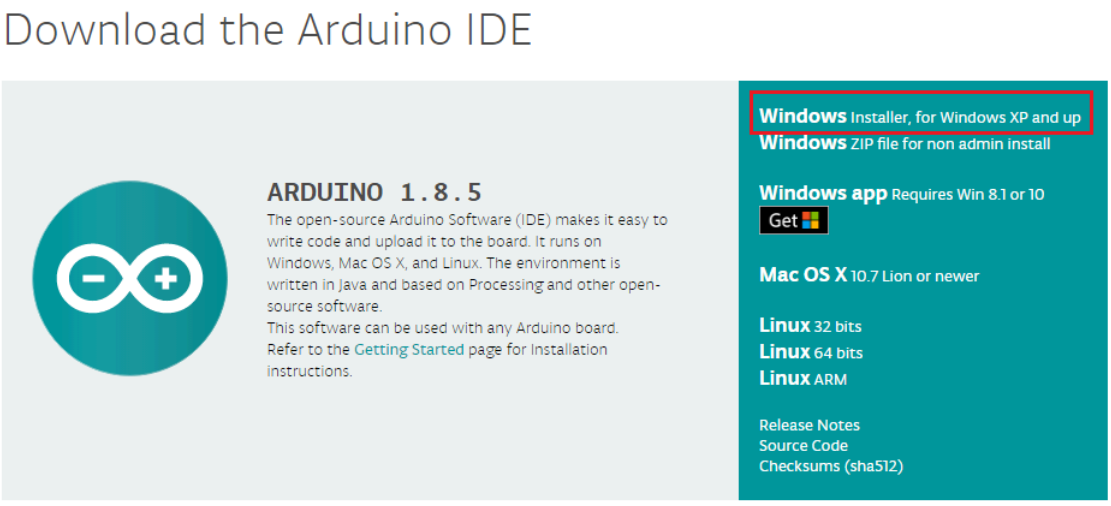
하드웨어 보드가 준비되셨다면, 아두이노 소프트웨어가 필요합니다. 아두이노는 코딩, 컴파일, 업로드가 가능한 IDE를 기본 제공합니다.

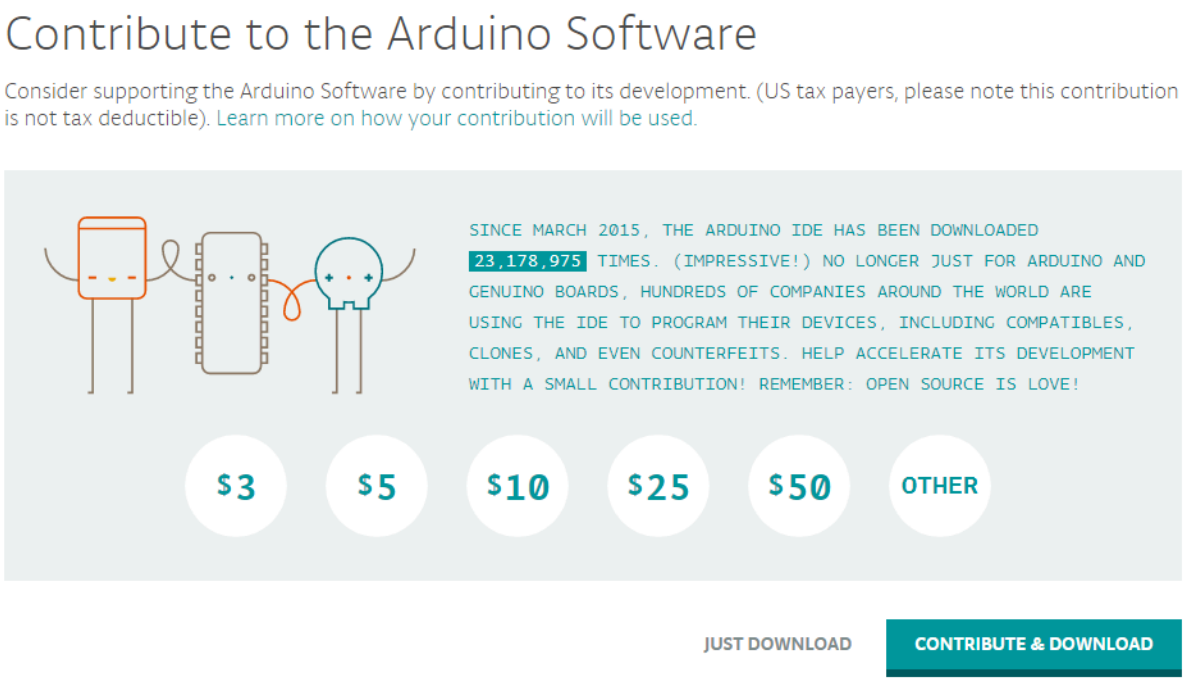
> *"IDE란, 통합 개발 환경(統合開発環境, Integrated Development Environment, IDE)으로 코딩, 디버그, 컴파일, 배포 등 프로그램 개발에 관련된 모든 작업을 하나의 프로그램 안에서 처리하는 환경을 제공하는 소프트웨어를 뜻합니다."*

아두이노 IDE 다운로드 홈페이지를 방문하시면 web을 이용한 아두이노 editor가 있지만, 본 매뉴얼에서는 Windows 기반의 아두이노 IDE에 대해서 설명을 합니다.

가장먼저 아래 링크에서 아두이노 IDE를 다운로드 받습니다.

아두이노 다운로드 링크 : https://www.arduino.cc/en/Main/Software





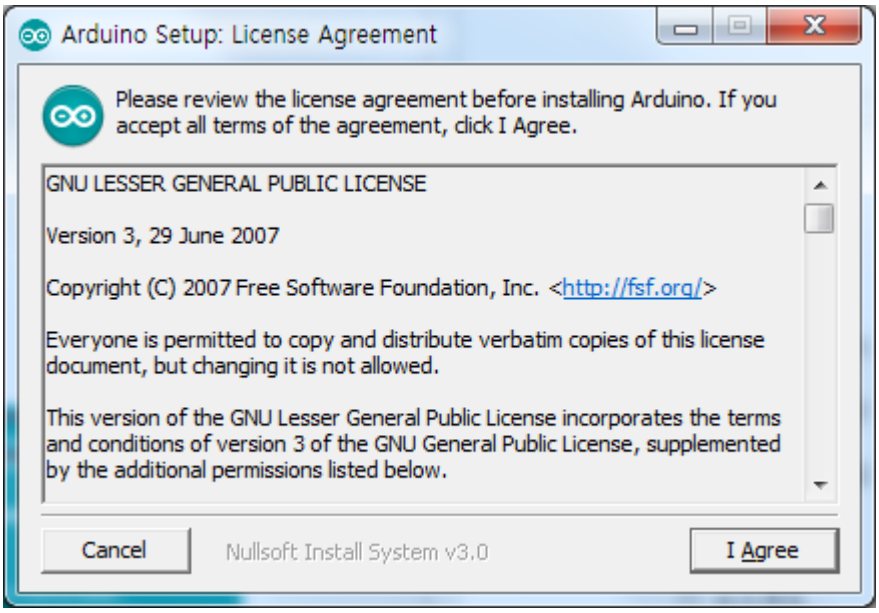
기부를 원하시는 분들은 "Contribute & DOWNLOAD"를 누르시고,

아니시라면 "JUST DOWNLOAD"를 눌러서 다운로드를 진행합니다.

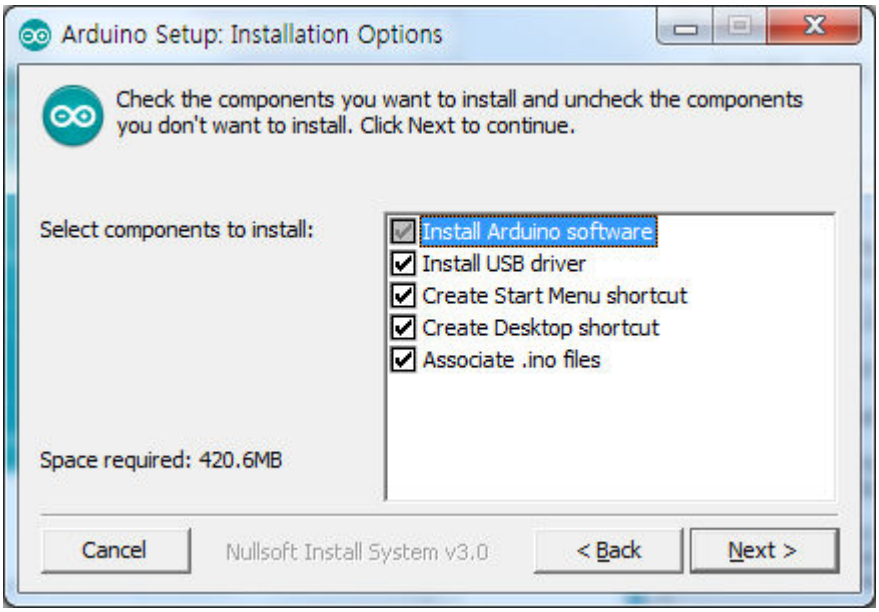
○ 아두이노 IDE 설치

다운로드 받은 "arduino-1.8.5-windows.exe" 파일을 실행합니다.(버전은 변경 될 수 있습니다.)

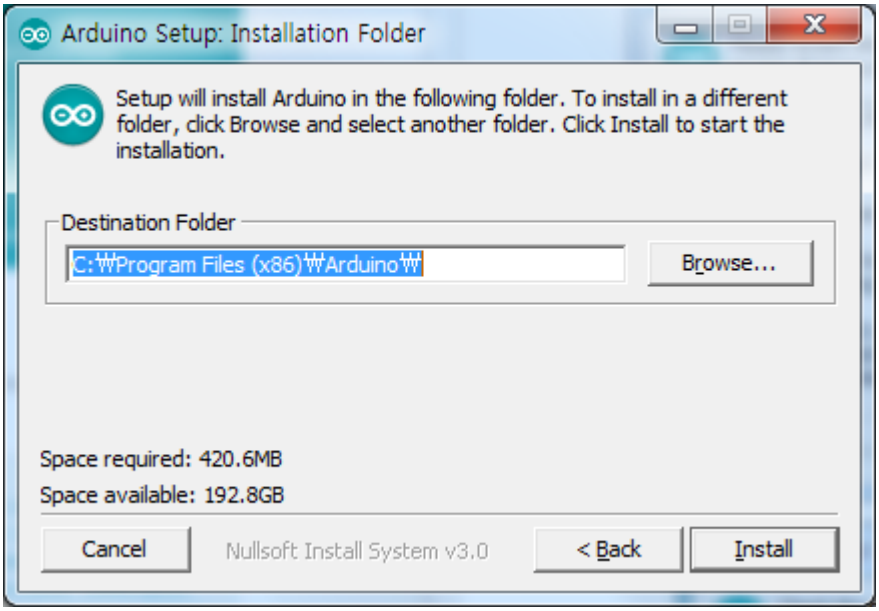
아래 라이선스 정책에 "I Agree"(동의) 버튼을 누릅니다.



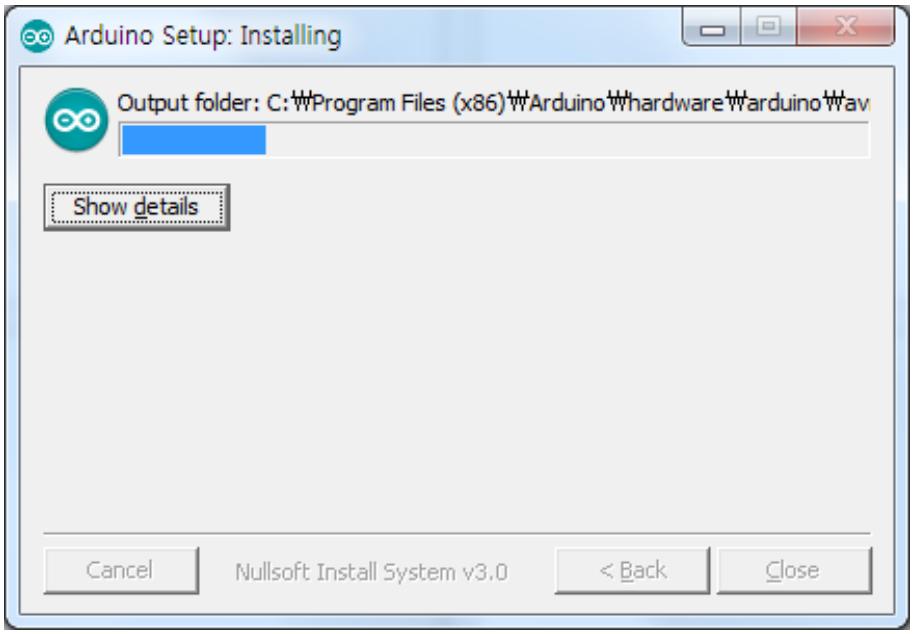
설치 옵션에서 USB Driver를 포함해서 모두 체크하시고 "NEXT>"를 누릅니다.



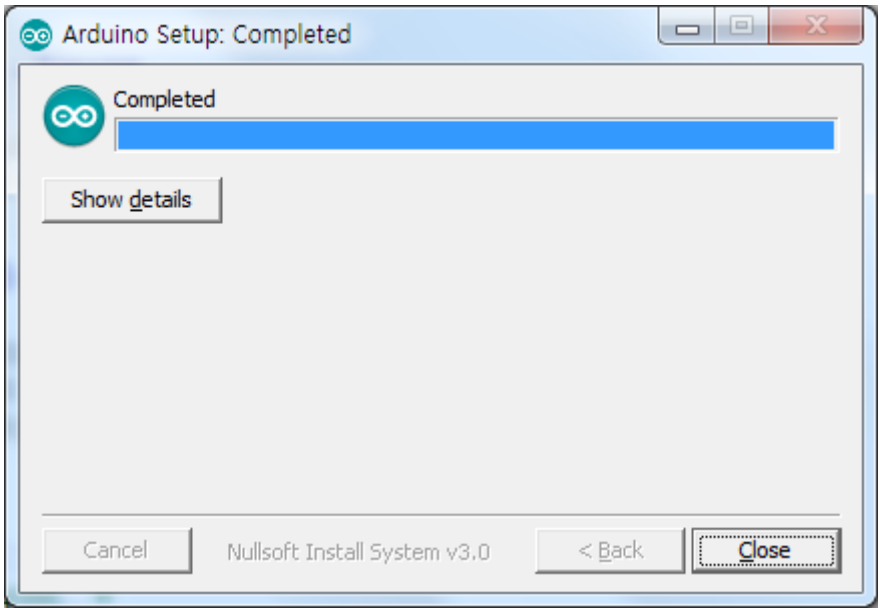
설치 경로를 입력하신 후 "Install"를 누릅니다.



설치를 시작합니다. 2~3분정도의 시간이 필요합니다.



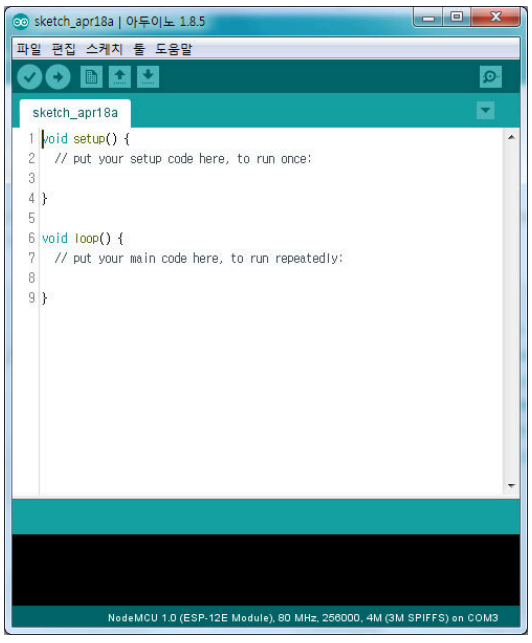
설치가 완료되면 "Close"를 눌러 설치를 종료합니다.



바탕화면의 "Arduino" 아이콘을 두번 클릭해서 아두이노 IDE를 실행합니다.

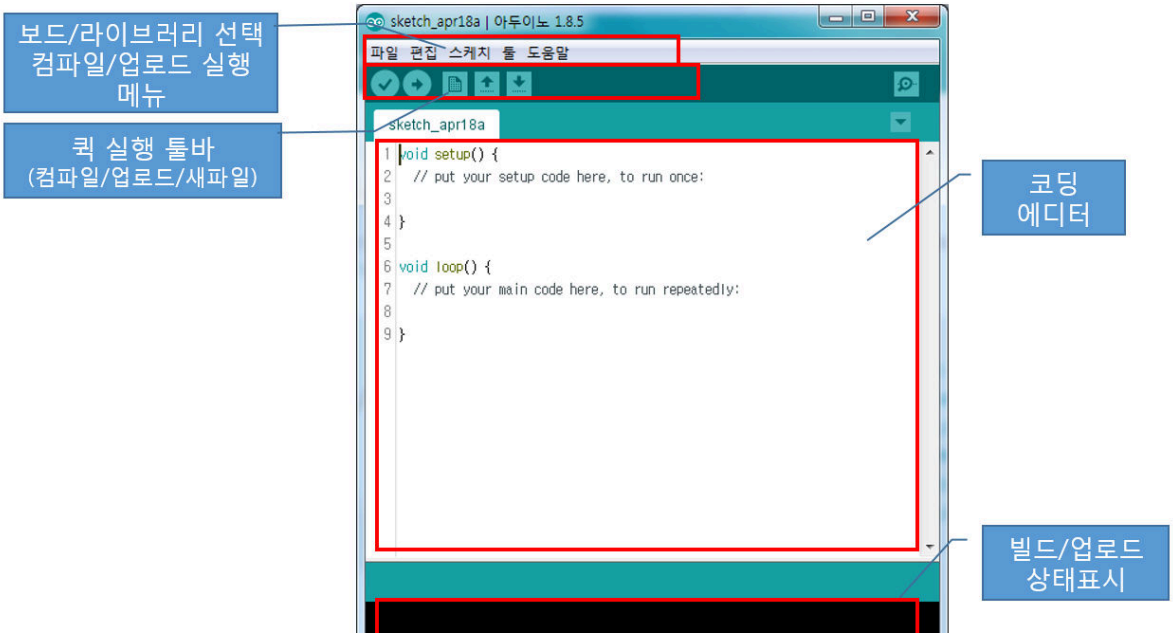






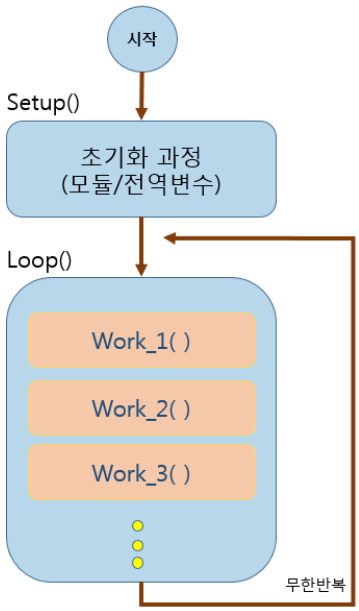
○ 아두이노 IDE 및 소프트웨어 구조

아두이노 IDE는 크게 메뉴와 코딩 에디터, 상태 표시 콘솔로 구성되어있습니다.



아두이노의 소프트웨어 구조는 객체지향 C++ / C 를 사용합니다.

전원인가 이후 RESET이 최초로 실행되는 코드는 Setup() 함수에서 필요한 모듈 및 변수들을 초기화 합니다. 이후 Loop()에서 무한 반복을 하면서 일을 수행합니다.



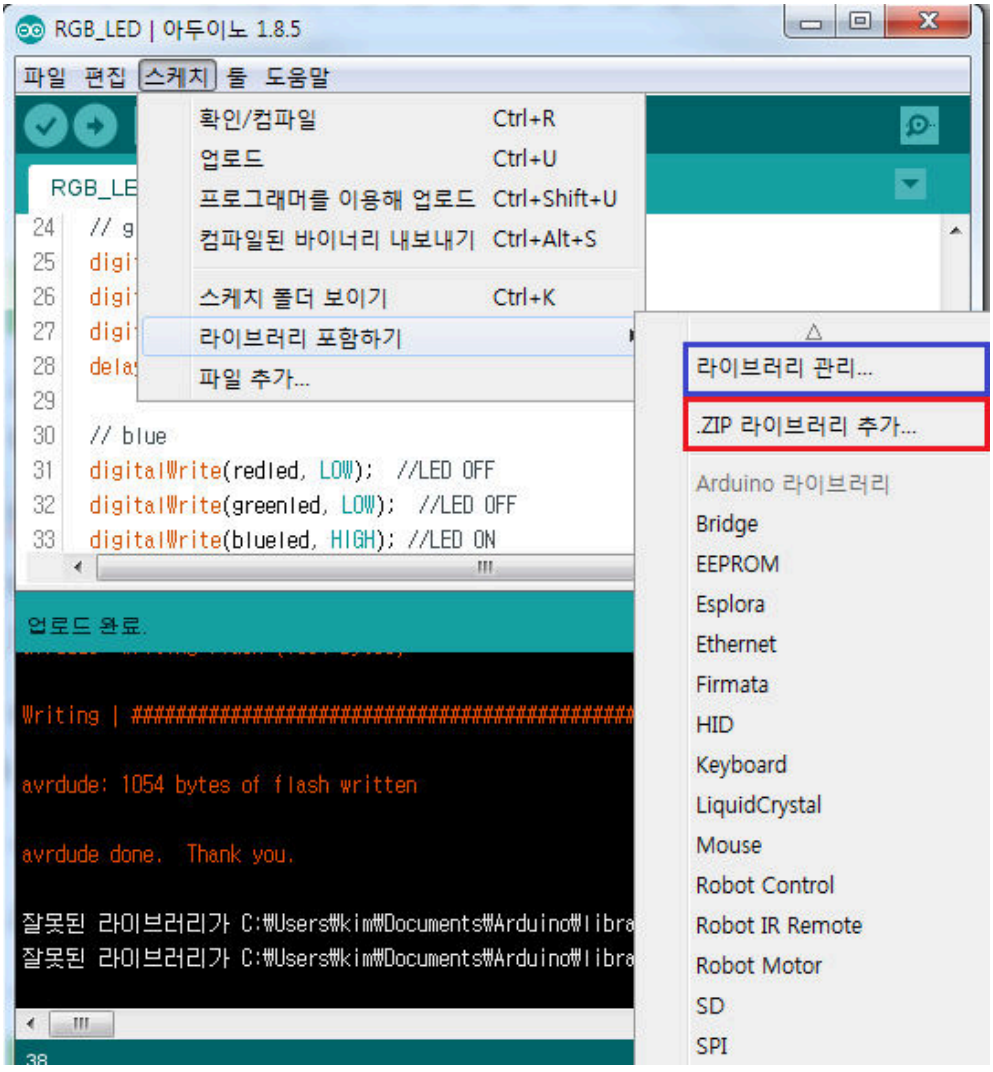
○ 아두이노 라이브러리 추가

소프트웨어 라이브러리란 센서, 디스플레이, 모터 등과 같은 모듈들을 쉽게 사용하기 위한 코드들의 집합체입니다.

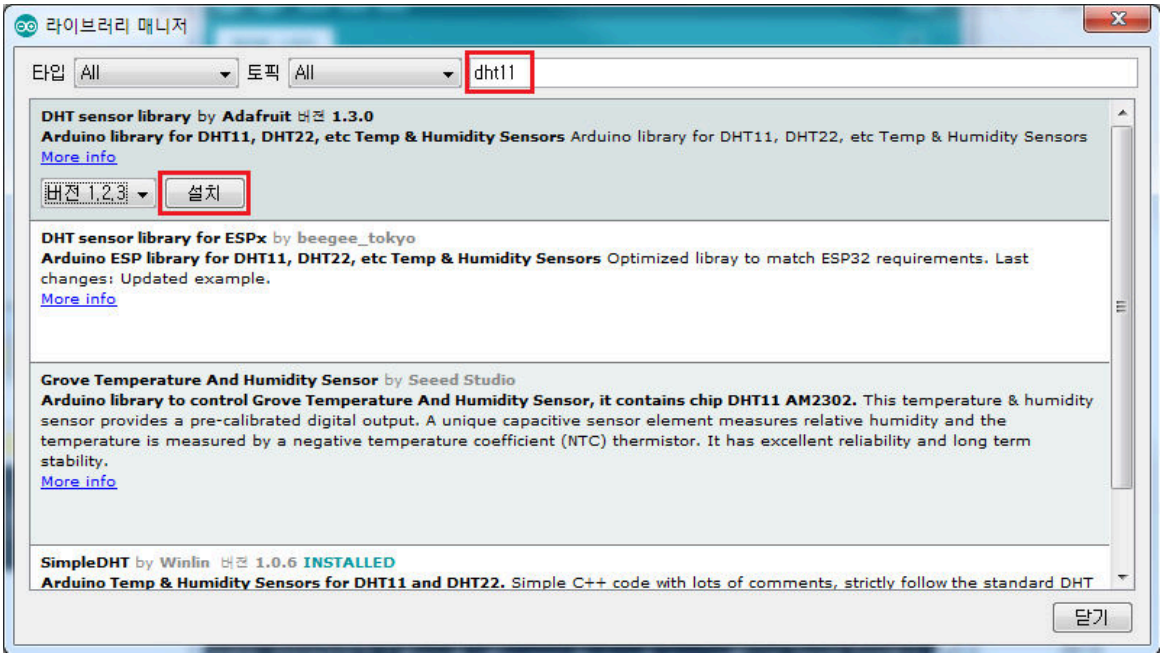
아두이노 IDE에는 기본라이브러리가 포함되어있습니다.

하지만 기본 라이브러리에 없는 새로운 모듈을 사용하기 위해서는 라이브러리를 추가로 설치해야 합니다.

다음 그림은 온습도 센서인 DHT11를 사용하기 위한 라이브러리를 설치하는 과정입니다. 아두이노 온라인을 이용한 라이브러리 추가는 "라이브러리 관리"를 선택하시면 됩니다. 별도의 라이브러리를 ZIP으로 다운로드 받으셨다면 ".ZIP 라이브러리 추가"를 누르시고 탐색창에서 라이브러리 ZIP파일만 선택해주시면 됩니다.



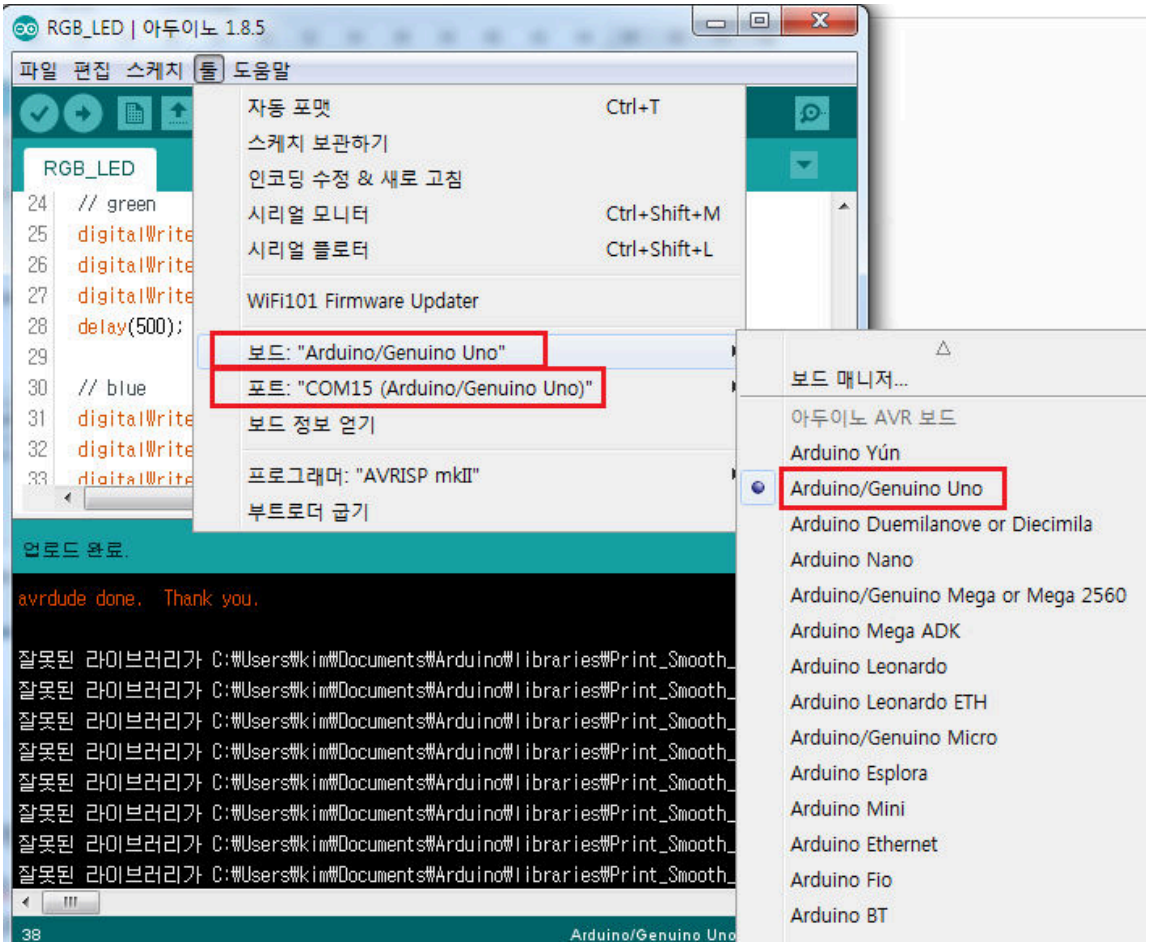
"라이브러리 관리"를 누르시면 아래와 같은 창이 나옵니다. 찾으시는 라이브러리를 검색을 하시면 관련된 라이브러리들이 표시가 됩니다. 설치버튼을 누르시면 라이브러리가 설치가 됩니다.



○ 아두이노 보드 선택

아두이노는 학습자/사용자 요구에 맞는 다양한 보드들을 있습니다. 정식으로 출시하고 있는 아두이노 미니, 아두이노 마이크로, 아두이노 레오나로드, 아두이노 메가 등이 있으며, 3rd 파트 MCU들은 ESP8266, STM32도 지원이 가능합니다.

다음 그림은 앞으로 사용하는 아두이노 우노 보드와 통신포트를 선택 과정입니다.



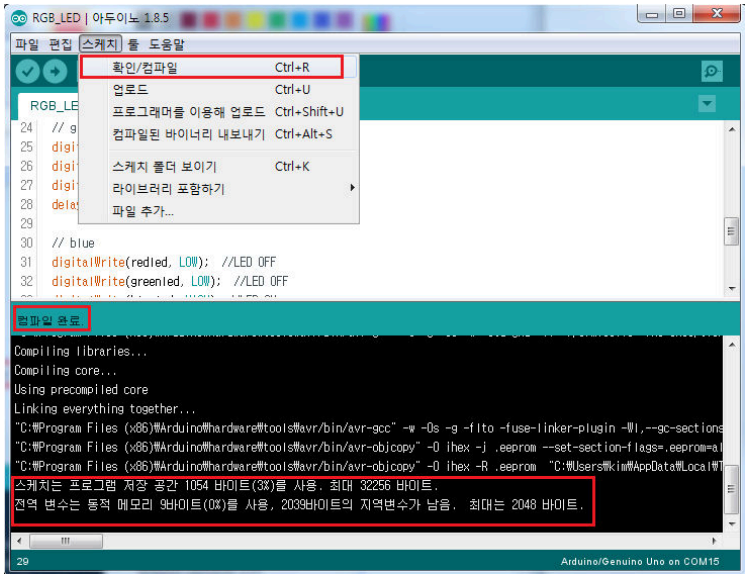
○ 컴파일 및 업로드

코딩을 완료하셨다면, 컴파일 과정을 거쳐 실행파일을 만들어야 합니다. 에러가 없다면 만들어진 실행파일을 아두이노로 업로드를 진행해야 합니다.

"컴파일이란 사람이 작성하는 프로그램(고급언어)을 기계가 인식할 수 있는 기계어로 변환하는 과정을 뜻합니다."

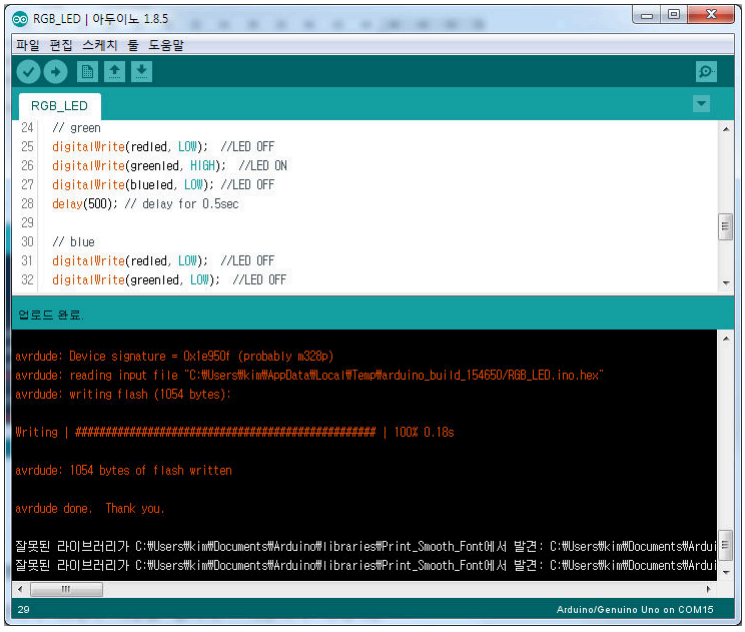
"업로드란 컴파일 과정을 거쳐 생성된 실행파일을 마이크로컨트롤로 내부 또는 외부 메모리로 주입하는 과정을 뜻합니다"

다음 그림은 스케치 -> 확인/컴파일을 실행한 그립니다. 컴파일이 완료되었다면 아래 콘솔창에 사용되는 메모리 사이즈가 표시됩니다. (단축키 Ctrl + R)



다음 그림은 스케치->업로드를 실행한 화면입니다. 주입이 100% 된 것을 확인 하시면 됩니다.

만약 COM port Error가 발생된다면, 장치관리자의 COM port와 아두이노 COM port를 확인하십시오.



단축키 Ctrl + U 를 누르면 컴파일과 업로드가 같이 진행이 됩니다.

○ 시리얼 모니터 사용법

시리얼 모니터는 아두이노가 실행되는 단계를 관찰할 수 있는 도구 입니다.

주로 디버깅이나 결과값을 확인하기 위해서 사용을 합니다. 시리얼모니터를 사용하기 위해서는 Setup에서 Serial speed만 설정하시면 사용이 가능합니다.

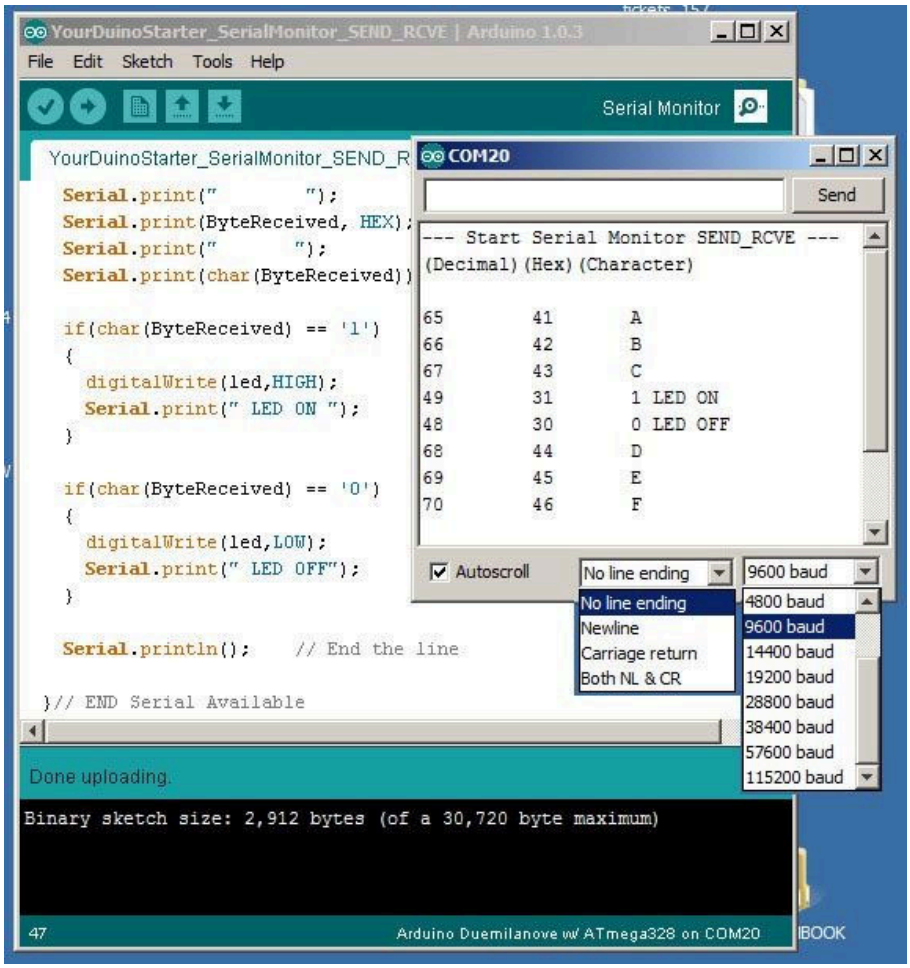
아두이노 우노에서는 Printf()는 기본적으로 지원하지 않고, Serial.print() 또는 Serial.println()을 지원합니다. Printf()를 사용하기 위해서는 별도의 class 소스 작업이 필요합니다.

[기본문법]

```
Serial.print(val);
Serial.print(val, format);
```

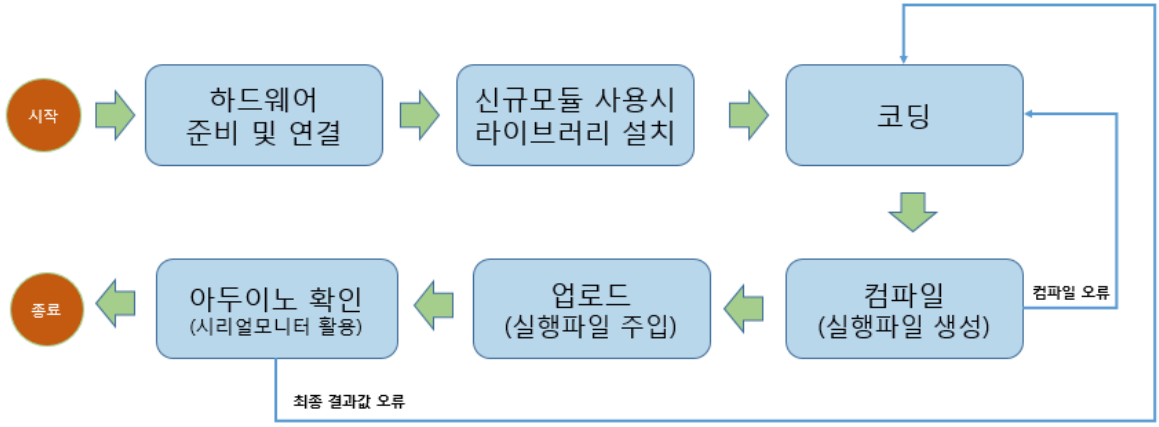
[사용예]

아두이노 IDE에서 툴->시리얼 모니터 실행(단축키 Ctrl+Shfit+M)



○ 아두이노 코딩절차

지금까지 설명 드린 절차를 요약해서 코딩과정을 순서도로 표현하면 아래와 같습니다.

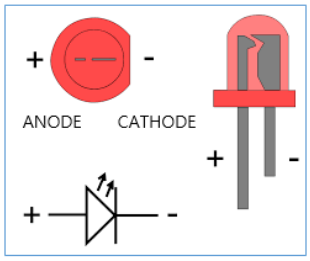


□ 예제 학습하기

○ LED blink

- 개요

LED 이론과 아두이노 우노의 디지털 출력을 방법에 대해서 설명하겠습니다.





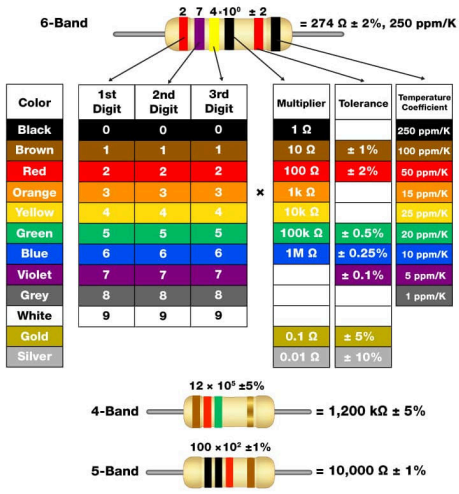
LED 란 전기에너지를 빛 에너지로 변환이 가능한 반도체 소자입니다.

+,- 양단에 순방향 전압을 입력하게 되면, 전기에너지가 빛 에너지로 변환되어 빛으로 방출이 소자입니다. 과거의 전기에너지를 열에너지로 전환하여 빛을 방출하는 전구와 다르게 직접 전기에너지를 빛 에너지로 변환하는 물질적인 성질 때문에 높은 효율의 빛을 표현할 수 있고 반영구적으로 사용이 가능합니다.

저항이란 전류의 흐름을 제어하고 전압을 조정하는 수동소자입니다. 아두이노에서 주로 사용하는 탄소피막/카본저항은 표면에 저항 값을 표시하는 색띠가 도장되어있어 저항 값을 육안으로 확인이 가능합니다.

다음의 표는 막대저항의 색상을 육안으로 확인해서 저항 값을 확인할 수 있는 표입니다.

만약 저항 없이 LED 를 직접 아두이노에 연결하게 된다면, 과전류(많은 전류)가 발생해서 LED 가 손상될 수 있습니다. 그래서 일반적으로 LED 에는 220~330 옴 저항을 같이 연결해야 합니다.



[저항 읽는 법 - 출처 : https://www.arrow.com]

- 준비물

. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

. RED LED X 1

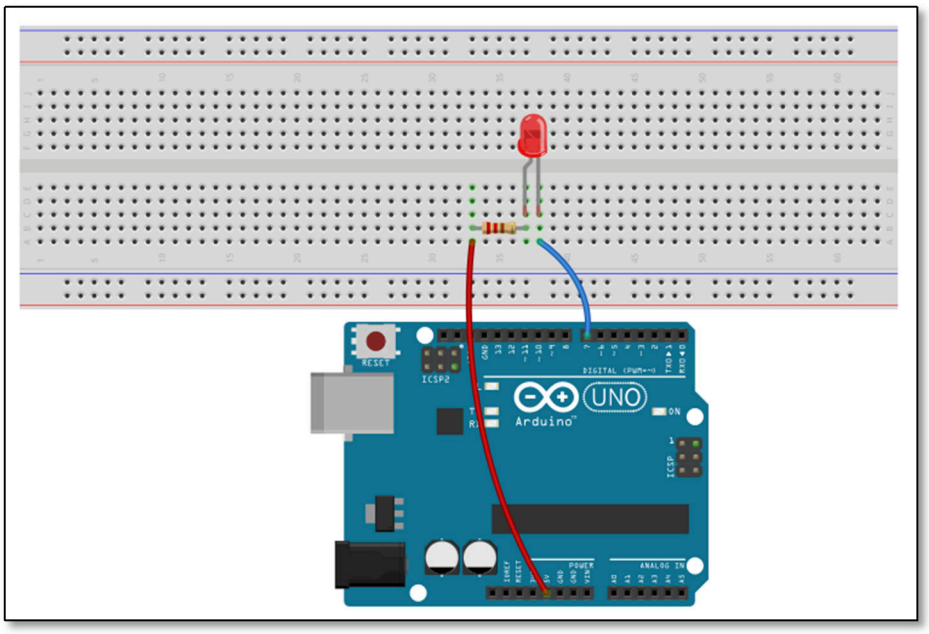
. 220 옴 저항 X 1

. 점퍼와이어

- 연결도

위에 말씀 드린 것 같이 LED 보호를 위해서 + 또는 – 단자에 220Ω 저항을 추가합니다. 옴의 법칙의 V=IR, I=V/R 이므로 저항 값이 높을 수록 전류가 낮아지기 때문에 LED 밝기가 어두워 집니다.

LED 의 다리가 긴 핀에 VCC(+)를 연결하시면 됩니다.



- **코딩**

다음 코드는 Setup()에서 출력포트를 설정하고 Loop 에서 1 초 단위로 LED 를 깜박입니다.

```
//*************************************************************************
// Project : LED Blink in Learning kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
#define LedPin  7
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LedPin, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LedPin, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LedPin, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

- **결과**

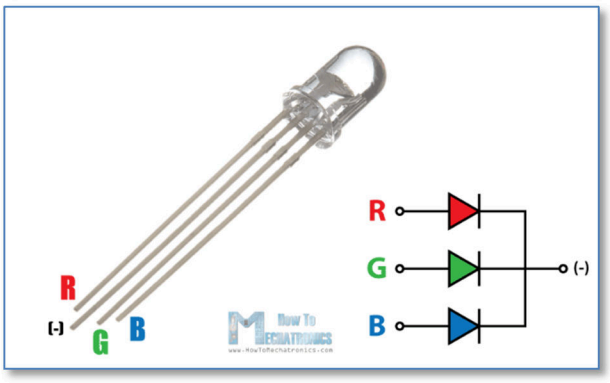
브래드보드에 연결된 빨강 LED 가 1 초 주기로 깜빡임을 확인할 수 있습니다.

○ RGB LED

- 개요

RGB LED 동작 방법에 대해서 설명하겠습니다.

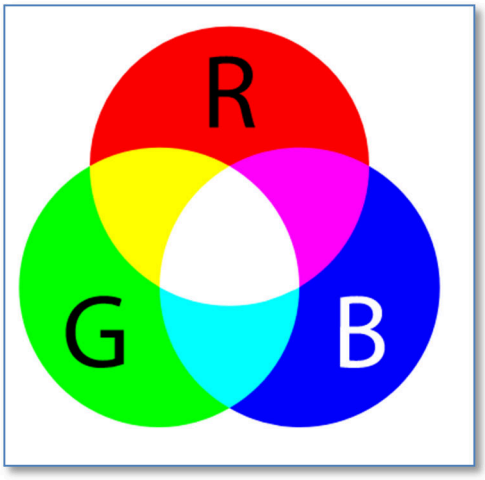
RGB 3 색 LED 는 빨강/파랑/녹색 LED 를 하나의 LED 로 합친 LED 입니다.





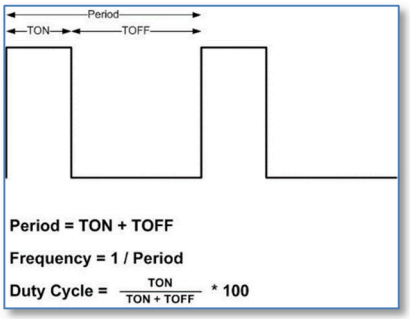
[출처 : https://howtomechatronics.com]

각 LED 를 ON/OFF 조합해서 7 가지 색상을 표시할 수 있습니다. 그리고 PWM 과 같은 밝기 조절을 사용하게 되면 빛의 3 요소를 구성하는 다양한 색으로 표시가 가능합니다.

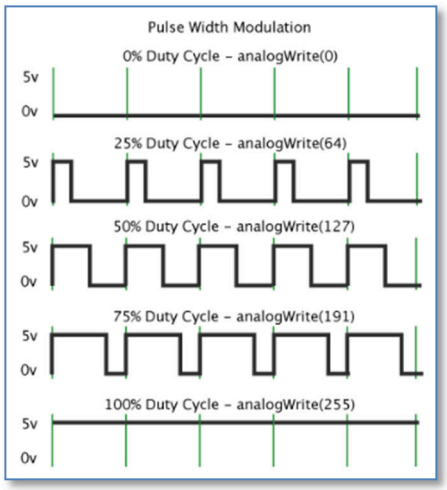


LED 밝기를 소프트웨어로 조절하기 위해서는 PWM 을 사용하면 됩니다.

PWM 은 Pulse Width Modulation 의 약자로서, Pulse 폭을 조정해서 출력되는 디지털 신호입니다. 전압이 균일한 교류 신호(구형파)를 생각해보면 Frequency 와 Duty 라는 두 개의 파라미터가 있습니다. Frequency 는 1 초 동안 발진되는 파형 개수이고, Duty 는 1 개의 주기에서 On 되는 시간을 비율을 계산된 수치입니다.



아래는 Duty 가 조정될 경우에 출력되는 신호의 그림입니다.



- 준비물

. 아두이노 우노(호환보드) X 1

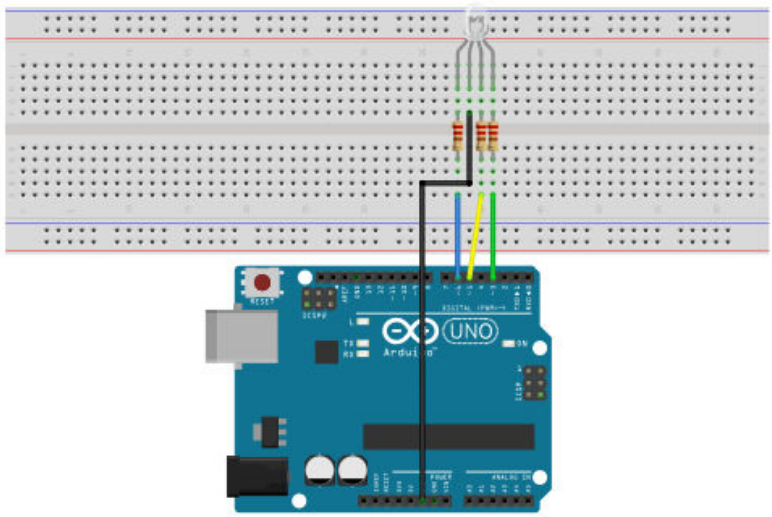
. 브래드보드 X 1

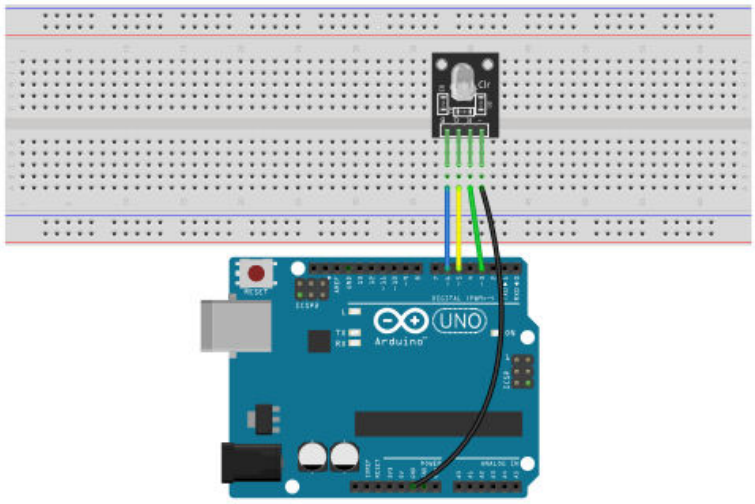
. RGB LED 모듈 X 1

. 220 옴 막대저항 X 3

. 점퍼와이어

- **연결도**





* RGB 모듈은 모듈 내부에 저항이 포함되어있어 추가 저항 연결이 필요 없습니다.

- **코딩**

소프트웨어 PWM 을 사용하여 다양한 색상이 표시할 수 있습니다.

```
//*********************************************************************************
// Project : RGB_LED Blink in Learning kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
// Define Pins
#define BLUE 3
#define GREEN 5
#define RED 6
void setup()
{
```

```
  pinMode(RED, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BLUE, OUTPUT);
  digitalWrite(RED, HIGH);
  digitalWrite(GREEN, LOW);
  digitalWrite(BLUE, LOW);
}
// main loop
void loop()
{
  int redValue;
  int greenValue;
  int blueValue;

  redValue = 255;
  greenValue = 0;
  blueValue = 0;

  for(int i = 0; i < 255; i += 1)
  {
    redValue -= 1;
    greenValue += 1;
    analogWrite(RED, redValue);
    analogWrite(GREEN, greenValue);
    delay(10);
  }

  redValue = 0;
  greenValue = 255;
  blueValue = 0;
  // Green, blue fade
  for(int i = 0; i < 255; i += 1){

    greenValue -= 1;
    blueValue += 1;
    analogWrite(GREEN, greenValue);
    analogWrite(BLUE, blueValue);
    delay(10);
  }

  redValue = 0;
  greenValue = 0;
  blueValue = 255;

  for(int i = 0; i < 255; i += 1){
    blueValue -= 1;
    redValue += 1;
    analogWrite(BLUE, blueValue);
    analogWrite(RED, redValue);
    delay(10);
  }
}
```

- **결과**

브래드보드에 연결되어 있는 LED 에서 다양한 색상이 표시되는 것을 확인 할 수 있습니다..

○ 버튼입력

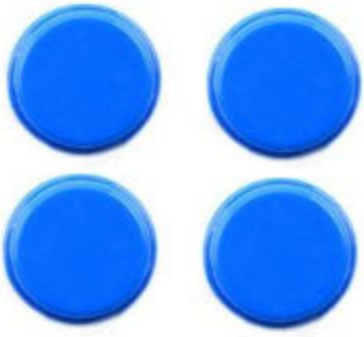
- 개요

아두이노 우노의 기본 디지털 입력(버튼)을 방법에 대해서 설명하겠습니다.

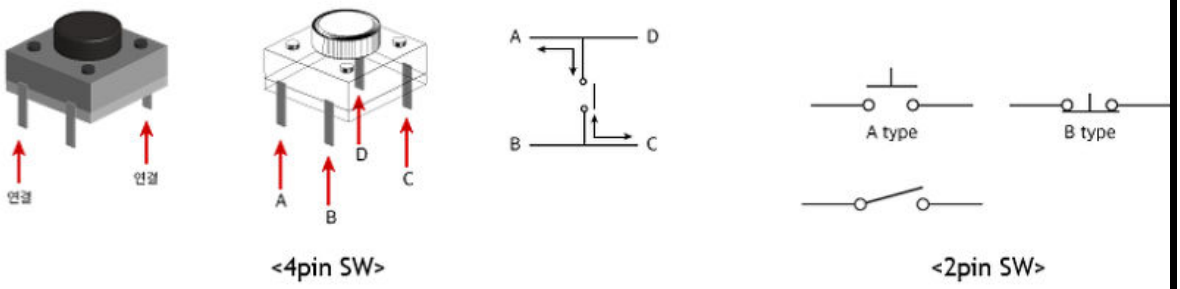
택트스위치는 내부 금속판이 서로 떨어진 상태를 유지하다가 스위치를 누르면 서로 연결되는 스위치를 말합니다.



[버튼사진]



[버튼 캡]



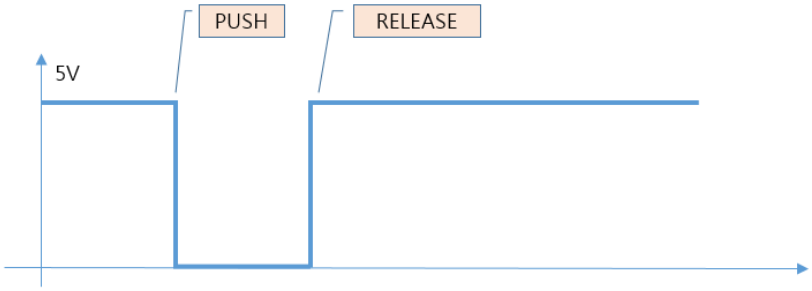
<4pin SW>

<2pin SW>

[4pin 과 2pin 스위치 원리]

구성된 회로에 따라서 버튼을 누르게 되면 아두이노로 입력되는 디지털 신호가 High->Low 또는 Low->High 로 변하게 됩니다.

다음그림은 PULL-UP 으로 구성된 버튼을 누를(PUSH) 경우의 신호 파형의 변화 입니다.



- **준비물**

. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

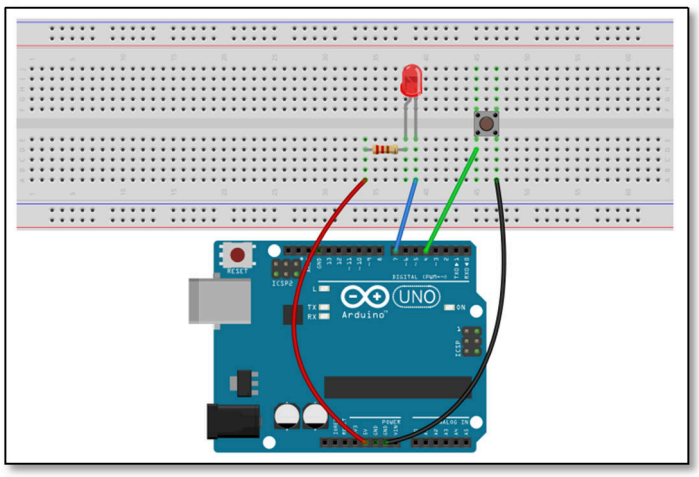
. RED LED X 1

. 220 옴 저항 X 1

. 버튼 스위치 X 1

. 점퍼와이어

- **연결도**



- **코딩**

버튼을 인식해서 LED ON/OFF 를 동작을 반복합니다.

```
//*************************************************************************
// Project : BUTTON  Blink in Learning kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
// Define Pins
#define RED 7
#define BUTTON 4
void setup()
{
  pinMode(RED, OUTPUT);
  pinMode(BUTTON, INPUT_PULLUP);
}
void loop()
{
  if ( digitalRead(BUTTON) ) {
    digitalWrite(RED, HIGH); // LED OFF
  }else{
    digitalWrite(RED, LOW);  // LED ON
  }

}
```

- **결과**

버튼을 누를 때 LED 가 ON 되는 것을 확인하실 수 있습니다.

○ 회로 내장 부저

- 개요



[수동부저]



[능동부저]

부저(Buzzer)는 전기신호를 소리로 출력하는 신호 알림 장치 입니다. 원리는 부저 내부의 금속판에 특정 주파수를 입력하여 진동 유발하여 소리를 발생시킵니다. 아두이노에서 많이 사용되는 부저 종류는 회로가 내장되어있는 능동부저(Active Buzzer)와 외부에서 주파수 전기신호를 입력 해주어야 하는 수동부저(Passive Buzzer)가 있습니다.

능동부저는 내부의 회로가 포함되어 있어 전원이 입력되면 특정 주파수 소리가 출력되는 구조입니다. 그래서 소리 음색은 변함 없이 동일한 소리로 출력하도록 되어있습니다.

반면, 수동부저는 외부에서 신호를 입력해야 하므로 여러 가지 음색으로 출력이 가능합니다. 아두이노 기본 라이브러리인 Tone() 함수를 사용하여 다양한 음색으로 출력을 바꿀 수 있습니다.

Tone()함수는 PWM 의 출력 주파수 변경해주는 함수로 torn(포트,주파수,시간); 설정할 수 있습니다.

- 준비물

. 아두이노 우노(호환보드) X 1

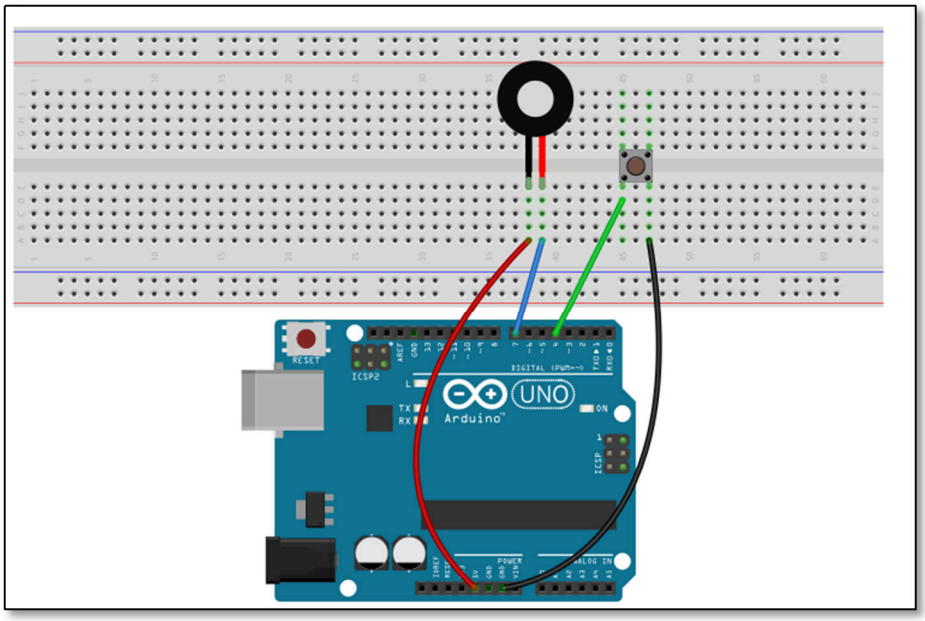
. 브래드보드 X 1

. 회로내장 부저 X 1

. 버튼스위치 X 1

. 점퍼와이어

- **연결도**



- **코딩**

버튼을 인식해서 BEEP 음을 발생시킵니다.

```
//*************************************************************************
// Project : Buzzer in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
// Define Pins
#define BUZZER 7
#define BUTTON 4
void setup()
{
  pinMode(BUZZER, OUTPUT);
  pinMode(BUTTON, INPUT_PULLUP);
}
void loop()
{
  if ( digitalRead(BUTTON) ) {
    digitalWrite(BUZZER, HIGH); // BUZZER OFF
  }else{
    digitalWrite(BUZZER, LOW);  // BUZZER ON
  }

}
```

- **결과**

버튼을 누를 때 동일한 음색의 소리를 들을 수 있습니다.

○ 무회로 부저

- 개요

이전에 설명 드린 것처럼 무회로 부저는 외부에서 금속판을 진동하기 위한 발진 신호를 입력해 주어야 합니다. 아두이노에서 사용하는 torn()함수가 PWM 을 발생시켜는 **금속판을 진동시키는** 기능을 하고 있습니다.

도 레 미 파 솔 라 시 도의 주파수는 도(523Hz), 레(587Hz), 미(659 Hz), 파(698 Hz), 솔(784), 라(880 Hz), 시(988 Hz), 도(1047 Hz) 입니다.

바로 코딩해서 소리로 해보도록 하겠습니다.

- 준비물

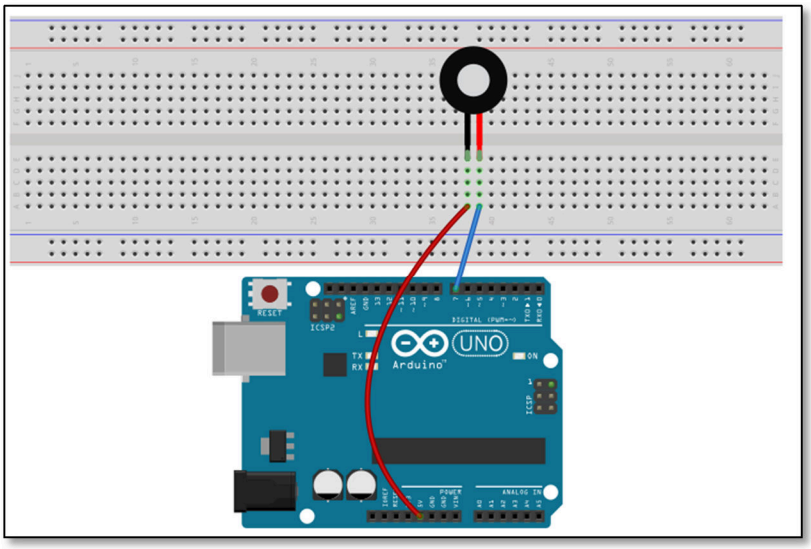
. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

. 무회로 부저 X 1

. 점퍼와이어

- 연결도



- **코딩**

도 레 미 파 솔 라 시 도를 반복해서 출력하는 코드입니다.

```
//*************************************************************************
// Project : Buzzer in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
// Define Pins
#define BUZZER 7
#define BUTTON 4
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
int melody[] = {
  NOTE_C5, NOTE_D5, NOTE_E5, NOTE_F5, NOTE_G5, NOTE_A5, NOTE_B5, NOTE_C6};
int duration = 500;  // 500 miliseconds

void setup() {

}

void loop() {
  // 도레미파솔라시도
  for (int thisNote = 0; thisNote < 8; thisNote++) {
    tone(BUZZER, melody[thisNote], duration);
    delay(1000);
  }

  delay(1000);
}
```

- **결과**

도 레 미 파 솔 라 시 도 음을 반복적으로 들으실 수 있습니다.

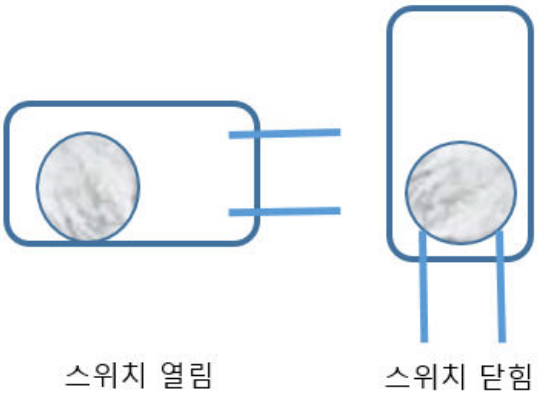
○ 틸트스위치

- 개요



틸트스위치(tilt switch)는 기울기스위치 또는 볼 스위치 라고도 합니다. 내부구조는 내부에 원형 쇠구슬 또는 수은이 중력에 따라 이동하면서 스위치를 열고 닫는 방식입니다. 아래그림과 같이 스위치가 수평을 유지할 경우 스위치가 열리는 형태이고, 수직을 유지할 경우 스위치가 닫히는 형태입니다.

스위치를 흔들면 내부의 금속 움직임을 느낄 수 있습니다.





- 준비물

. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

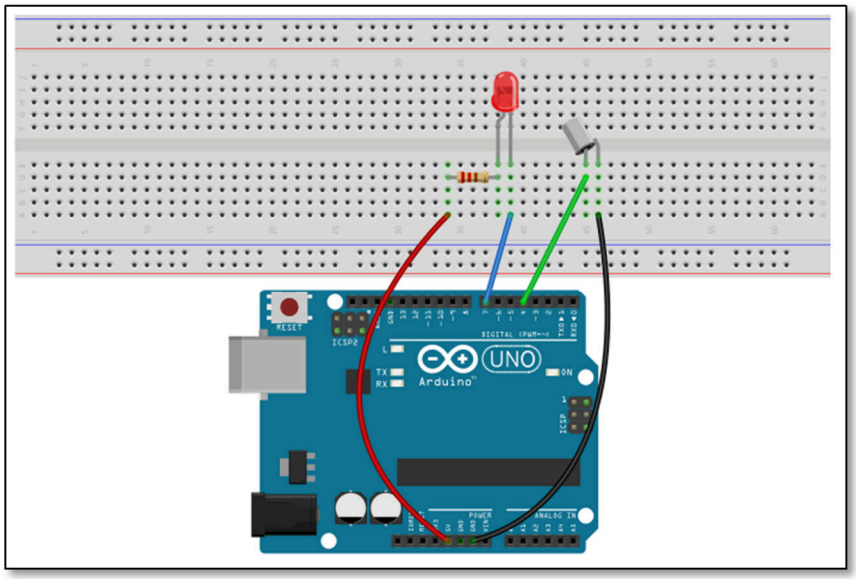
. 틸드스위치 X 1

. 빨강 LED X 1

. 220 옴 저항 X 1

. 점퍼와이어

- **연결도**



- **코딩**

틸트스위치를 인식해서 LED 를 제어하는 코드입니다.

```
//*********************************************************************************
// Project : 06_BALL_SW in Learning kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
// Define Pins
#define RED 7
#define BALL_SW 4
void setup()
{
  pinMode(RED, OUTPUT);
  pinMode(BALL_SW, INPUT_PULLUP);
}
void loop()
{
  if ( digitalRead(BALL_SW) ) {
    digitalWrite(RED, LOW);  // LED ON
  }else{
    digitalWrite(RED, HIGH); // LED OFF
  }

}
```

- **결과**

틸트스위치가 세워져 있다면 LED 가 꺼져있고, 틸트스위치를 기울이게 되면 LED 가 켜지는 것을 확인할 수 있습니다.

○ 서보모터

- **개요**

서보모터(Servo) 대해서 알아 보도록 하겠습니다.



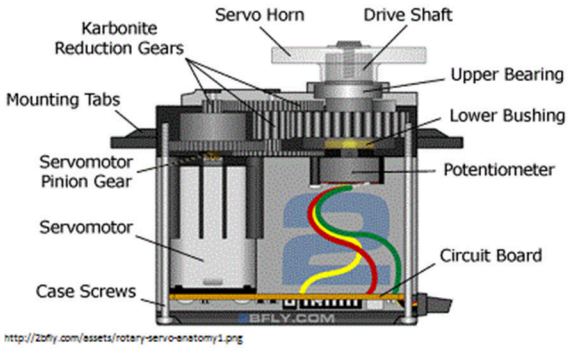
우선 모터를 정의해보면 모터란 전기에너지를 회전운동이 가능한 물리적인 운동 에너지로 변환하는 기계입니다.

서보모터의 어원은 추종하다 따르다 라는 용어에서 시작되었습니다.

다시 말해, 컨트롤러의 명령에 따라 특정한 위치로 회전/이동이 가능한 모터입니다. 서보모터의 내부구조 물리적인 회전이 가능한 모터와 위치를 모니터링 할수 있는 보드가 결합된 형태 이루어져 있습니다.

구조에 대해서 자세한 설명을 드리면,

아래 그림은 서보모터의 단면도입니다.
사실 이 그림 한 장으로 서보모터의 모든 것이 설명이 가능해 보입니다. .



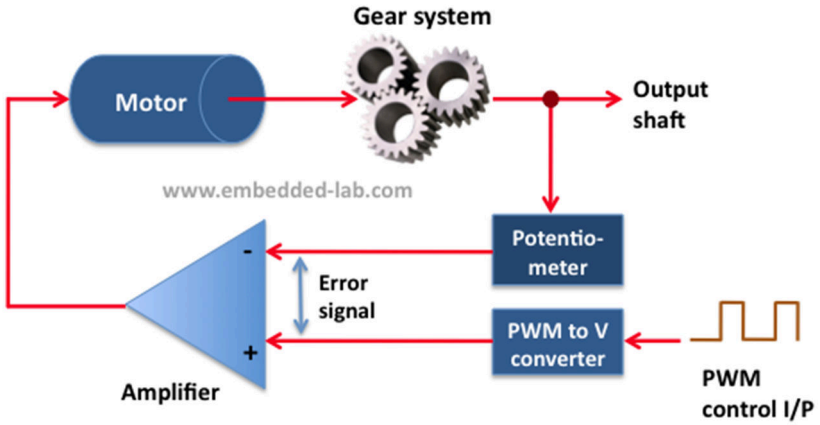
[출처 : http://www.arxterra.com/spring-2016-velociraptor-servos/]

구동 순서를 단계별로 설명하면,

STEP 1 : 그림 하단 오른쪽으로 전원 및 위치신호가 입력되면 Circuit 보드는 왼쪽 서보모터를 구동합니다.
STEP 2 : 서보모터가 회전하게 되면 기어박스로 회전이 전달되고 최종 샤프트가 회전합니다.
STEP 3 : 샤프트 하단에 위치한 포텐션미터라는 가변저항으로 현재 위치값을 센싱해서 다시 Circuit 보드로 전달을 합니다.
STEP 4: 현재 회전 위치 값을 전달받은 Circuit 보드는 입력 위치 비교해서 서보모터 출력을 다시 조정합니다.
STEP 5: 현재 위치와 입력 받은 위치가 동일하다면 서보모터구동은 정지합니다. .

위에 순서는 전원입력을 시작으로 작성되었지만,
실제 순서는 포텐션미터로 현재 위치를 감지하고 서보모터 구동을 한다고 생각하시면 됩니다.

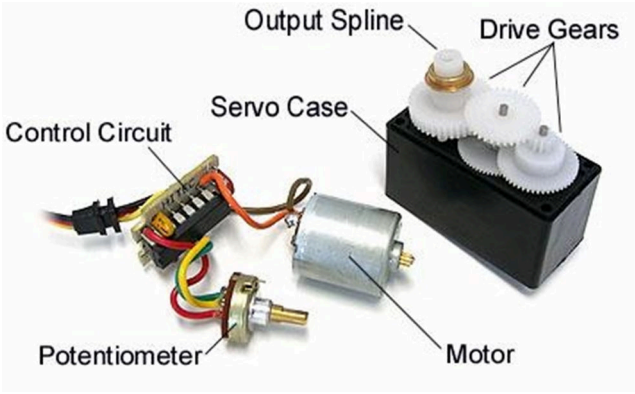
위에서 설명드린 구동순서는 아래 그림 처럼 간단히 요약이 가능합니다.
즉, 현재 모터의 위치를 피드백 받아 출력 위치를 다시 제어하는 메커니즘입니다.



[출처 : http://embedded-lab.com/blog/lab-21-servo-motor-control/]

다음 그림은 일반적인 서보모터 분해도입니다.

위에서 설명 드린 구동부/제어부 되어있는 것을 확인할 수 있습니다.



[출처 : http://embedded-lab.com/blog/wp-content/uploads/2012/04/servo-parts.jpg]

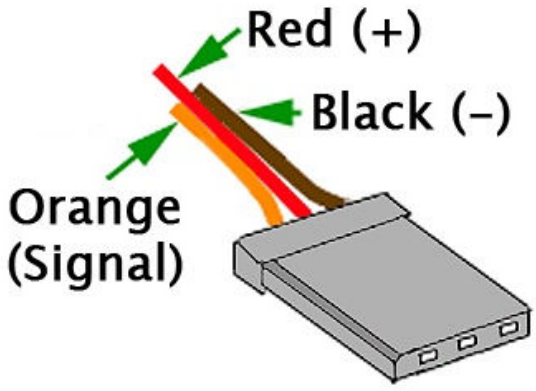
일반적인 서보모터의 회전각도는 0~90 도 또는 0~180 도 까지만 동작이 가능합니다. 그리고 위치를 제어하기 위해서는 PWM 의 Duty 를 사용합니다.(PWM Duty 는 RGB LED 챕터에서 참고)

- 준비물

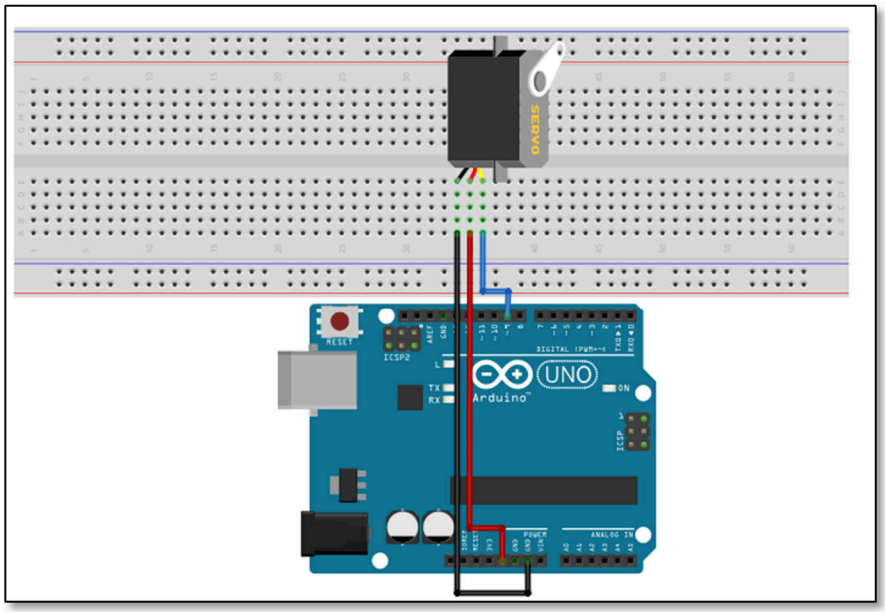
. 아두이노 우노(호환보드) X 1

. 서보모터 X 1

- 연결도



SG90 서보모터 핀맵



- **코딩**

Servo.h 라이브러리 설치가 필요합니다.
라이브러리에서 PWM 을 발생시켜 서보 모터를 회전을 하는 원리입니다.
총 180 도 각도 범위를 1 도 단위로 좌우 회전하는 코드입니다.

```
//*****************************************************************************
// Project : 07_SERVO_MOTOR in Learning kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
#include <Servo.h>
#define SERVOPIN 9
Servo myservo;  // servo object 생성
int pos = 0;    // 각도
void setup() {
  myservo.attach(SERVOPIN);  // attaches to the servo object
}
void loop() {
  for (pos = 0; pos <= 180; pos += 1) {
    // 1 도씩 이동
    myservo.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    myservo.write(pos);
    delay(15);
  }
}
```

- **결과**

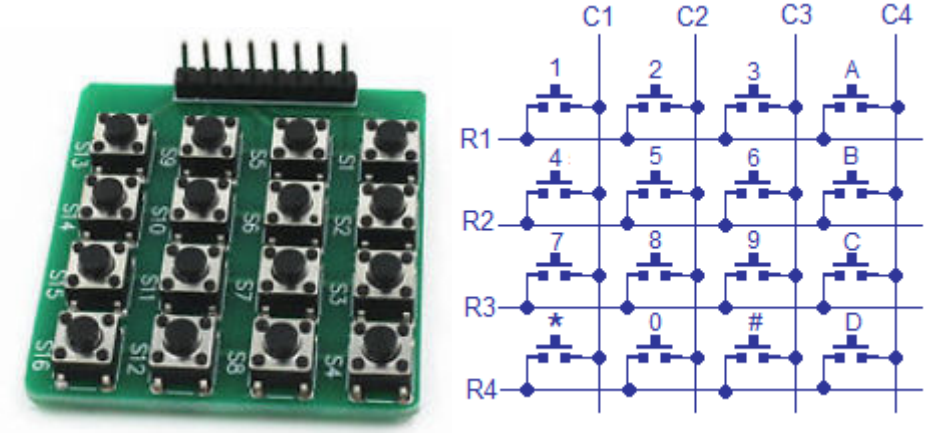
서보모터가 1 도 단위로 180 도 좌우 회전하는 것을 확인할 수 있습니다.

○ 4x4 키패드

- 개요

4x4 키패드 모듈에 대해서 알아 보겠습니다.

8 개의 디지털 출력, 입력을 사용해서 총 16 개의 버튼 입력을 사용할 수 있습니다. 키패드모듈의 모양과 내부 회로도는 아래와 같습니다.



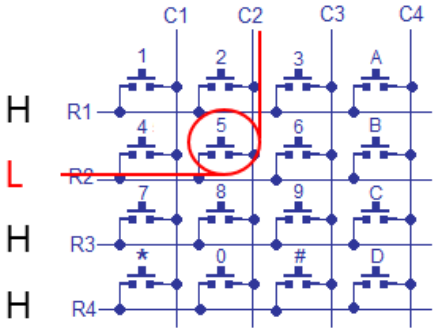
내부는 matrix 형태로 Row 열과 Column 이 스위치로 연결이 되어있습니다.

Row 라인을 출력으로 선택하고 Column 라인을 입력으로 설정을 하고 출력 포트를 순차적으로 Active Low 를 주고 입력라인을 체크는 방식을 사용합니다.

5 번을 인식하는 과정을 그림으로 표시하면 아래와 같습니다.

1) R1,R3,R4 는 High 를 유지하고 R2 를 LOW 신호를 출력합니다.

2) C1,C2,C3,C4 를 체크하여 C2 가 Low 인식이 되면 5 버튼이 눌렸다고 판단을 합니다.



일반적으로 사용하는 컴퓨터 키보드도 동일한 알고리즘으로 동작을 합니다. 다만 키 버튼이 많으므로 8x16 을 사용하는 것과 중복 키가 눌리는 현상인 Ghost key 처리하는 알고리즘이 추가되었을 뿐입니다.

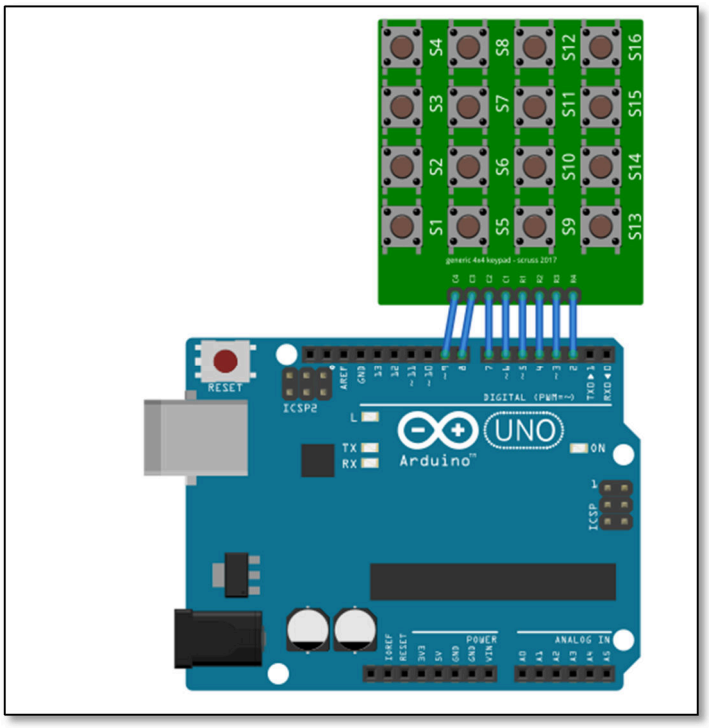
- 준비물

. 아두이노 우노(호환보드) X 1

. Keypad X 1

. 점퍼와이어

- 연결도



- 코딩

버튼을 Scan 해서 눌린 버튼을 인식합니다.

```
//************************************************************************
// Project : 08_KEYPAD in Learning kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************
```

```
#define Rows  4
#define Cols  4
char keys[Rows][Cols] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

const int RowPinTbl[Rows] = {5, 4, 3, 2};
const int ColPinTbl[Cols] = {6, 7, 8, 9};

void setup() {
    Serial.begin(9600);    // init Serial

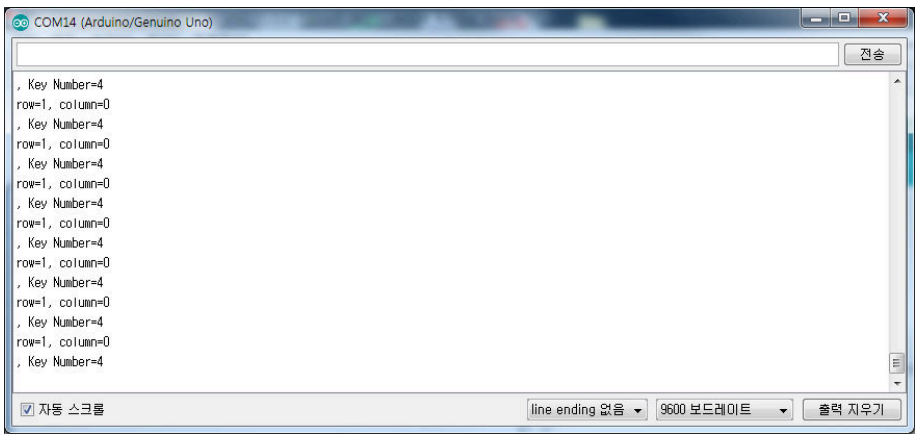
    for(int i=0; i<Rows; i++) {
        pinMode(RowPinTbl[i], INPUT_PULLUP);
    }

    for(int j=0; j<Cols; j++) {
        pinMode(ColPinTbl[j], OUTPUT);
        digitalWrite(ColPinTbl[j], HIGH);    // 초기값 HIGH
    }
}
void loop() {
    // 버튼 인식
    for(int j=0; j<Cols; j++) {
        digitalWrite(ColPinTbl[j], LOW);    // Col 라인 Active Low
        for(int i=0; i<Rows; i++) {
            if( digitalRead(RowPinTbl[i]) == LOW ) {    // check Row 라인
                Serial.print("row=");
                Serial.print(i);
                Serial.print(", column=");
                Serial.println(j);
                Serial.print(", Key Number=");
                Serial.println(keys[i][j]);

            }
        }
        digitalWrite(ColPinTbl[j], HIGH);    // Col 라인 다시 High
    }
}
```

- **결과**

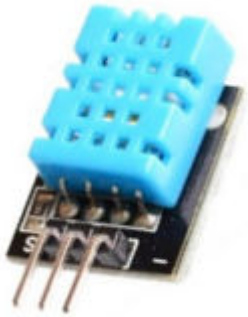
버튼이 인식된 Row, Col 그리고 매칭된 키 값을 확인할 수 있습니다.



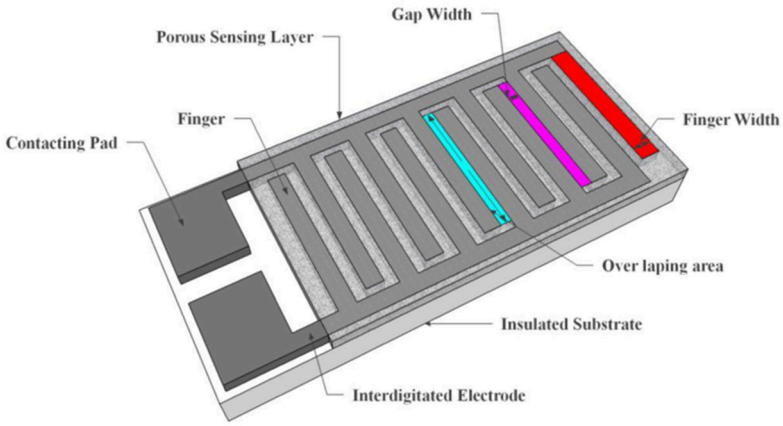
○ DHT11 온습도센서

- 개요

DHT11 온습도 센서에 대해서 알아 보겠습니다.

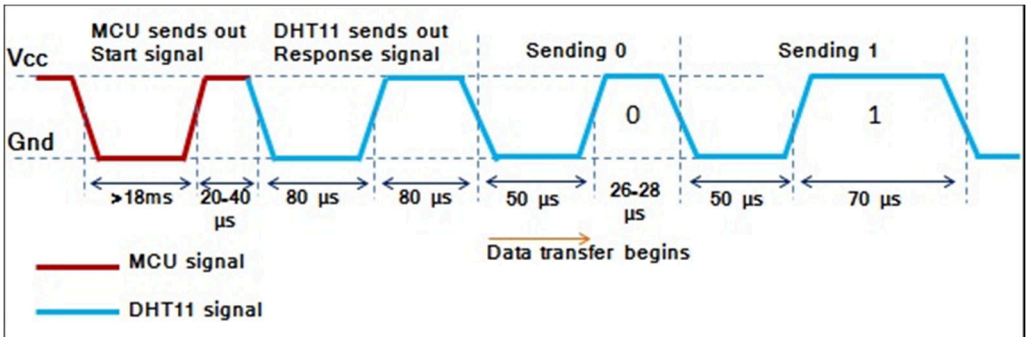


DHT11 온습도센서는 온도와 습도의 데이터를 디지털로 출력해주는 센서입니다. 내부 온도센서는 서미스터의 저항으로 온도에 따라 저항 값이 변화하는 센서를 사용합니다. 습도센서 역시 저항센서를 사용하는데 수위센서와 동작이 유사합니다. 빗살무늬 선들의 사이에 습도를 체크하여 변화된 저항 값을 체크 방식입니다.



[출처 : http://www.mdpi.com/1424-8220/14/5/7881/htm]

출력방식은 Single wire 방식으로 입력과 출력을 같이 사용 통신 프로토콜입니다. 초기 신호를 Active Low 로 18ms 이상 트리거를 할 경우 DHT11 에서는 온도와 습도 그리고 체크섬 값을 전송합니다.



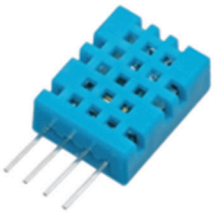
[출처 https://libstock.mikroe.com]

다음 표는 40bit 에 대한 Packet 구조입니다.

parameter	습 도		온 도		체크섬
총 40bit	8bit integral Data	8bit Decimal Data	8bit integral Data	8bit Decimal Data	Check Sum

DHT11 보다 정확성이 향상된 DHT22 가 있습니다.

아래 표는 두 개 모듈을 비교한 자료입니다.



VS



DHT11 ***DHT22***

0-50도 (오차 +- 2도)	온도 범위	-40-125도 (오차 +- 2도)
20~80%(오차 +- 5%)	습도 범위	0~100%(오차 +- 2~5%)
1Hz	샘플링 주기	0.5Hz
15.5 x 12 x 5.5 (mm)	크기	15.1 x 25 x 7.7 (mm)
3~5V	동작전압	3~5V
2.5mA	최대 소비전력	2.5mA

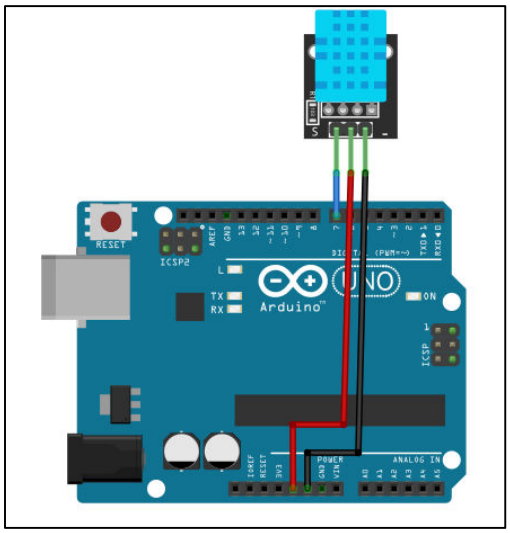
- 준비물

. 아두이노 우노(호환보드) X 1

. DHT11 X 1

. 점퍼와이어

- 연결도

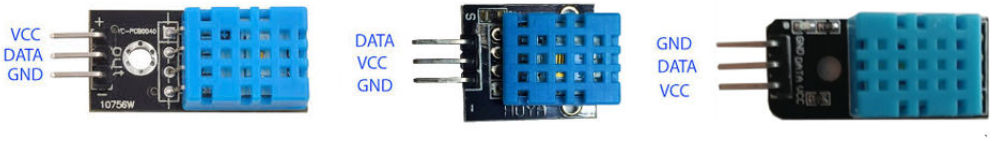


*참고사항

아래 그림과 같이 DHT11은 두 종류의 모듈이 있습니다.

연결도는 아래 모듈 중 오른쪽 모듈을 기준으로 작성되었습니다.

왼쪽 모듈이 포함되어있는 경우는 연결선을 변경해주시면 됩니다.



- 코딩

DHT 11 온습도값을 읽어서 시리얼 모니터로 표시합니다.

```
//************************************************************************************
// Project : DHT11 in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************************
int DHT11pin = 7;
unsigned char dat[5];
unsigned char read_DHT11 () {
  unsigned char data;
  for (int i = 0; i < 8; i ++) {
    if (digitalRead (DHT11pin) == LOW) {
      while (digitalRead (DHT11pin) == LOW); // wait for 50us
      delayMicroseconds (30);  // determine the duration of the high level to
determine the data is '0 'or '1'
      if (digitalRead (DHT11pin) == HIGH)
        data |= (1 << (7-i)); // high front and low in the post
      while (digitalRead (DHT11pin) == HIGH); // data '1 ', wait for the next one
receiver
     }
  }
return data;
}
```

```
void start_read () {
  digitalWrite (DHT11pin, LOW); // start signal
  delay (20);

  digitalWrite (DHT11pin, HIGH);
  delayMicroseconds (40);  // DHT11 response

  pinMode (DHT11pin, INPUT);
  while (digitalRead (DHT11pin) == HIGH);
  delayMicroseconds (80);
  if (digitalRead (DHT11pin) == LOW);
  delayMicroseconds (80);

  for (int i = 0; i < 4; i ++) // receive temperature and humidity data
    dat[i] = read_DHT11 ();

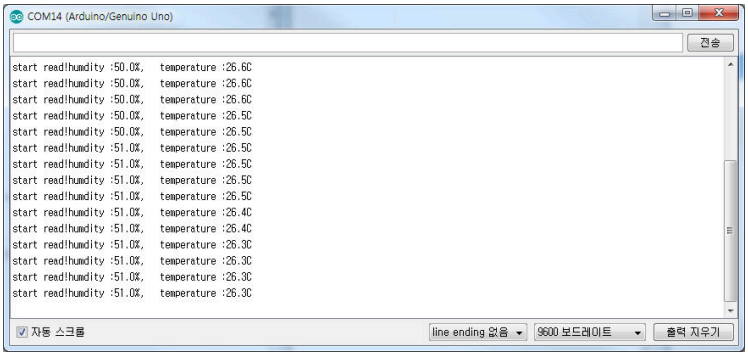
  pinMode (DHT11pin, OUTPUT);
  digitalWrite (DHT11pin, HIGH); // Stop signal
}

void setup () {
  Serial.begin (9600);
  pinMode (DHT11pin, OUTPUT);
}

void loop () {
  Serial.print ("start read!");
  start_read ();
  Serial.print ("humdity :");
  Serial.print (dat [0], DEC); // display  humidity-bit;
  Serial.print ('.');
  Serial.print (dat [1], DEC); // display humidity decimal ;
  Serial.print ("%,   ");
  Serial.print ("temperature :");
  Serial.print (dat [2], DEC); // display  temperature-bits;
  Serial.print ('.');
  Serial.print (dat [3], DEC); // display  temperature  decimal ;
  Serial.println ('C');
  delay (2000);
}
```

- 결과

습도와 온도값을 시리얼모니터를 통해서 확인할 수 있습니다.



○ 조이스틱

- 개요

조이스틱에 대해서 알아 보겠습니다.



PIN	이름	기 능
1	SW	버튼(디지털출력)
2	VRy	Y축(아날로그 출력)
3	VRx	X축(아날로그 출력)
4	5V	VCC
5	GND	GND

게임기에서도 주로 사용하는 조이스틱 스위치입니다. X 축과 Y 축의 변화를 미세하게 감지할 수 있으며, 푸쉬 버튼 기능이 포함되어 있습니다.

출력 신호를 살펴보면 X, Y 의 이동은 아날로그 신호로 출력이 되고, 푸쉬 버튼 기능은 디지털신호로 출력이 됩니다.

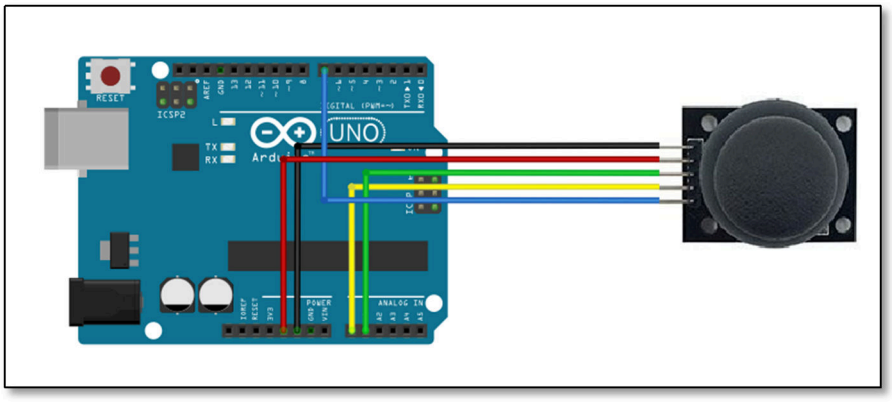
- 준비물

. 아두이노 우노(호환보드) X 1

. 조이스틱 X 1

. 점퍼와이어

- 연결도



- 코딩

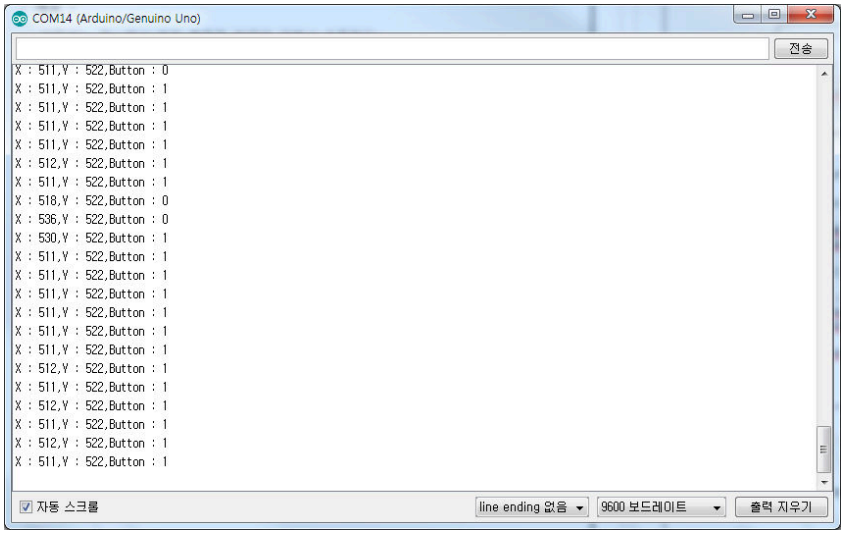
XY 는 아날로그, 스위치는 디지털 입력으로 설정하고 100ms 마다 인식합니다.

```
//*********************************************************************************
// Project : joystick  in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int JoyStick_X = A0; // x-axis
int JoyStick_Y = A1; // y-axis
int JoyStick_Z = 7;  // key-axis
void setup ()
{
  Serial.begin (9600); // speed 9600
  pinMode (JoyStick_X, INPUT);
  pinMode (JoyStick_Y, INPUT);
  pinMode (JoyStick_Z, INPUT_PULLUP);
}
void loop ()
{
  int x, y, z;
  x = analogRead (JoyStick_X);
  y = analogRead (JoyStick_Y);
  z = digitalRead (JoyStick_Z);

  Serial.print ("X : ");
  Serial.print (x, DEC);
  Serial.print (",");
  Serial.print ("Y : ");
  Serial.print (y, DEC);
  Serial.print (",");
  Serial.print ("Button : ");
  Serial.println (z, DEC);
  delay (100);
}
```

- 결과

X,Y 의 출력 범위는 0~1024 이고 버튼은 0 과 1 입니다.



○ IR 적외선 수신

- 개요

IR 센서 (infrared sensor)에 대해서 알아 보겠습니다.

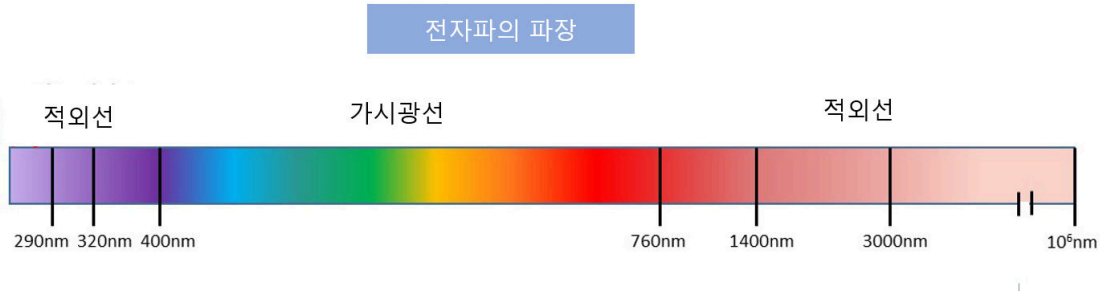


[적외선 수신 센서]

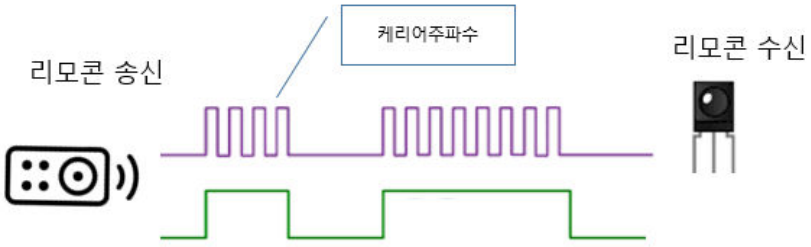


[IR 리모콘]

IR 이란 적외선으로 우리가 육안으로 볼 수 없는 빛의 파장으로 우리 눈으로 볼 수 있는 가시광선 보다 파장이 0.74~1000um 으로 깁니다.



적외선을 이용한 활용 범위는 가정에서 사용하는 리모콘, 온도감지센서, 적외선 터치센서 등 다양합니다.



일상 생활에서 IR 를 가장 많이 사용하는 기계장치는 TV, 에어컨을 조정하는 리모콘입니다.

- **준비물**

. 아두이노 우노(호환보드) X 1

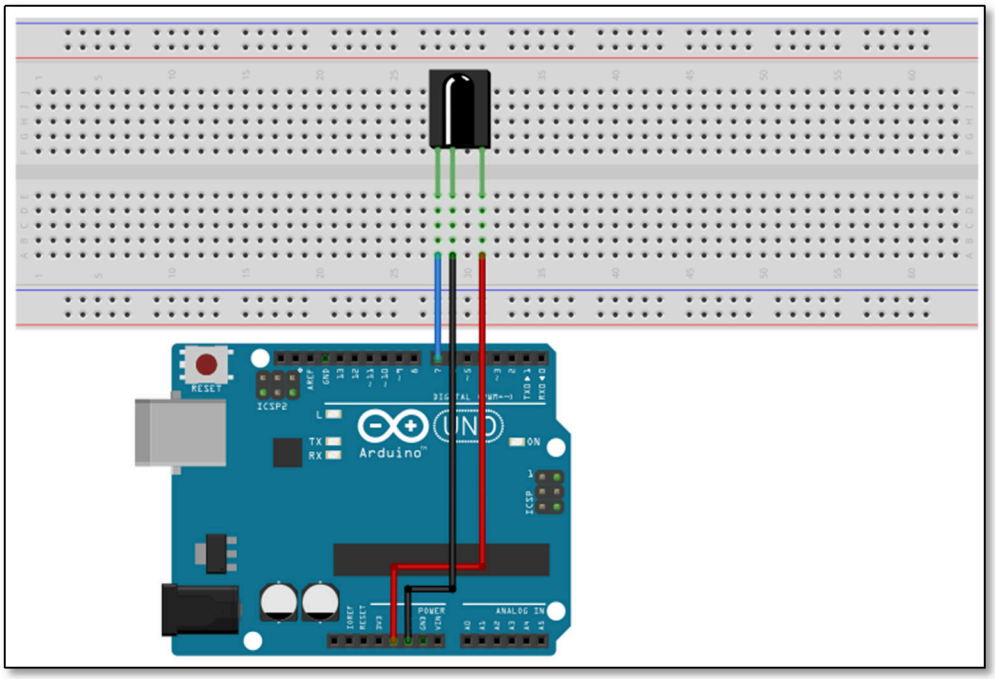
. 브래드보드 X 1

. IR 수신센서 X 1

. IR 리모콘 X 1

. 점퍼와이어

- **연결도**



- **코딩**

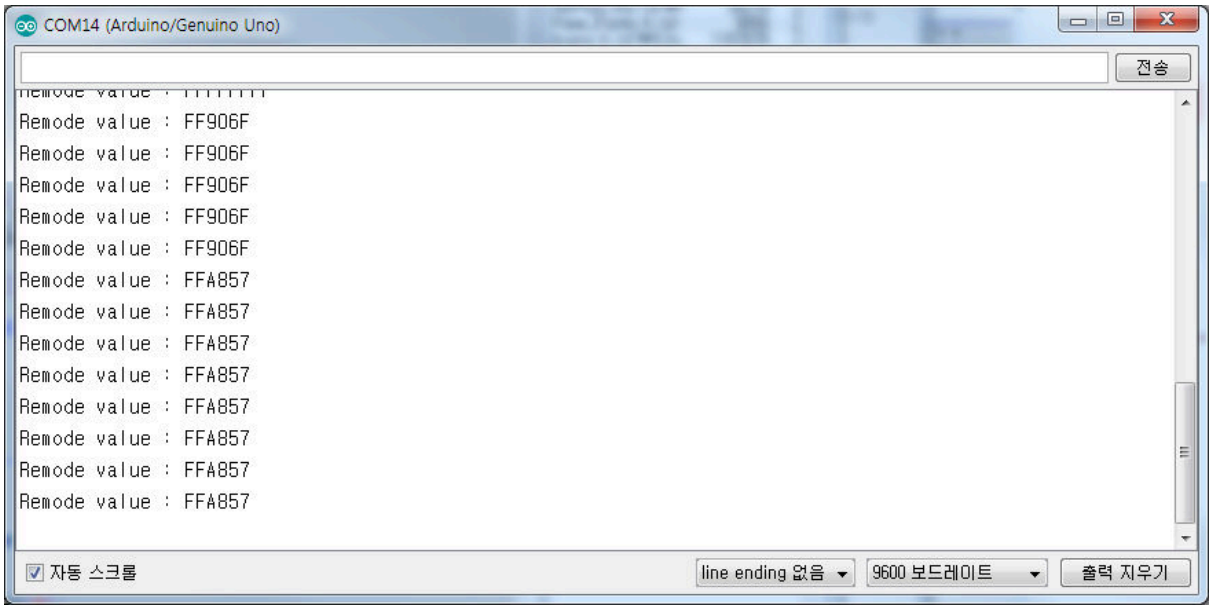
https://github.com/shirriff/Arduino-IRremote/ 라이브러리 설치가 필요합니다.

```
//********************************************************************************
// Project : IR receiver  in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//********************************************************************************
#include <IRremote.h>
int RECV_PIN=7;
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup() {
  Serial.begin(9600);
  irrecv.enableIRIn();
}
void loop() {
  if(irrecv.decode(&results)){
    Serial.print("Remode value : ");
    Serial.println(results.value, HEX);
    irrecv.resume();
  }
  delay(100);
}
```

- **결과**

수신 받는 IR 데이터를 확인할 수 있습니다.

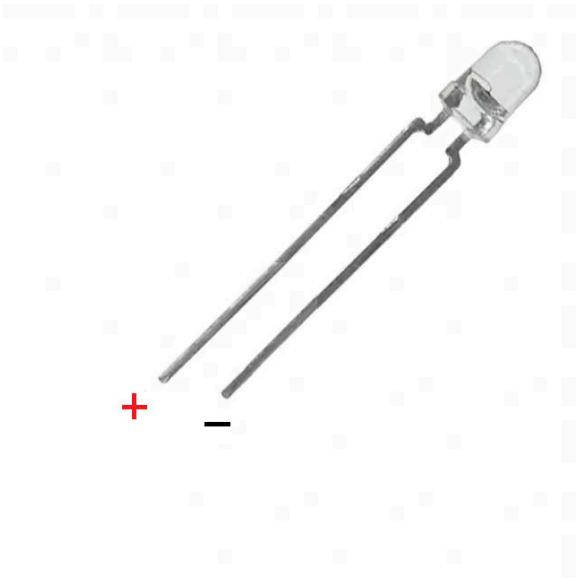
대부분의 리모콘은 키값은 처음 누를 때와 누르고 있을 때(반복) 값이 다릅니다.



○ IR 적외선 송신

- 개요

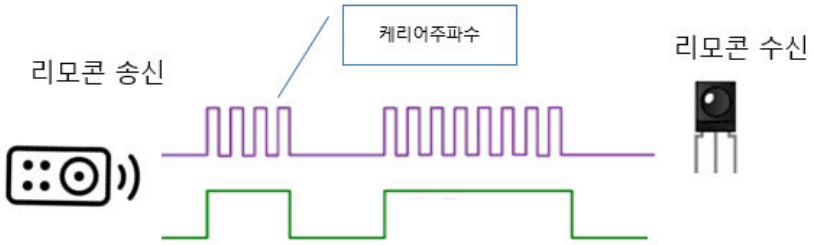
적외선 LED (infrared LED)에 대해서 알아 보겠습니다.



[적외선 LED 센서]

이전 장에서 설명드린 적외선 리모콘에서 사용되는 IR LED 입니다.

아래 그림에서 캐리어 주파수라는 것은 태양광 또는 형광등에서 발산되는 적외선 노이즈를 제거하기 위해서 사용되는 주파수 있습니다. 일반 가전제품 회사마다 캐리어주파수를 다르게 사용합니다. 소니, 삼성, 필립스(38Khz, 40Khz, 56Khz 등) 캐리어 주파수가 다를 경우에는 수신 감도가 떨어질 수 있지만 수신은 가능합니다.



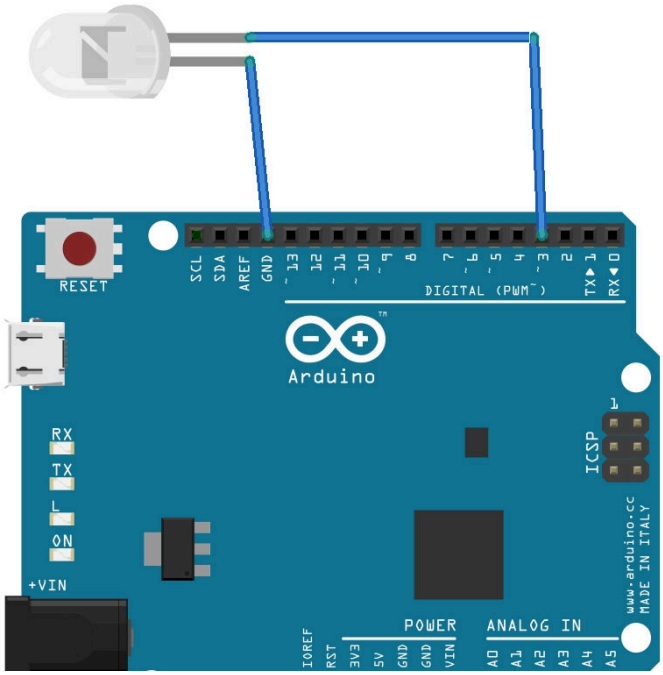
- **준비물**

. 아두이노 우노(호환보드) X 1

. IR LED X 1

. 점퍼와이어

- **연결도**



- **코딩**

https://github.com/shirriff/Arduino-IRremote/ 라이브러리 설치가 필요합니다.

```
//*****************************************************************************
// Project : IR_LED in Learning kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
#include <IRremote.h>
IRsend irsend;
void setup ()
{
  Serial.begin (9600);
}
void loop () {
  for (int i = 0; i <50 ; i++) {
    irsend.sendSony(0xa90, 12); // Sony TV power code
    delay (40);
  }
}
```

- **결과**

이전 장에서 구현 해보았던 IR 수신센서로 수신을 확인할 경우 반복적으로 0xA90 값이 수신되는 것을 확인할 수 있습니다.

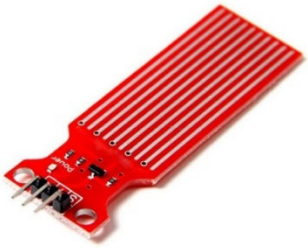
만약 가정에서 사용하시는 리모콘을 아두이노로 만드**시려면, 각 회사별/제품별 IR 프로토콜을 알아야 합니다. 인터넷을 찾아 보시면** 많은 자료를 확인할 수 있습니다.

Ex) 소니 DVR : https://www.sbprojects.net/knowledge/ir/sirc.php

○ 수위센서

- 개요

수위센서/ 물센서 (Water sensor)에 대해서 알아 보겠습니다.



물센서는 물을 감지센서입니다. 넓은 인식판 위에 빗살 무니 동판 무늬 센서가 위치하고 있습니다. 원리는 빗살무늬에 물방물이 형성이 되면 서로 도체가 형성되어 미세한 전류가 흐리게 되어있습니다.

물이 인식되게 되면 아날로그신호의 변화가 출력이 됩니다. 물이 인식되지 않는다면 아날로그 값은 0 이 출력됩니다.

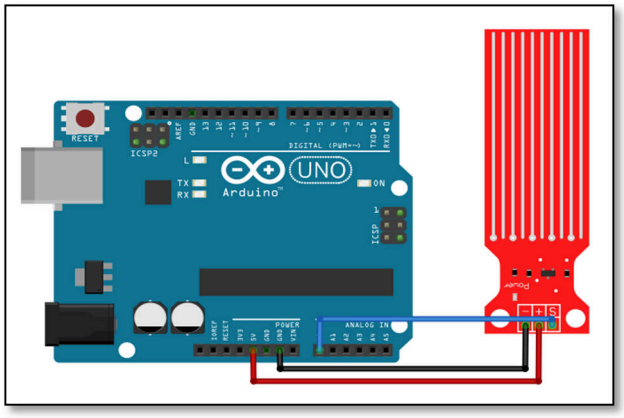
- 준비물

. 아두이노 우노(호환보드) X 1

. 물센서 X 1

. 점퍼와이어

- 연결도



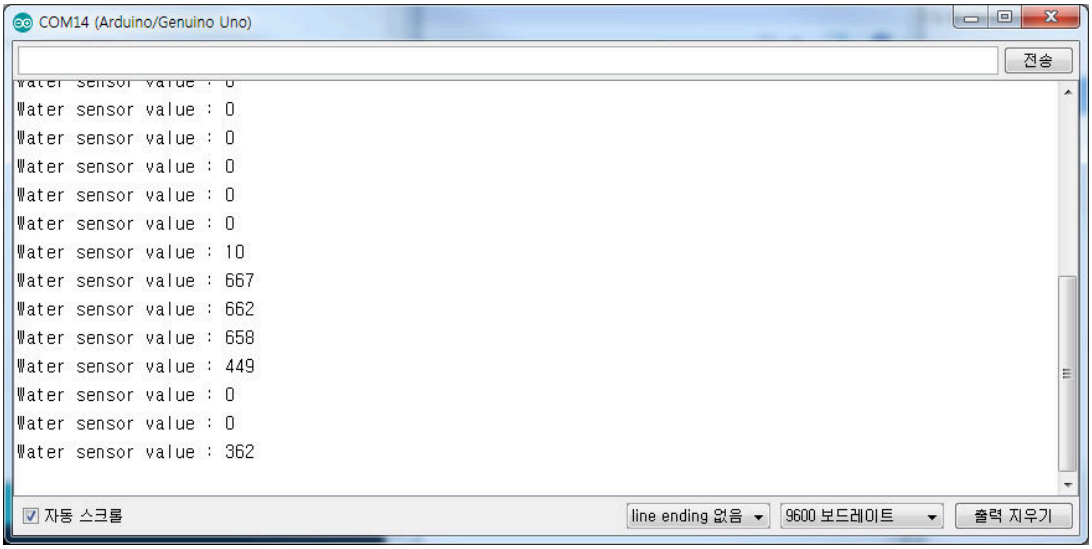
- 코딩

아날로그값(물센서)을 받아서 결과를 시리얼로 출력을 합니다.

```
//*********************************************************************************
// Project : Water Sensor in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
#define SensorIn A0
void setup(){
    Serial.begin(9600);
}
void loop(){
    int R = analogRead(SensorIn); //Read Sensor 값
    Serial.print("Water sensor value : ");
    Serial.println(R);    //시리얼 모니터에 출력
    delay(1000);
}
```

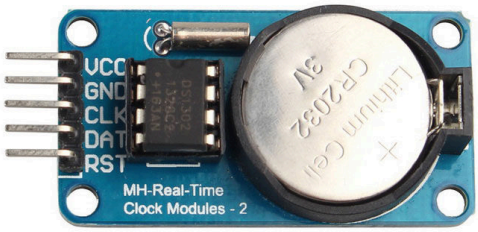
- 결과

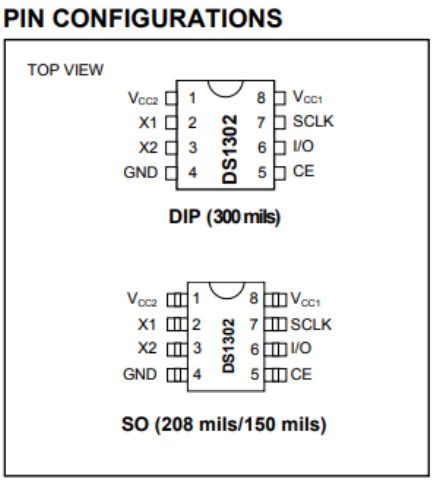
물이 인식되면 아날로그 값이 높아 집니다.



○ Clock 모듈

- 개요





RTC 는 Real Time Clock 의 약자입니다. 아두이노 내부에 Main clock 이 있고, Timer 가 있지만 시간을 계산하는데는 정확성에 한계가 있습니다.

RTC 모듈을 사용하는 이유는 아래와 같습니다.

- 자체 배터리를 사용해서 전원이 꺼진 상태도 시간이 유지
- 낮은 소비전력
- 날짜, 시간, 요일을 계산
- 32.768Khz Quartz Crystal 을 사용하므로 정확한 시간 계산이 가능

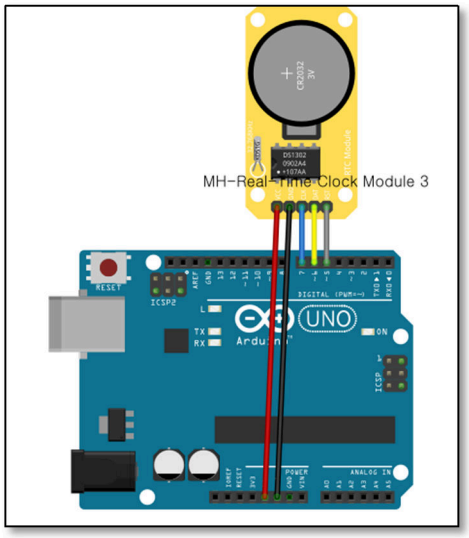
- 준비물

. 아두이노 우노(호환보드) X 1

. DS1302 RTC 모듈 X 1

. 점퍼와이어

- 연결도



- 코딩

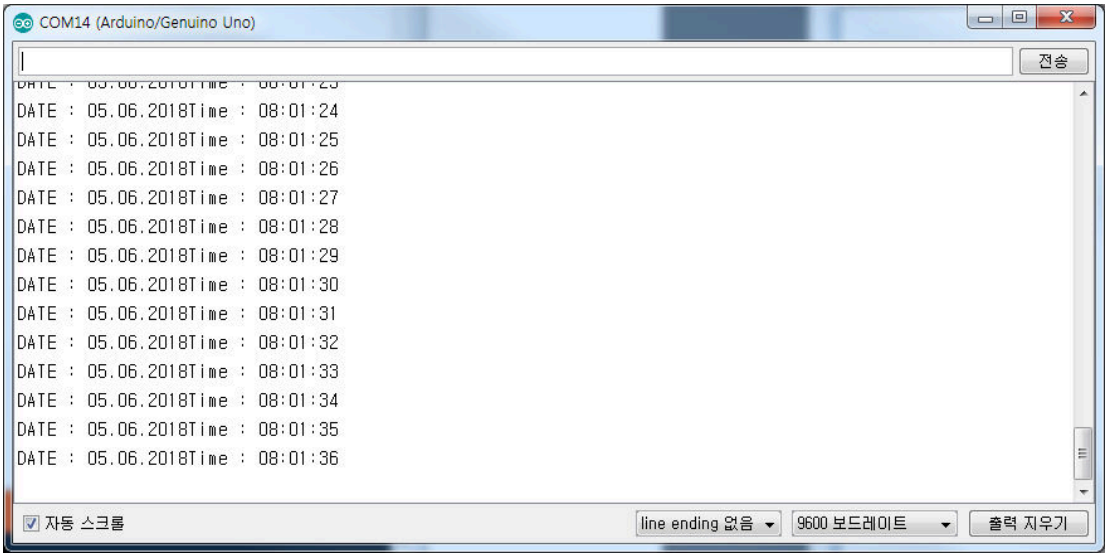
DS1302 라이브러리 설치가 필요합니다.

```
//*********************************************************************************
// Project : RTC Sensor in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
#include <DS1302.h>
DS1302 rtc(5, 6, 7);  // DS1302 init(Reset, Data, Clock)
void setup() {
  Serial.begin(9600);
  rtc.halt(false);              //Set Run-Mode
  rtc.writeProtect(false);      //disable write protect

  rtc.setDOW (WEDNESDAY);        // 일요일
  rtc.setTime(8, 0, 0);       // 08 시 00 분 00 초
  rtc.setDate(5, 06, 2018);     // 2017 년 4 월 5 일
}
void loop() {
  Serial.print("DATE : ");
  Serial.print(rtc.getDateStr());   // Output Date
  Serial.print("Time : ");
  Serial.println(rtc.getTimeStr());  // Output Time
  delay (1000);
}
```

- **결과**

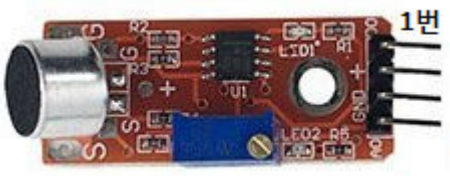
설정된 시간부터 현재시간을 표시합니다



○ 소리감지센서

- 개요

소리감지센서 (Big Sound module)에 대해서 알아 보겠습니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

소리감지센서는 주변에 발생하는 소리를 마이크로폰을 사용하여 소리 크기를 측정하는 센서입니다. 소리는 공기의 진동을 통해서 전달되는 신호입니다. 그러므로 소리감지센서는 공기의 진동을 인식해서 전기신호(전압)으로 출력해주는 장치입니다.

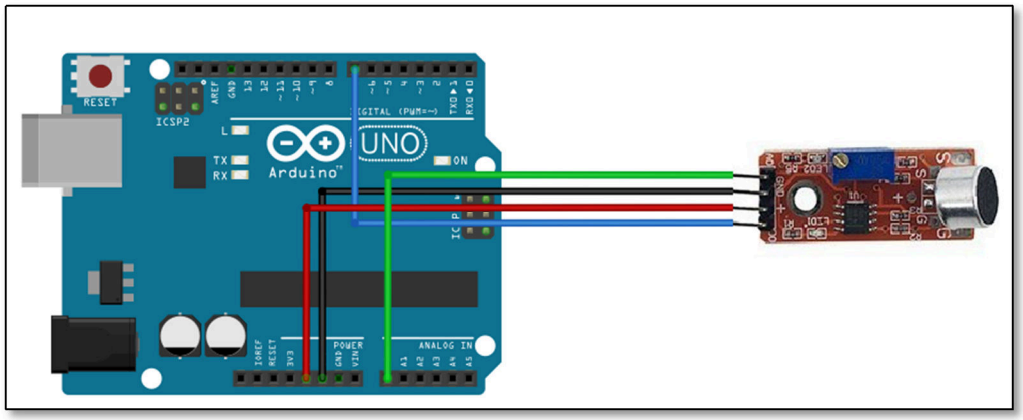
- 준비물

. 아두이노 우노(호환보드) X 1

. 소리감지센서 X 1

. 점퍼와이어

- 연결도



- 코딩

아날로그를 반복적으로 인식해서 특정소리 크기를 넘어가게 되면 LED 를 ON 합니다.

```
//*****************************************************************************
// Project : big sound in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int sensorPin = A0; // Sensor ouput
int ledPin = 13;     // for the LED
int sensorValue = 0; // Sensor Data

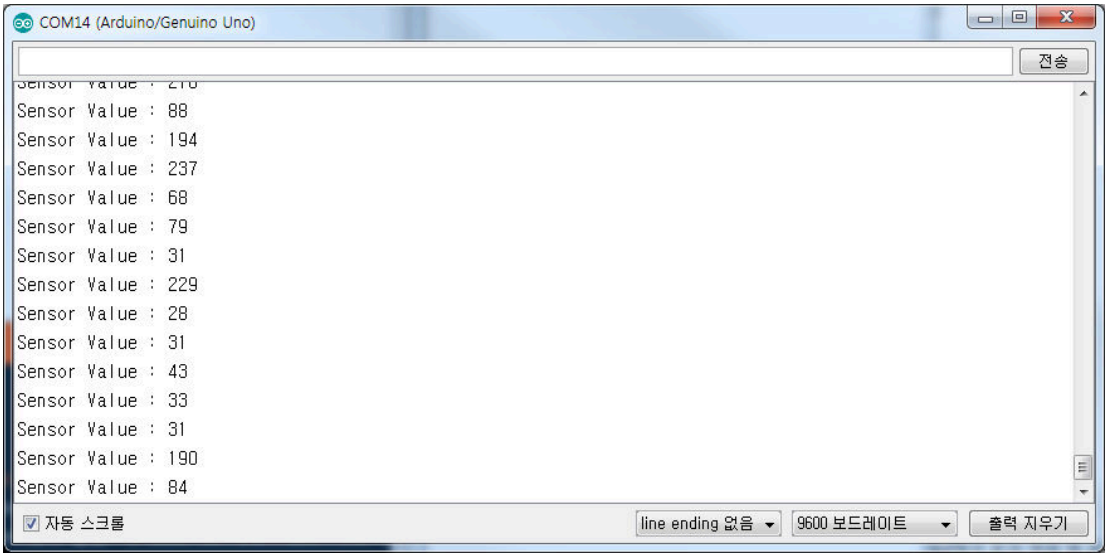
void setup ()
{
  pinMode (ledPin, OUTPUT);
  Serial.begin (9600);
}

void loop ()
{
  sensorValue = analogRead (sensorPin);
  if(sensorValue>100)
    digitalWrite (ledPin, HIGH);
  else
    digitalWrite (ledPin, LOW);

  Serial.print ("Sensor Value : ");
  Serial.println (sensorValue, DEC);
}
```

- 결과

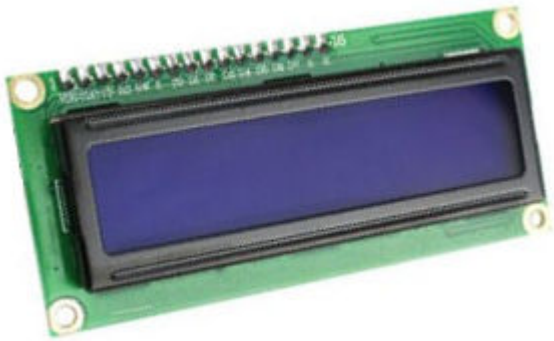
소리크기의 변화를 확인하실 수 있습니다.



○ 1602 LCD

- 개요

캐릭터 LCD 에 대해서 알아 보겠습니다.



1602 LCD 는 16 글자를 2 줄로 표현이 가능한 캐릭터 LCD 입니다. 백라이터를 내장 하고 있으며, 1602 LCD 인터페이스는 아래와 같습니다.

1602 LCD 핀맵

```
1 | Ground
2 | VCC (+3.3 to +5V)
3 | Contrast adjustment (sweeper of 10k potentiometer with
  | opposing contacts connected to VCC and GND respectively)
4 | Register Select (RS). 0: Command, 1: Data -> MSP430 P2.5
5 | Read/~Write (R/W). 0: Write, 1: Read -> GND
6 | Clock (Enable). Falling edge triggered -> MSP430 P2.4
7 | Data 0 (N.C.)
8 | Data 1 (N.C.)
9 | Data 2 (N.C.)
10| Data 3 (N.C.)
11| Data 4 -> MSP430 P2.0
12| Data 5 -> MSP430 P2.1
13| Data 6 -> MSP430 P2.2
14| Data 7 -> MSP430 P2.3
15| Backlight Anode (+) -> 100 ohm resistor to VCC
16| Backlight Cathode (-) -> GND
```

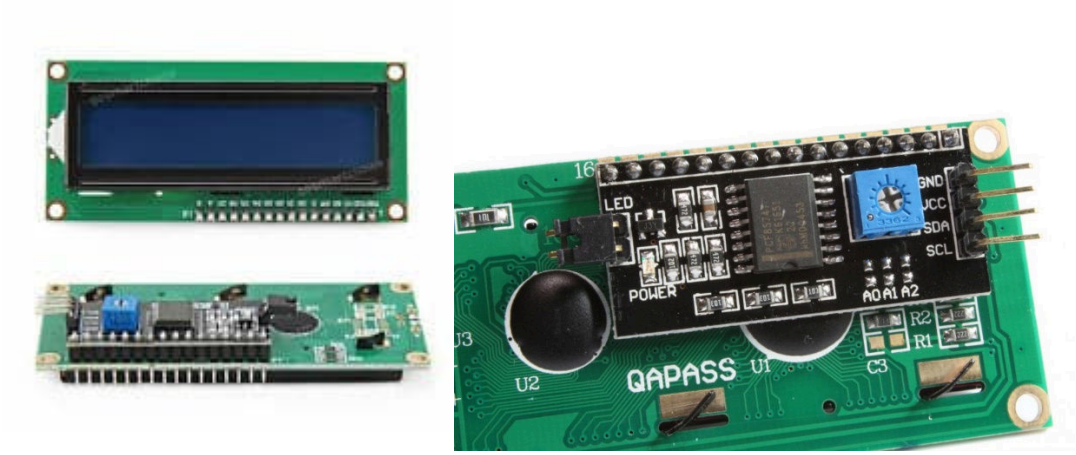
VSS 에 GND 를 VDD 에 5V 를 연결해주시면 됩니다.

3 번 V0 핀은 LCD 의 contrast(대비)를 조절하는 핀입니다. 가변저항을 통해서 선명도를 조정할 수 있습니다. RS 핀은 DB0-DB7 까지의 데이터핀의 기능을 결정합

니다. HIGH 일 경우 데이터를 읽고 쓰는 용도이고, LOW 일 경우는 명령어를 쓰거나 상태를 읽어올 수 있습니다. R/W 핀은 데이터핀의 읽기 또는 쓰기 모드를 결정합니다. 6 번 E 핀은 데이터를 보내기 위한 신호핀입니다. DB0-DB7 핀은 데이터를 읽고 쓰는 핀이며 15 번 16 번 핀은 백라이트 전원핀입니다.

[I2C LCD 보충 설명]

I2C 인터페이스 모듈을 포함하고 있는 I2C LCD 는 아래그림과 같이 생겼습니다.. 이전에 설명드린 오리지날 1602LCD 보다 연결해야 하는 점퍼선이 줄어들어 사용하기가 편리합니다. 아래 사진에서 검정색 I2C 모듈이 I2C 입력을 이전에 설명드린 1602 LCD 신호로 변환해주는 역할을 합니다.



- 준비물

. 아두이노 우노(호환보드) X 1

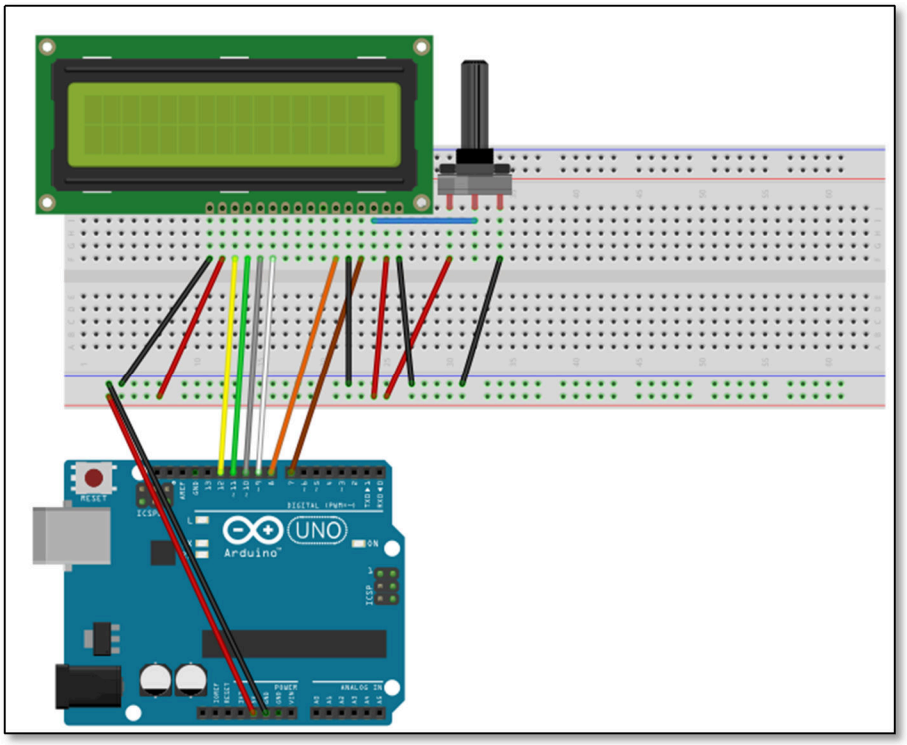
. 브래드보드 X 1

. 1602 LCD X 1 또는 I2C 1602 LCD X1

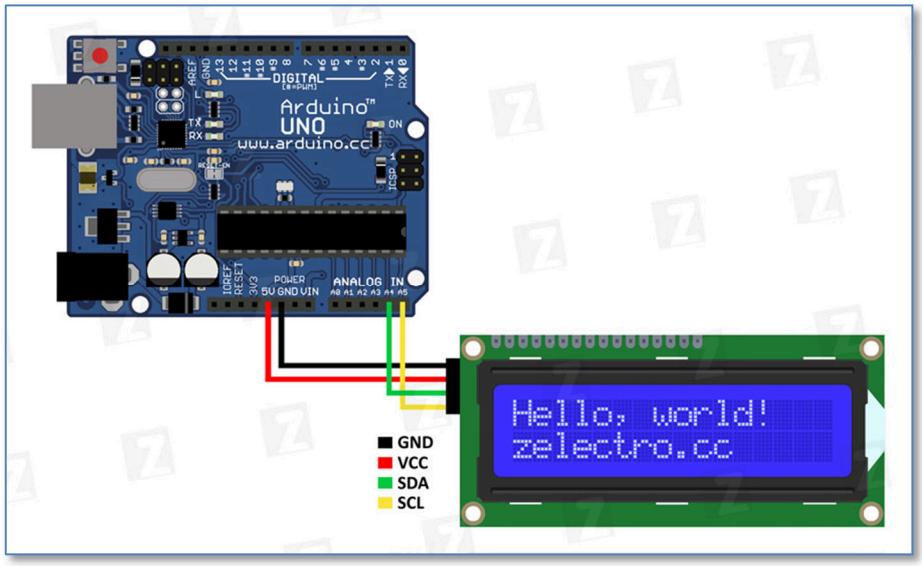
. 가변저항 10K X 1

. 점퍼와이어

- 연결도



[오리지날 1602 LCD 연결도]



[I2C 1602LCD 연결도]

- 코딩

오리지날 1602 LCD 는 LiquidCrystal 라이브러리 설치가 필요합니다.

```
//*********************************************************************************
// Project : 1602 LCD in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
```

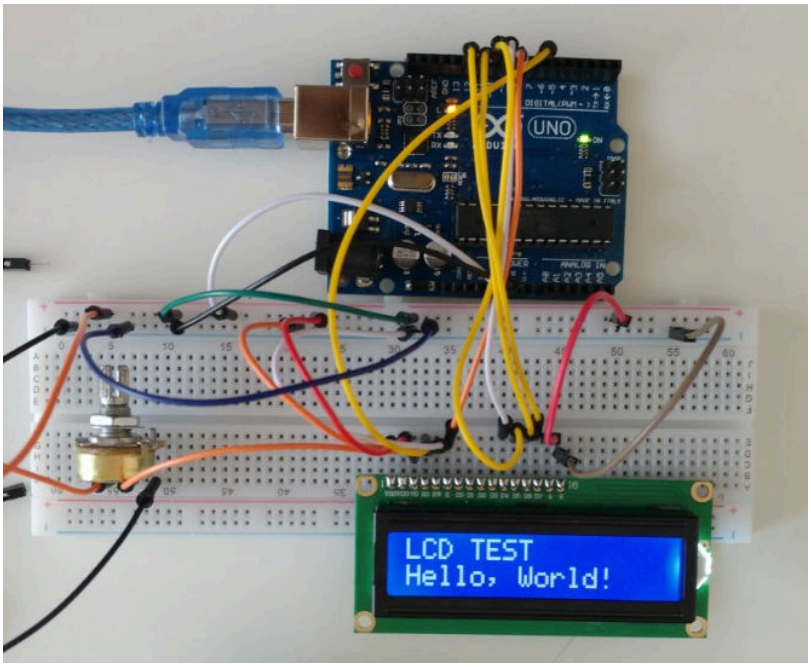
```
// include the library code:
#include <LiquidCrystal.h>
// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.setCursor(0,0);
  lcd.print("LCD TEST ");
  lcd.setCursor(0,1);
  lcd.print("Hello, World!");
}
void loop() {
}
```

I2C 1602 LCD 는 LiquidCrystal_I2C 라이브러리 설치가 필요합니다.

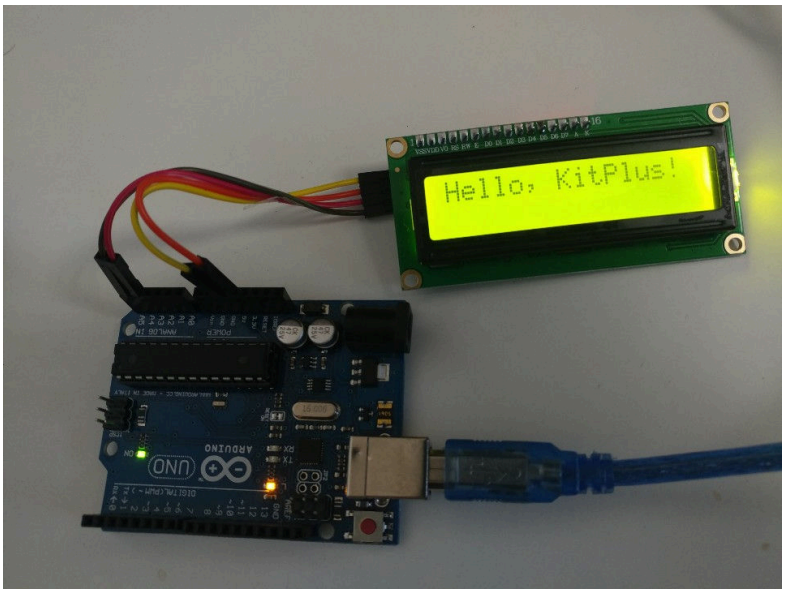
```
//*************************************************************************
// Project : 1602 LCD(for I2C)  in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
// include the library code:
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup()
{
        // initialize the LCD
        lcd.begin();
        // Turn on the blacklight and print a message.
        lcd.backlight();
        lcd.print("Hello, KitPlus!");
}
void loop()
{
        // Do nothing here...
}
```

- **결과**

LCD 에서 글자를 바로 확인하실 수 있습니다.



[오리지날 1602 LCD]

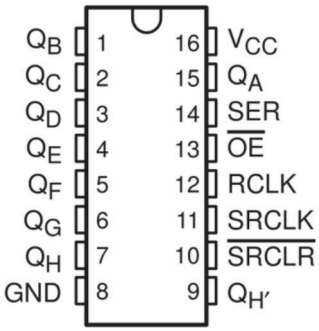


[I2C 1602 LCD]

○ 74HC595

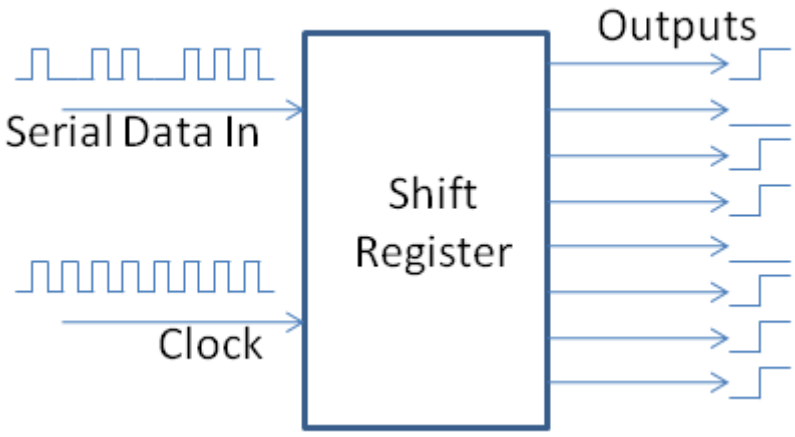
- 개요

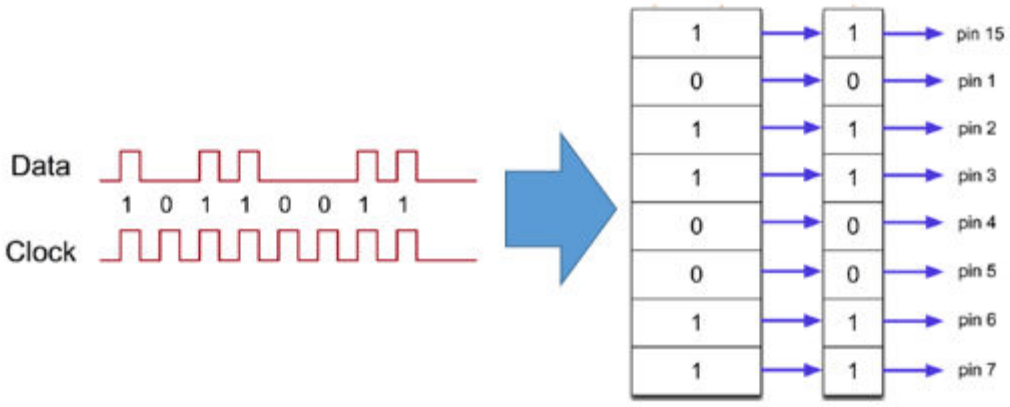




PINS 1-7, 15	Q0 " Q7	Output Pins
PIN 8	GND	Ground, Vss
PIN 9	Q7"	Serial Out
PIN 10	MR	Master Reclear, active low
PIN 11	SH_CP	Shift register clock pin
PIN 12	ST_CP	Storage register clock pin (latch pin)
PIN 13	OE	Output enable, active low
PIN 14	DS	Serial data input
PIN 16	Vcc	Positive supply voltage

74HC595 는 8 bit shifter 로 입력되는 직렬신호를 내부의 플립플롭을 사용하여 병렬신호변환해주는 IC 입니다. 3 개의 신호선 8 개의 출력을 동작 시킬 수 있습니다. 여러 개의 LED 를 제어할 경우 부족한 디지털 출력 포트를 보충할 수 있습니다.





[입력] [출력]

- 준비물

. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

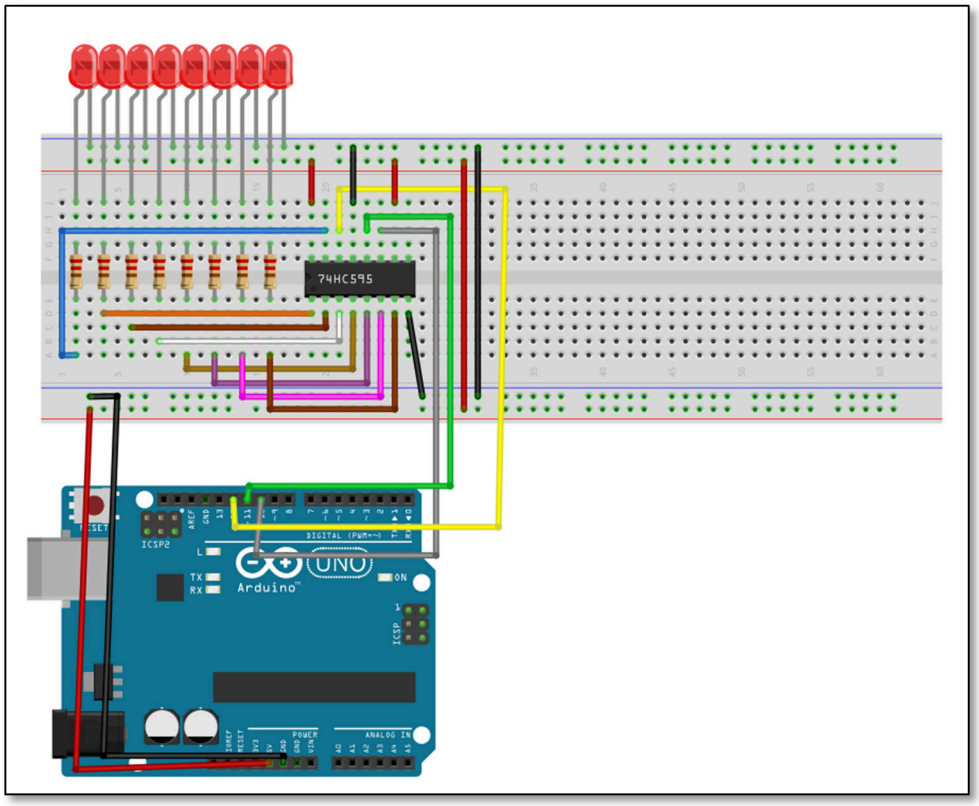
. 74HC595 X 1

. 빨강 LED X 8

. 220 옴 X 8

. 점퍼와이어

- 연결도



- 코딩

```
//********************************************************************************
// Project : 74hc595 LCD in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//********************************************************************************

int latchPin = 11;    //Pin connected to ST_CP of 74HC595
int clockPin = 8;   //Pin connected to SH_CP of 74HC595
int dataPin = 12;    //Pin connected to DS of 74HC595

byte leds = 0;

void setup() {
  pinMode(latchPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
}

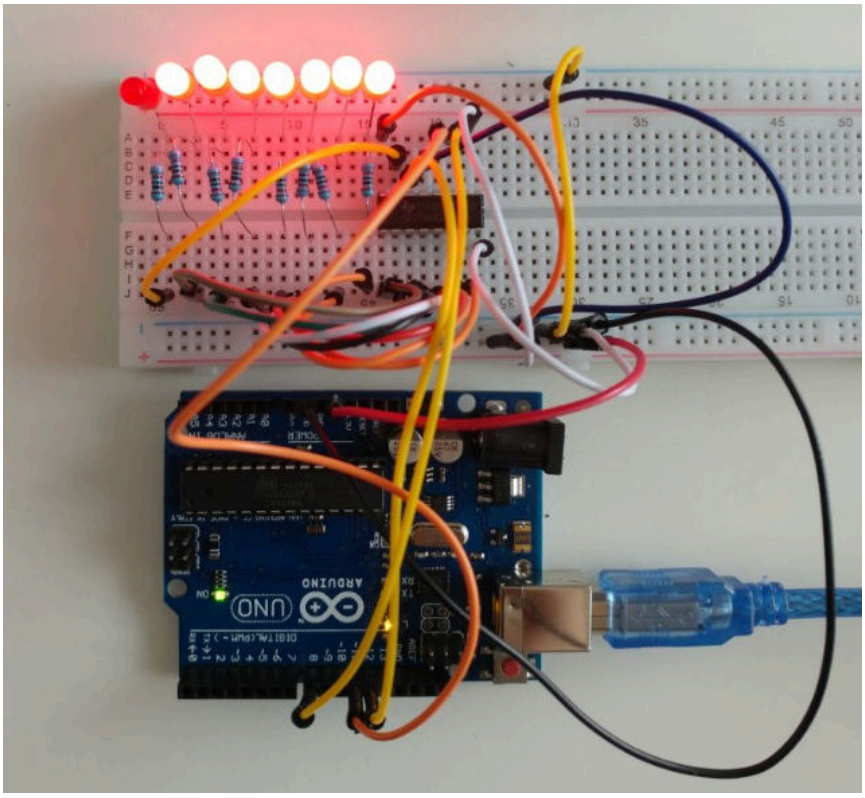
void loop() {
  leds = 0;
  updateShiftRegister();
  delay(500);
  for (int i = 0; i < 8; i++)  {
    bitSet(leds, i);
    updateShiftRegister();
    delay(500);
```

```
  }
}

void updateShiftRegister() {
   digitalWrite(latchPin, LOW);
   shiftOut(dataPin, clockPin, LSBFIRST, leds);
   digitalWrite(latchPin, HIGH);
}
```

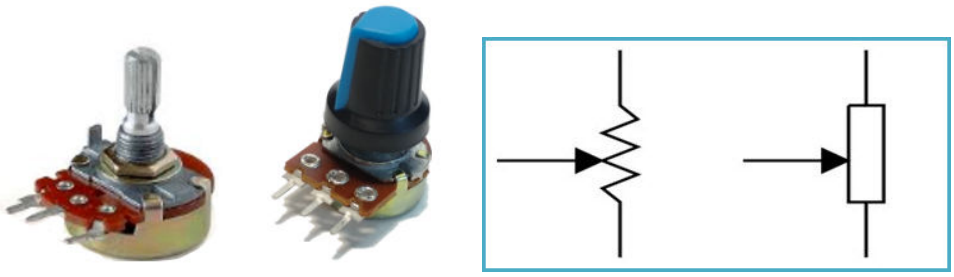
- **결과**

LED 가 순차적으로 움직이는 것을 확인할 수 있습니다. .



○ 아날로그 입력(포텐션미터)

- 개요

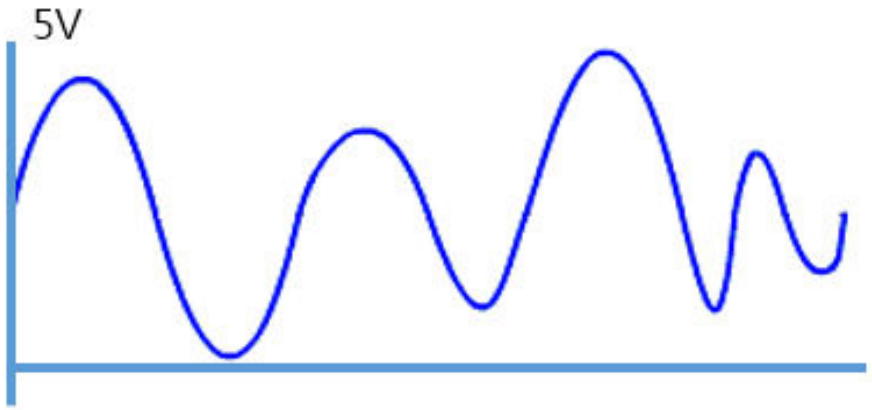


[가변저항] [가변저항 캡] [회로 심볼]

포텐션미터(Potentiometer)를 우리말로 전위차계, 분압기 등이 있지만 일반적으로 가변저항이라고 부릅니다.(이하 가변저항으로 칭합니다.) 즉, 고정되어있는 저항이 아닌 임의의 저항값으로 조절이 가능한 저항기입니다. 대부분의 가변저항은 최소저항이 0 옴으로 정해져 있고, 최대 저항은 제품별로 1K, 10K, 100K 로 결정이 되어있습니다.

우리가 주변에서 가변저항을 가장 많이 접하는 곳은 음향장치에 사용하는 볼륨이나 조명 밝기를 조절하는 장치들이 있습니다.

가변저항의 원리를 간단히 설명드리면 내부에 카본(탄소) 띠로 구성되어있는 저항 성분을 와이퍼로 길이를 조절해서 저항 값이 조절되도록 구조입니다.



[아날로그 신호 예]

- 준비물

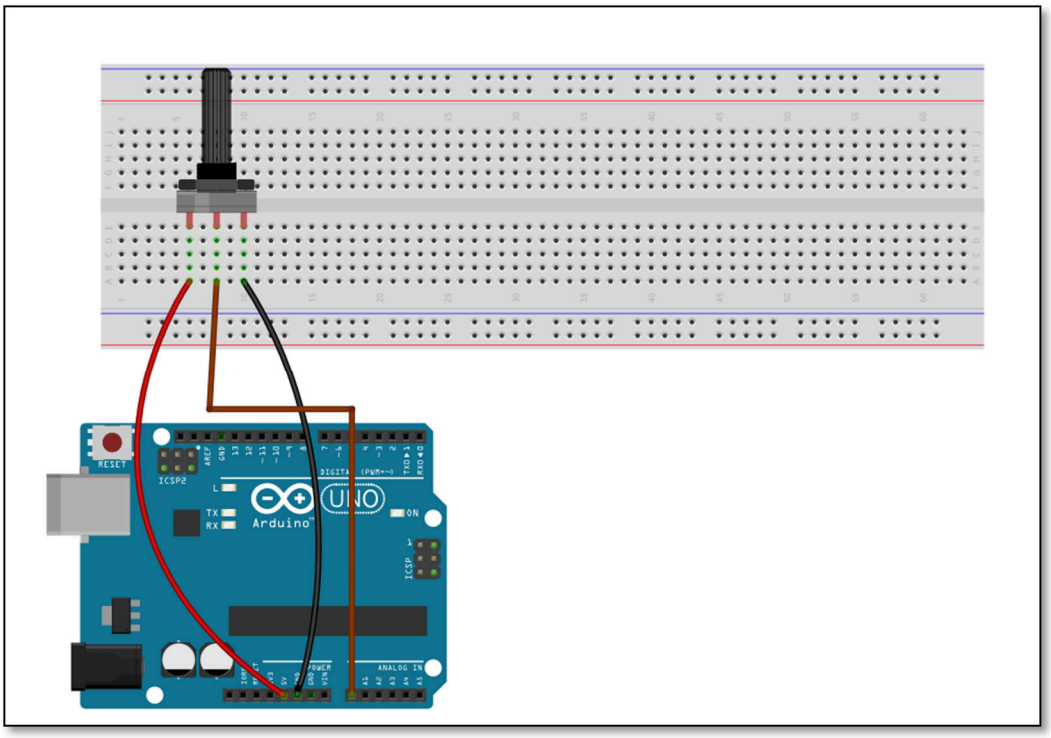
. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

. 포텐션미터 X 1

. 점퍼와이어

- **연결도**



- **코딩**

```
//*****************************************************************************
// Project : potentiometer in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
 #define SensorIn A0
#define LedPin 13
void setup(){
    Serial.begin(9600);
}
void loop(){

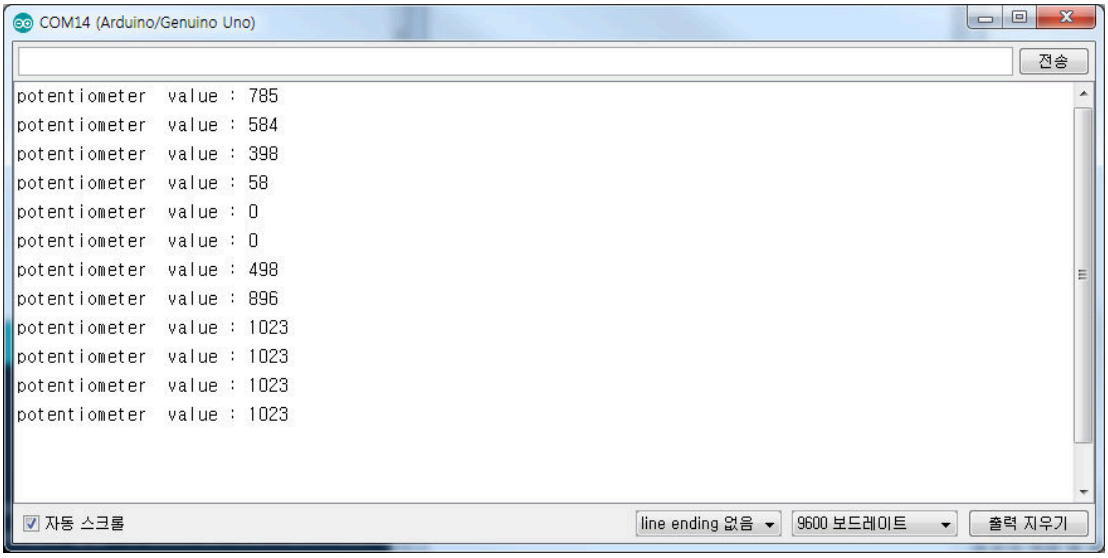
    int R = analogRead(SensorIn); //Read Sensor 값
    Serial.print("potentiometer  value : ");
    Serial.println(R);    //시리얼 모니터에 출력
```

```
    delay(1000);
}
```

- **결과**

가변저항을 회전할 경우 전압(5V)이 분배되는 것을(0~1024) 확인할 수 있습니다.

값 x 0.0049 하시면 입력되는 전압 값을 구할 수 있습니다.



○ 조도센서

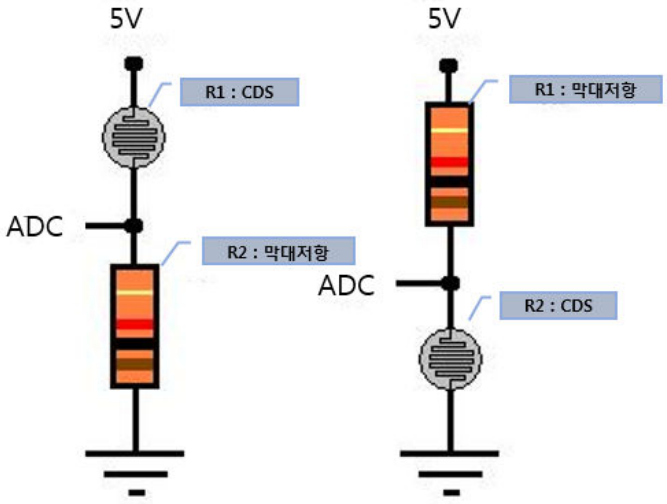
- 개요

조도센서란 빛의 밝기를 측정 할 수 있는 센서입니다. 빛을 받으면 내부의 저항 값이 변하는 광전효과를 재료를 사용합니다. CDS 라고도 부르는 이유는 황화카드뮴(cadmium sulfide)을 소자로 사용하기 때문입니다.



일반적인 연결 예제는 아래 그림과 같고,

아두이노로 입력되는 ADC 의 계산 값은 ADC = 5V * (R2/(R1+R2)) 입니다.



- 준비물

. 아두이노 우노(호환보드) X 1

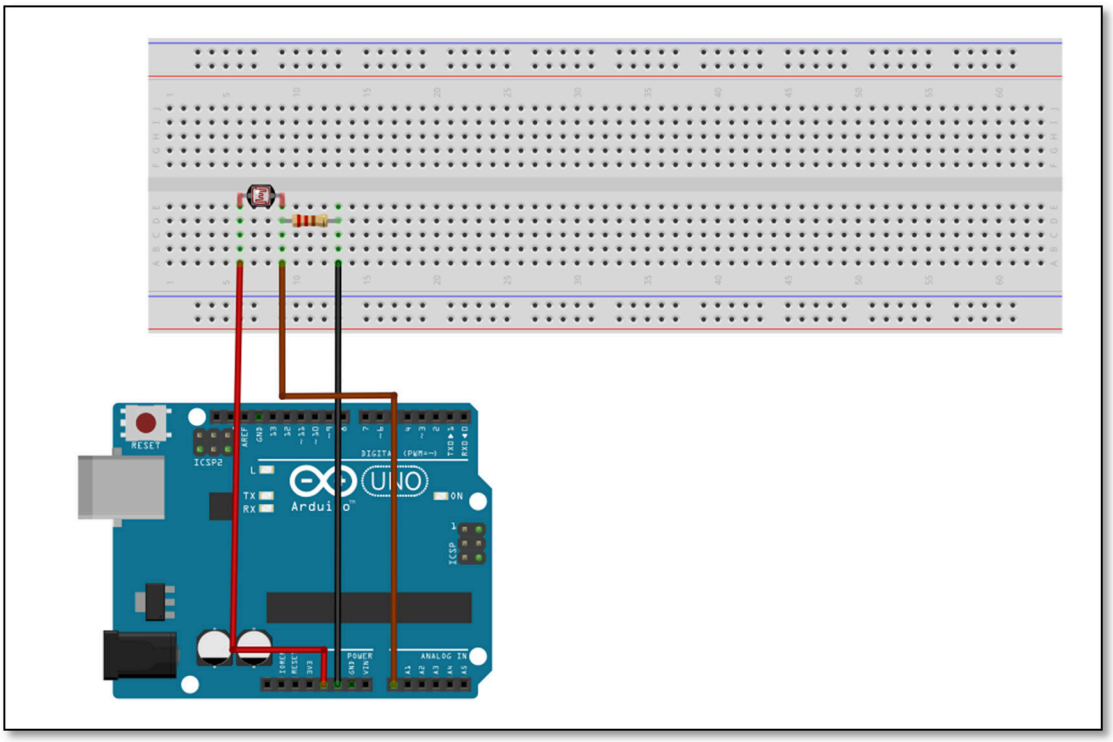
. 브래드보드 X 1

. 조도센서 X 1

. 1K 저항 X 1

. 점퍼와이어

- **연결도**

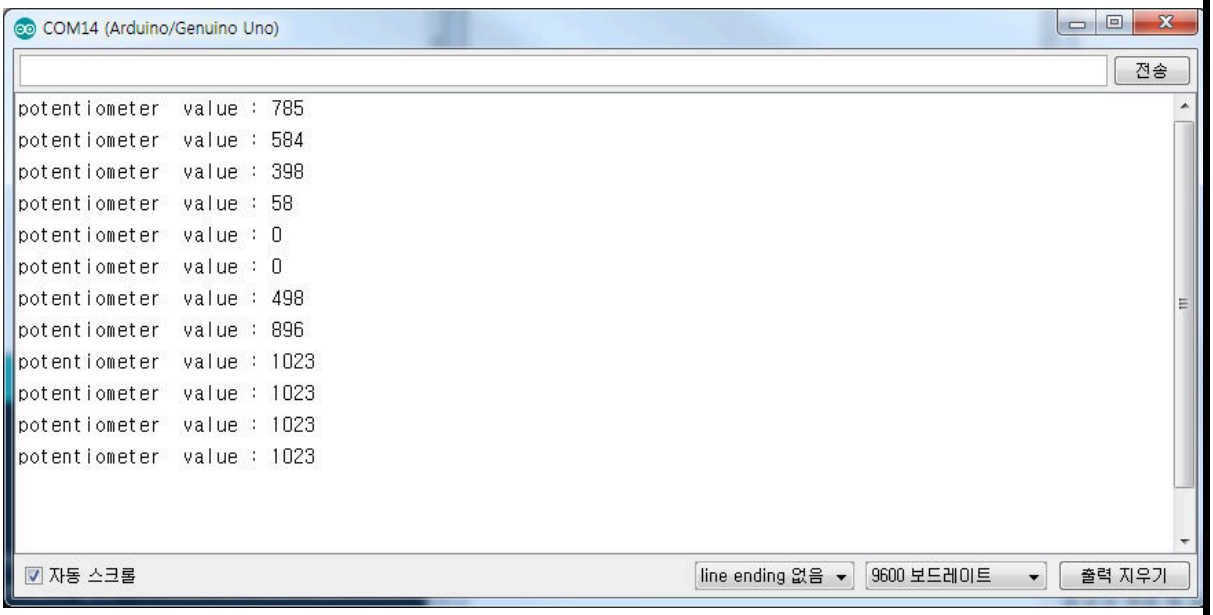


- **코딩**

```
//*********************************************************************************
// Project : CDS in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
#define  SensorIn A0
void setup(){
    Serial.begin(9600);
}
void loop(){
    int R = analogRead(SensorIn); //Read Sensor 값
    Serial.print("CDS  value : ");
    Serial.println(R);    //시리얼 모니터에 출력
    delay(1000);
}
```

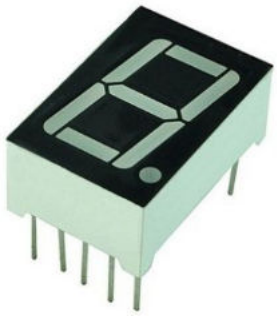
- **결과**

빛이 밝으면 입력 값이 높아 집니다..

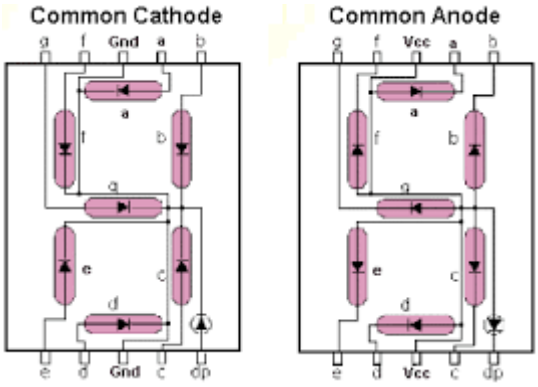


○ 7세그먼트

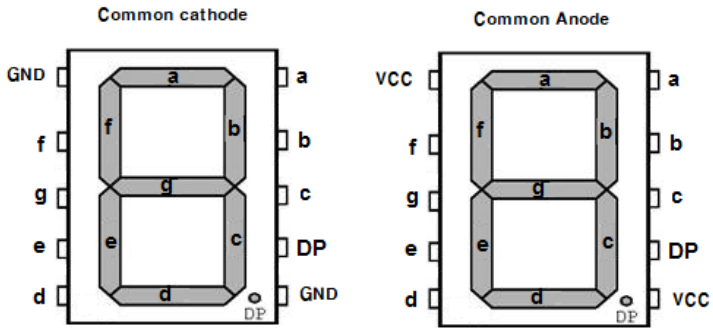
- 개요



7 세그먼트란 LED 를 포함하고 있는 7 개의 선분으로 구성되어 있는 표시장치 입니다. 7 개의 LED 를 켜거나 끄는 효과를 통해서 아라비아 숫자 형태로 표현이 가능합니다. 입력되는 전압의 방향에 따라 Cathode type 과 Anode type 으로 구별이 됩니다.



(가로 방향 FND 핀맵)



[세로방향 FND 핀맵]

7 세그먼트를 1 개 자리를 표하기 위해서는 점(Dot)을 포함해서 8 개의 디지털 신호가 필요하지만 효율적인 하드웨어 구성을 위해서 74HC595 와 같은 쉬프트 IC 를 사용하여 3 개 신호선만으로도 7 세그먼트를 표현할 수 있습니다.

다음 Table 을 사용해서 아라비아 숫자를 표시할 수 있습니다.(Cathode type 과 Anode type 에 의해서 On/Off 가 반대로 될 수 있습니다.)

DIG	HEX	a	b	c	d	e	f	g
0	0×7E	on	on	on	on	on	on	off
1	0×30	off	on	on	off	off	off	off
2	0×6D	on	on	off	on	on	off	on
3	0×79	on	on	on	on	off	off	on
4	0×33	off	on	on	off	off	on	on
5	0×5B	on	off	on	on	off	on	on
6	0×5F	on	off	on	on	on	on	on
7	0×70	on	on	on	off	off	off	off
8	0×7F	on	on	on	on	on	on	on
9	0×7B	on	on	on	on	off	on	on
A	0×77	on	on	on	off	on	on	on
B	0×1F	off	off	on	on	on	on	on
C	0×4E	on	off	off	on	on	on	off
D	0×3D	off	on	on	on	on	off	on
E	0×4F	on	off	off	on	on	on	on
F	0×47	on	off	off	off	on	on	on

- 준비물

. 아두이노 우노(호환보드) X 1

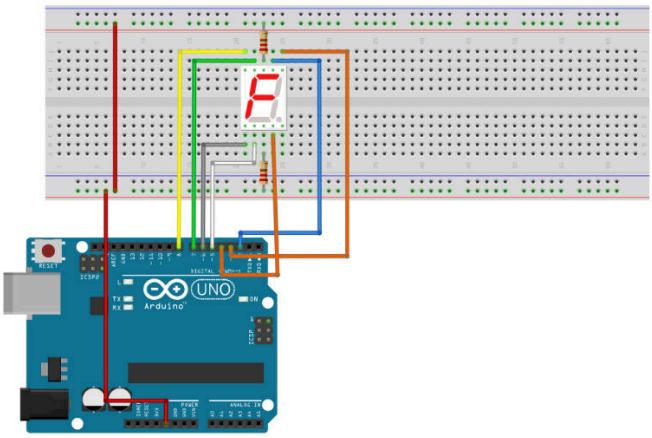
. 브래드보드 X 1

. FND Segment X 1

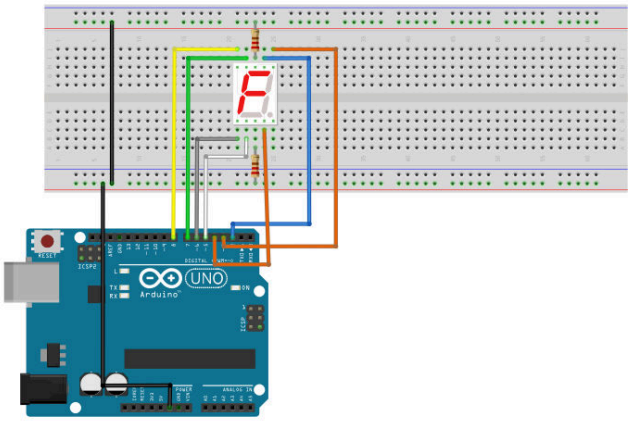
. 220 옴 저항 X 2

. 점퍼와이어

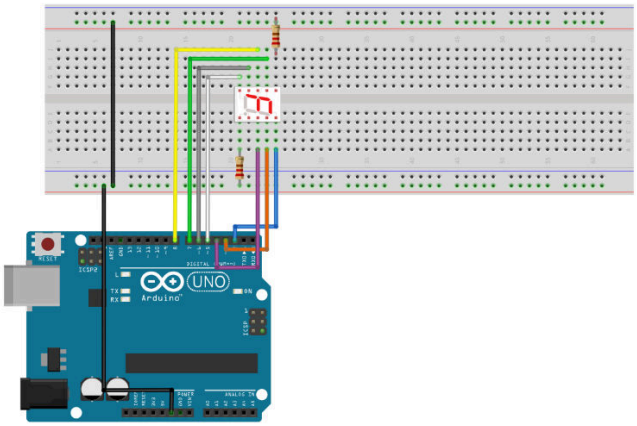
- **연결도**



[FND 가로 핀맵 Anode type 연결도]



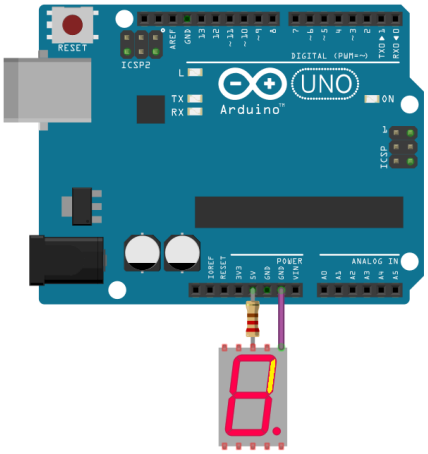
[FND 가로 핀맵 Cathode Type 연결도]



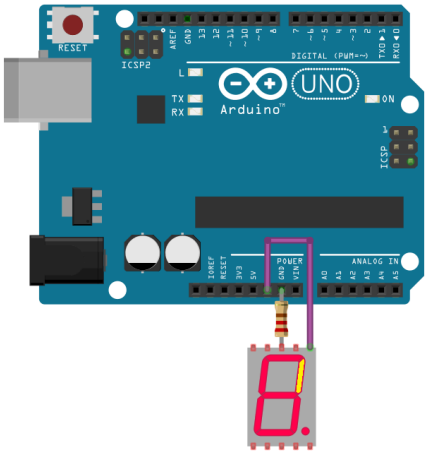
[FND 세로 핀맵 Cathode Type 연결도]

* FND Type 구분법

아래와 같이 연결해서 'a' LED의 On/OFF를 보고 Anode / Cathode를 구분할 수 있습니다.

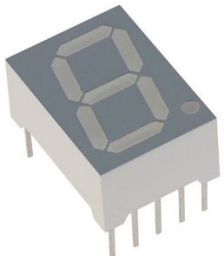


[Anode]

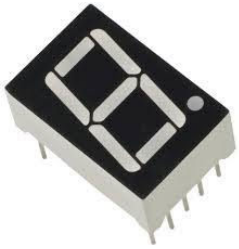


[Cathode]

키트플러스에서 현재까지 제공된 FND type



[Anode]



[Cathode]



[Cathode]

- 코딩

Cathod type의 FND를 사용하시려면," #define ANODE_TYPE"을 삭제 또는 주석 처리를 하시면 됩니다.

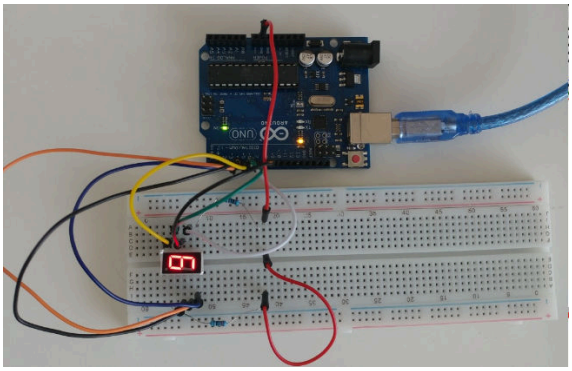
```
//*********************************************************************************
// Project : FND 7 segement in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
#define ANODE_TYPE
int num_array[10][7] = {  { 1,1,1,1,1,1,0 },    // 0
                          { 0,1,1,0,0,0,0 },    // 1
                          { 1,1,0,1,1,0,1 },    // 2
                          { 1,1,1,1,0,0,1 },    // 3
                          { 0,1,1,0,0,1,1 },    // 4
                          { 1,0,1,1,0,1,1 },    // 5
                          { 1,0,1,1,1,1,1 },    // 6
```

```
                      { 1,1,1,0,0,0,0 },    // 7
                      { 1,1,1,1,1,1,1 },    // 8
                      { 1,1,1,0,0,1,1 }};   // 9
//display fnd
void Num_Write(int);
void setup()
{
  // set pin modes
  pinMode(2, OUTPUT);    //a
  pinMode(3, OUTPUT);    //b
  pinMode(4, OUTPUT);    //c
  pinMode(5, OUTPUT);    //d
  pinMode(6, OUTPUT);    //e
  pinMode(7, OUTPUT);    //f
  pinMode(8, OUTPUT);    //b

  for (int j=2; j <= 8; j++) {
   digitalWrite(j, HIGH);
  }
}
void loop()
{
  for (int counter = 10; counter > 0; --counter)
  {
   delay(1000);
   Num_Write(counter-1);
  }
  delay(3000);
}
void Num_Write(int number)
{
  int pin= 2;
  for (int j=0; j < 7; j++){
#ifdef ANODE_TYPE
   digitalWrite(pin, !num_array[number][j]);
#else
   digitalWrite(pin, num_array[number][j]);
#endif
   pin++;
  }
}
```

- **결과**

9 부터 0 까지 Down 카운터 되는 것을 확인하실 수 있습니다.



○ 4자리 7 세그먼트

- 개요

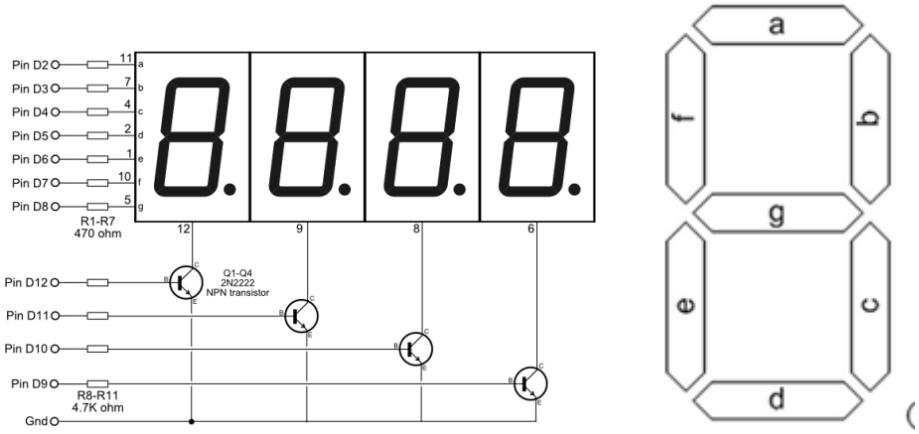


온도, 시계 등을 사용하기 위해서는 1 자리 세그먼트 가지고는 부족합니다. 4 자리 7 세그먼트를 사용하는 방법에 대해서 알아보겠습니다.

원칙적으로 4 자리 세그먼트를 표시하기 위해서는 32 개의 디지털 출력 핀이 필요합니다. 8bit shift IC 를 사용하더라도 4 개의 IC 가 필요합니다.

하지만 4 자리 세그먼트에는 12 개 디지털 핀밖에 없습니다. 그 이유는 잔상효과를 사용하기 때문입니다. 4 자리의 8 선분을 모두 공통으로 연결하고 1 자리씩 표시를 하게 되면 12 개 디지털출력으로 4 자리 세그먼트를 표시할 수 있습니다. 1 자리씩 이동되는 시간은 10ms 라고 생각하면 4 자리를 모두 표시하는데 40ms 가 필요합니다. 사람의 망막은 1/16 초(62.5ms) 동안 잔상이 남아 있습니다.

그러므로 짧은 시간을 반복해서 한자리를 표시한다면 사람은 4 자리 모두 눈에 잔상이 남게 되어있습니다. 애니메이션, 영화, 전광판과 같은 효과입니다.



[출처 https://www.electroschematics.com/10512/arduino-4-digit-7-segment-led-display/]

- 준비물

. 아두이노 우노(호환보드) X 1

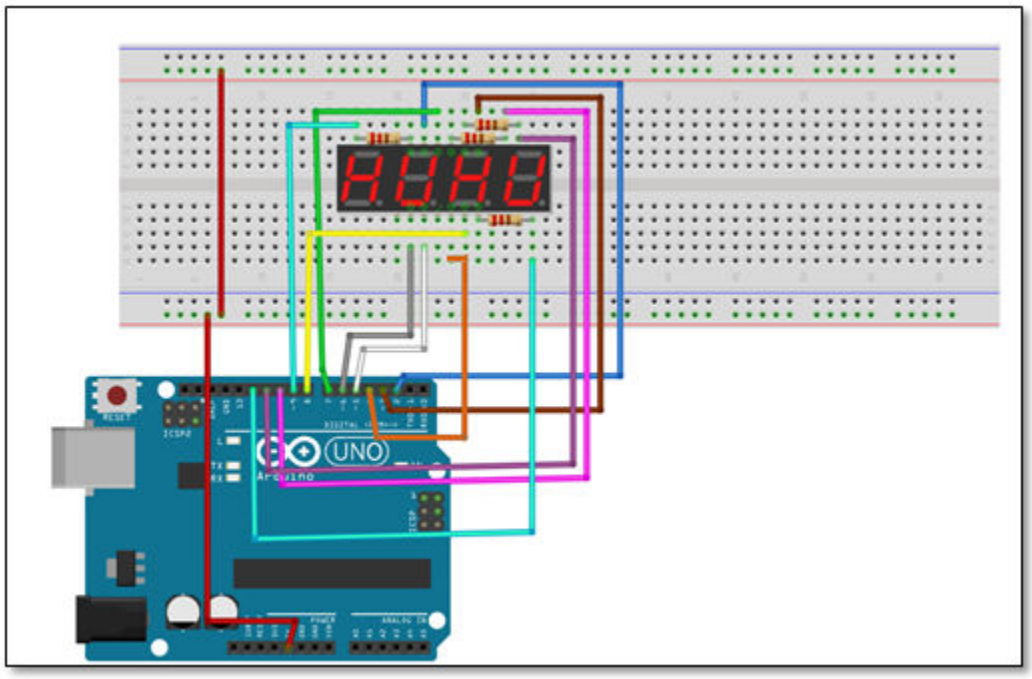
. 브래드보드 X 1

. 4 자리 FND Segment X 1

. 220 옴 저항 X 4

. 점퍼와이어

- 연결도



- 코딩

```
//*****************************************************************************
// Project : 4Dig FND 7 segement in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int num_array[10][7] = {  { 1,1,1,1,1,1,0 },    // 0
                          { 0,1,1,0,0,0,0 },    // 1
                          { 1,1,0,1,1,0,1 },    // 2
```

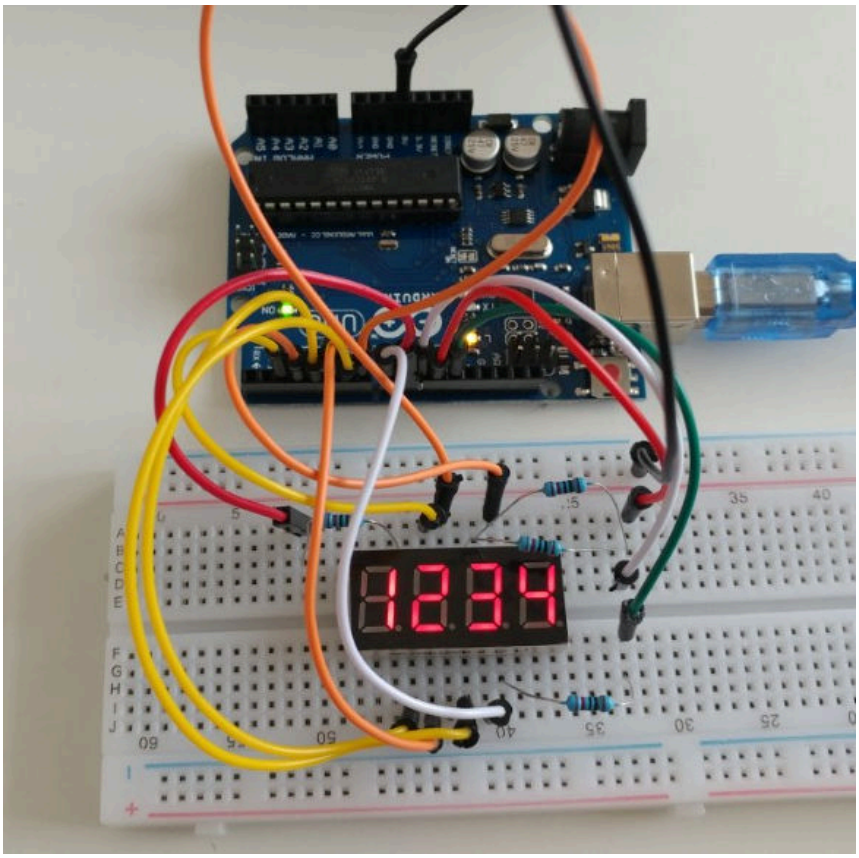
```
                        { 1,1,1,1,0,0,1 },    // 3
                        { 0,1,1,0,0,1,1 },    // 4
                        { 1,0,1,1,0,1,1 },    // 5
                        { 1,0,1,1,1,1,1 },    // 6
                        { 1,1,1,0,0,0,0 },    // 7
                        { 1,1,1,1,1,1,1 },    // 8
                        { 1,1,1,0,0,1,1 }};   // 9
int D1=9;
int D2=10;
int D3=11;
int D4=12;

//display fnd
void Num_Write(int);
void setup()
{
  // set pin modes
  pinMode(2, OUTPUT);    //a
  pinMode(3, OUTPUT);    //b
  pinMode(4, OUTPUT);    //c
  pinMode(5, OUTPUT);    //d
  pinMode(6, OUTPUT);    //e
  pinMode(7, OUTPUT);    //f
  pinMode(8, OUTPUT);    //b
  pinMode(9, OUTPUT);     //D1
  pinMode(10, OUTPUT);    //D2
  pinMode(11, OUTPUT);    //D3
  pinMode(12, OUTPUT);    //D4
  for (int j=2; j <= 8; j++) {
   digitalWrite(j, LOW);
  }
  for (int j=9; j <= 12; j++) {
   digitalWrite(j, HIGH);
  }
}
void loop()
{
    static int i=0,counter=0;

    Num_Write(1,D1);
    Num_Write(2,D2);
    Num_Write(3,D3);
    Num_Write(4,D4);
}
void Num_Write(int number,int dig)
{

  int pin= 2;
  digitalWrite(D1,  HIGH);
  digitalWrite(D2,  HIGH);
  digitalWrite(D3,  HIGH);
  digitalWrite(D4,  HIGH);
  digitalWrite(dig, LOW);
  for (int j=0; j < 7; j++){
   digitalWrite(pin, num_array[number][j]);
   pin++;
  }
  delay(2);
}
```

- **결과**



○ 릴레이

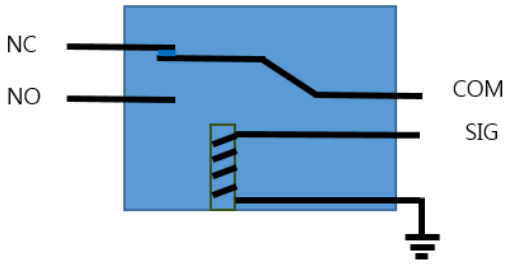
- 개요

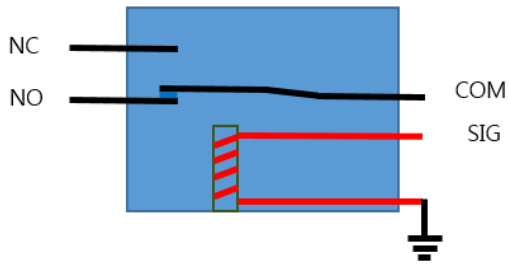
릴레이(Relay)에 대해서 알아 보겠습니다.



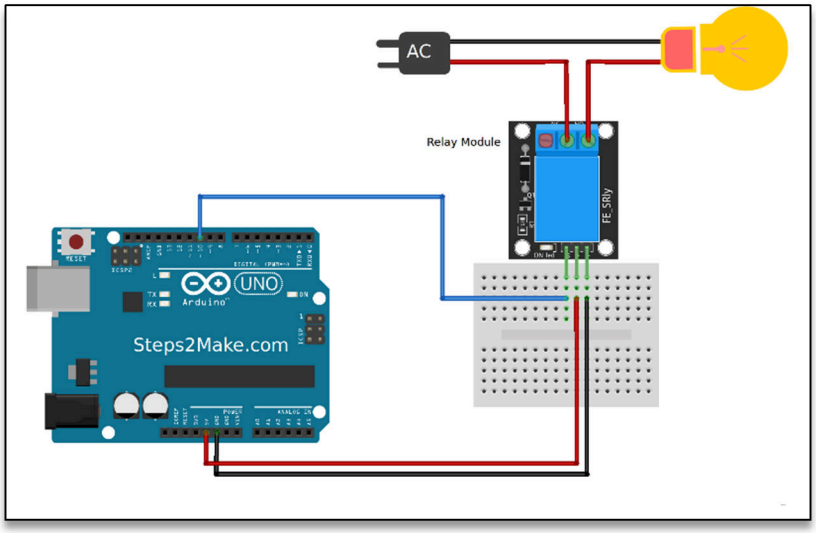
릴레이는 스위치에 종류로 전기신호로 제어가 가능합니다. 내부 구조는 코일에 전류를 흘리게 되면 모터처럼 자기장이 형성되어 내부 스위치의 열고 닫을 수 있습니다.

아래 왼쪽그림은 Signal 의 입력이 없을 때, 오른쪽그림은 Signal 에 전류가 흐를 때 자기장에 의해서 스위치가 닫는 그림입니다.





릴레이는 TR 이나 FET 와 다르게 자기장에 의한 물리적 스위치가 동작하기 때문에 높은 전압/전류에 응용될 수 있습니다. 보통 10A/300V 까지 연결이 가능합니다.



[IOT 예 출처 : https://steps2make.com/2017/10/arduino-5v-relay-module-ky-019/]

- 준비물

. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

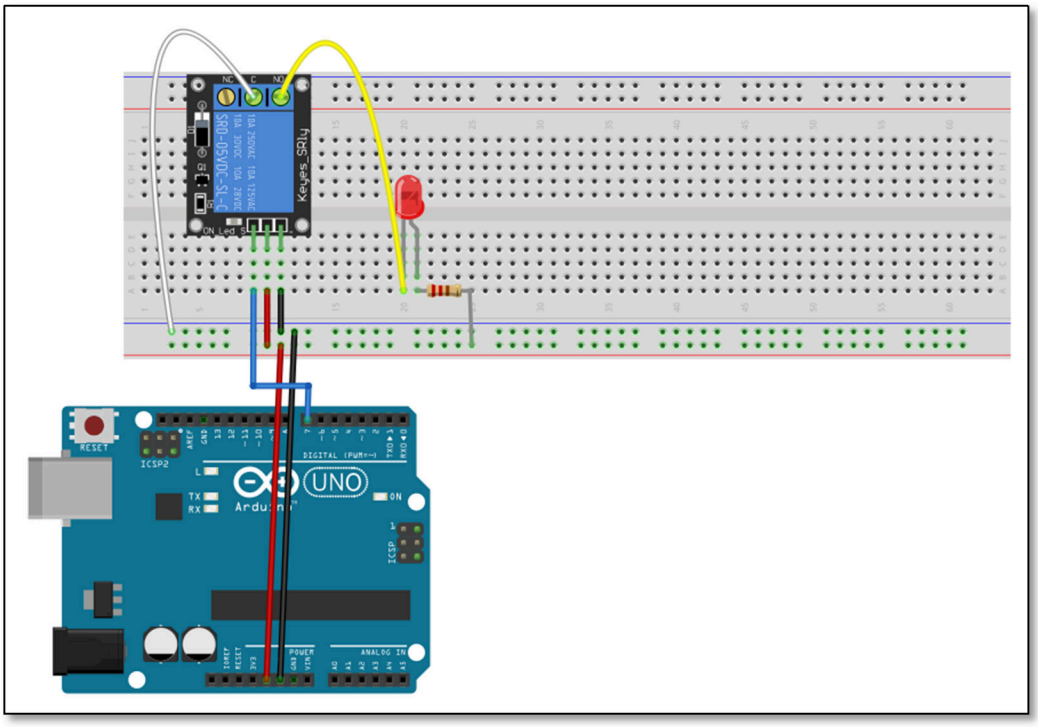
. 릴레이모듈 X 1

. 빨강 LED X 1

. 220 옴 저항 X 1

. 점퍼와이어

- 연결도

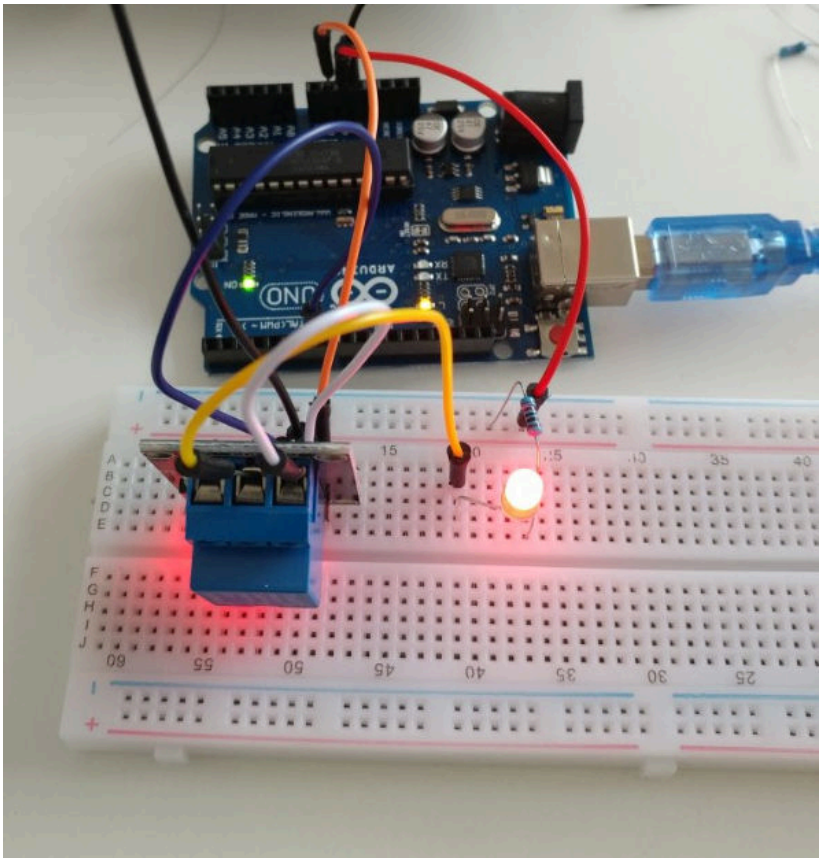


- 코딩

```
//*****************************************************************************
// Project : RELAY for led  in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
#define RELAY 7
void setup()
{
  // set pin modes
  pinMode(RELAY, OUTPUT);    //b
}
void loop()
{
  digitalWrite(RELAY,HIGH);
  delay(1000);
  digitalWrite(RELAY,LOW);
  delay(1000);
}
```

- **결과**

1 초에 한번씩 딸깍 소리와 함께 LED 가 ON 되는 것을 확인할 수 있습니다.



○ 스탭모터

- 개요

스텝핑(Stepping) 모터 대해서 알아 보도록 하겠습니다.



[28BYJ48 스테핑모터]

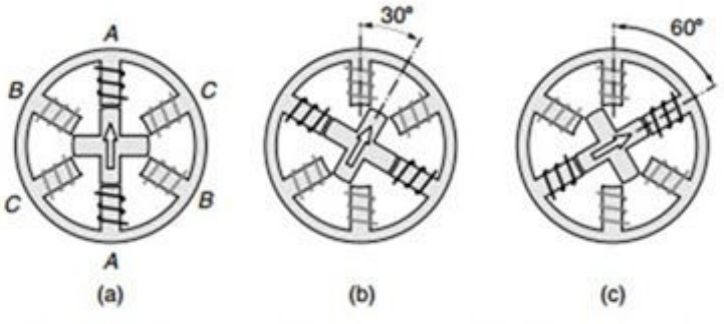


[ULN2003 드라이버 모듈]

스테핑 모터는 명칭이름과 같이 Step 단위로 이동이 가능한 모터입니다. 다시 말해서 1 개의 펄스가 입력이 될 때 정해진 각도로 1 스탭 회전이 가능한 모터입니다.

일반적인 서보 모터는 180 까지 회전이 가능하지만, 스테핑 모터는 무한 회전이 가능합니다. 하지만 엔코더가 없어서 자기 위치를 알 수 없고, 모터를 구동 하기 위해 별도의 스탭 모터 컨트롤로가 필요합니다.

아래그림은 스탭핑모터의 이해를 쉽게 하기 위해서 30 도 회전이 가능한 스탭모터 구조입니다.



[출처 : http://machineryequipmentonline.com/]

a) A 로 전류를 흘리면 플레밍의 왼손법칙에 의해서 코일에서 자기장이 형성되어 모터의 축이 북쪽을 향하게 되어있습니다.

b) B 로 전류를 흘리게 되면 A 의 자기장은 사리지고 B 의 자기장으로 회전 축이 회전 하게 되었습니다.

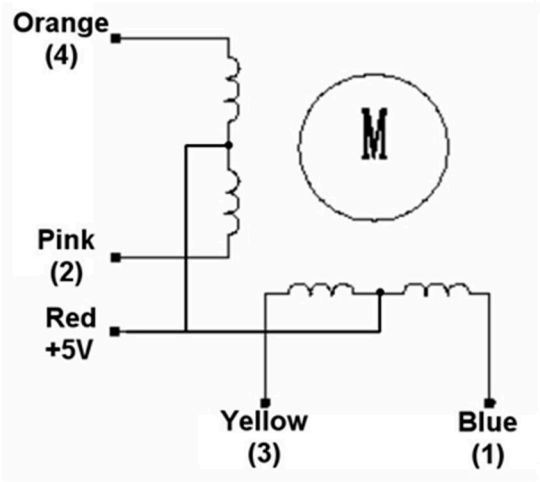
c) C 로 전류를 흘리게 되면 B 의 자기장은 사라지고 C 의 장기장으로 회전 측이 회전 하게 되어있습니다.

28BYJ48 스테핑모터는 5V 구동이 가능한 4 상(단계) 유니폴라 스텝모터입니다.

위에 설명 드린 것처럼 4 상은 서로 마주하는 개수가 4 개라는 뜻입니다. 즉, 4 개의 전자석이 반대쪽에 하나의 전자석을 마주보고 있는 구조입니다.

스탭모터 드라이버 모듈인 ULN2003 은 유니폴라 스텝모터를 제어하기 위한 드라이버 모듈입니다.

WIRING DIAGRAM



SWITCHING SEQUENCE

Lead Wire Color	---> CW Direction (1-2 Phase)							
	1	2	3	4	5	6	7	8
4 Orange	-	-						-
3 Yellow		-	-	-				
2 Pink				-	-	-		
1 Blue						-	-	-

- 준비물

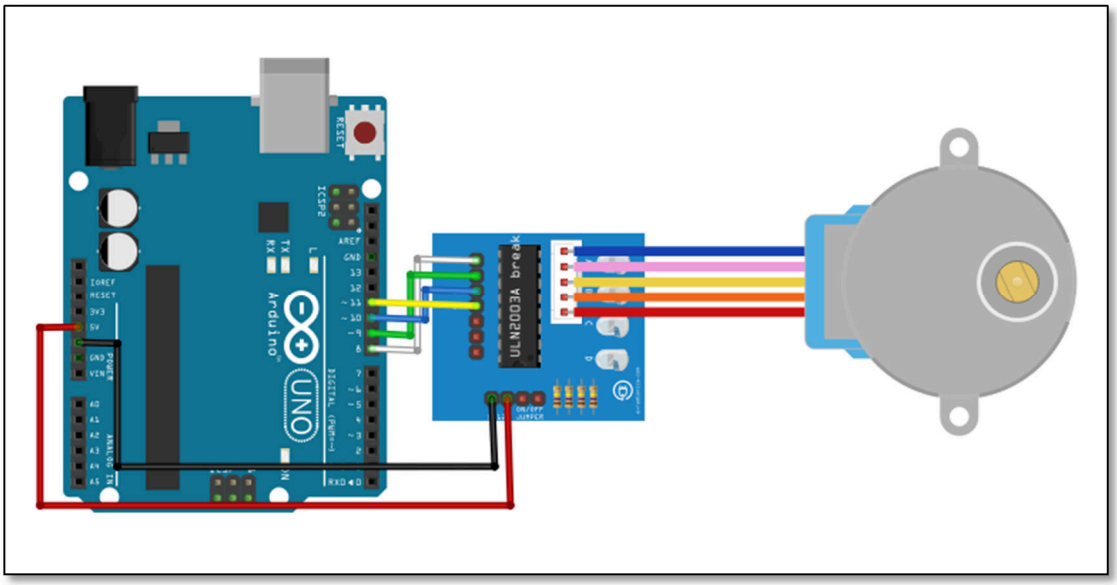
. 아두이노 우노(호환보드) X 1

. 5V 28BYJ48 Step Motor X 1

. ULN2003 X 1

. 점퍼와이어

- 연결도



- 코딩

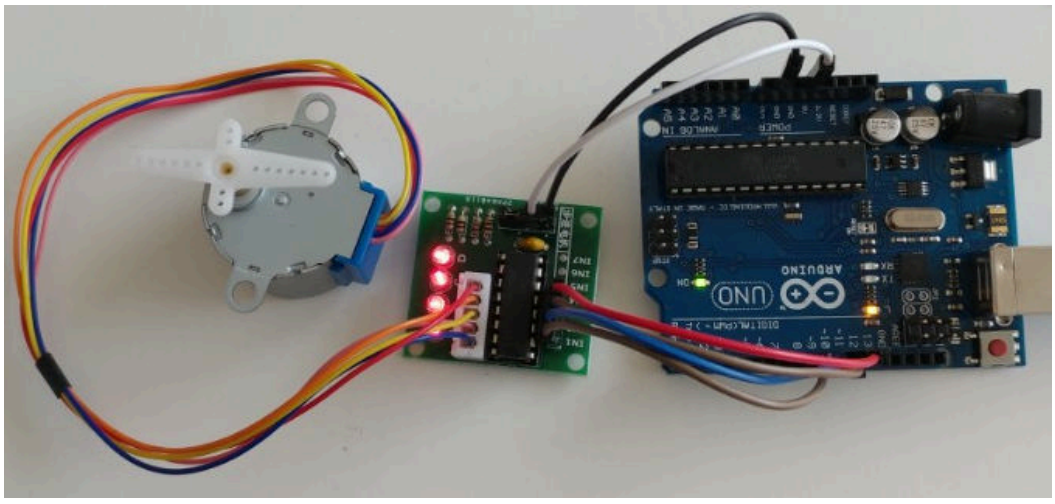
```
//**************************************************************************
// Project : Step Motor for led  in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//**************************************************************************
#define IN1  8
#define IN2  9
#define IN3  10
#define IN4  11
int Steps = 0;
boolean Direction = true;// gre
unsigned long last_time;
unsigned long currentMillis ;
int steps_left=4095;
long time;
void setup()
```

```
{
    Serial.begin(9600);
    pinMode(IN1, OUTPUT);
    pinMode(IN2, OUTPUT);
    pinMode(IN3, OUTPUT);
    pinMode(IN4, OUTPUT);
}
void loop()
{
  while(steps_left>0){
    currentMillis = micros();
    if(currentMillis-last_time>=1000){
    stepper(1);
    time=time+micros()-last_time;
    last_time=micros();
    steps_left--;
    }
  }
  delay(2000);
  Direction=!Direction;
  steps_left=4095;
}
void stepper(int xw){
    for (int x=0;x<xw;x++){
    switch(Steps){
       case 0:
         digitalWrite(IN1, LOW);
         digitalWrite(IN2, LOW);
         digitalWrite(IN3, LOW);
         digitalWrite(IN4, HIGH);
       break;
       case 1:
         digitalWrite(IN1, LOW);
         digitalWrite(IN2, LOW);
         digitalWrite(IN3, HIGH);
         digitalWrite(IN4, HIGH);
       break;
       case 2:
         digitalWrite(IN1, LOW);
         digitalWrite(IN2, LOW);
         digitalWrite(IN3, HIGH);
         digitalWrite(IN4, LOW);
       break;
       case 3:
         digitalWrite(IN1, LOW);
         digitalWrite(IN2, HIGH);
         digitalWrite(IN3, HIGH);
         digitalWrite(IN4, LOW);
       break;
       case 4:
         digitalWrite(IN1, LOW);
         digitalWrite(IN2, HIGH);
         digitalWrite(IN3, LOW);
         digitalWrite(IN4, LOW);
       break;
       case 5:
         digitalWrite(IN1, HIGH);
         digitalWrite(IN2, HIGH);
         digitalWrite(IN3, LOW);
         digitalWrite(IN4, LOW);
       break;
         case 6:
         digitalWrite(IN1, HIGH);
```

```
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, LOW);
      break;
      case 7:
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, HIGH);
      break;
      default:
        digitalWrite(IN1, LOW);
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, LOW);
      break;
   }
  SetDirection();
  }
}
void SetDirection(){
  if(Direction==1){ Steps++;}
  if(Direction==0){ Steps--; }
  if(Steps>7){Steps=0;}
  if(Steps<0){Steps=7; }
}
```

- **결과**

전원을 인가하면 좌, 우로 회전하는 것을 확인할 수 있습니다..

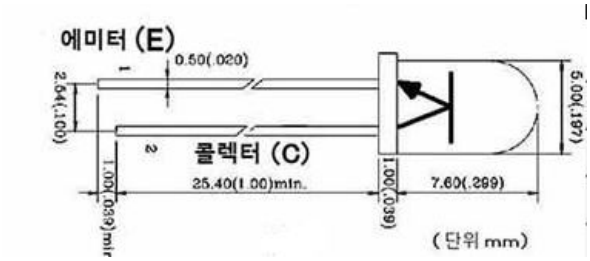


○ 불꽃감지센서

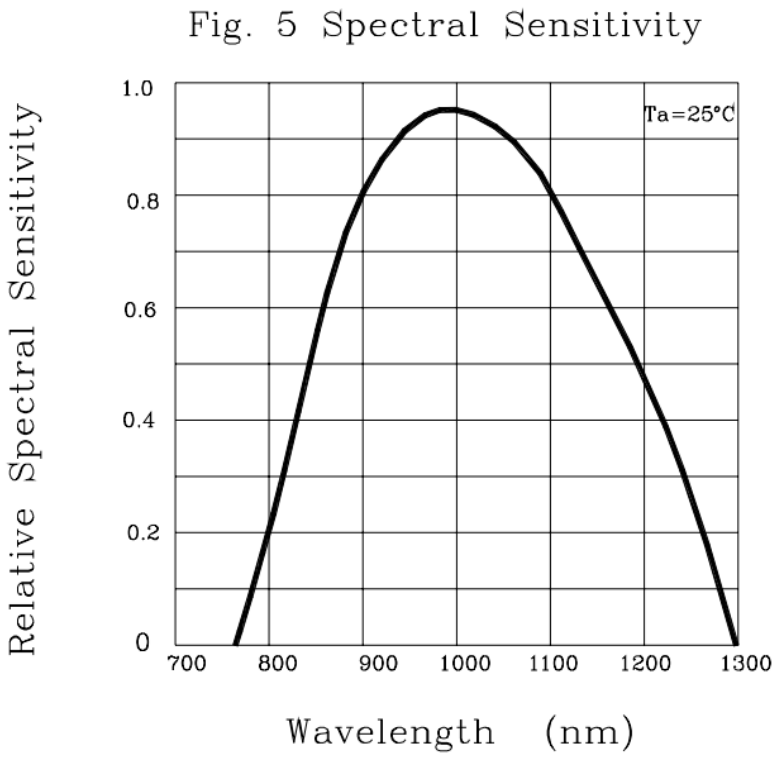
- 개요

불꽃감지센서 (Flame sensor)에 대해서 알아 보겠습니다.





불꽃 또는 화염은 사람의 눈으로 확인이 할 수 없는 자외선과 적외선의 파장이 발생이 됩니다. 불꽃감지센서는 적외선 감지센서로 760 nm – 1100 nm 의 파장을 감지할 수 있습니다.



- **준비물**

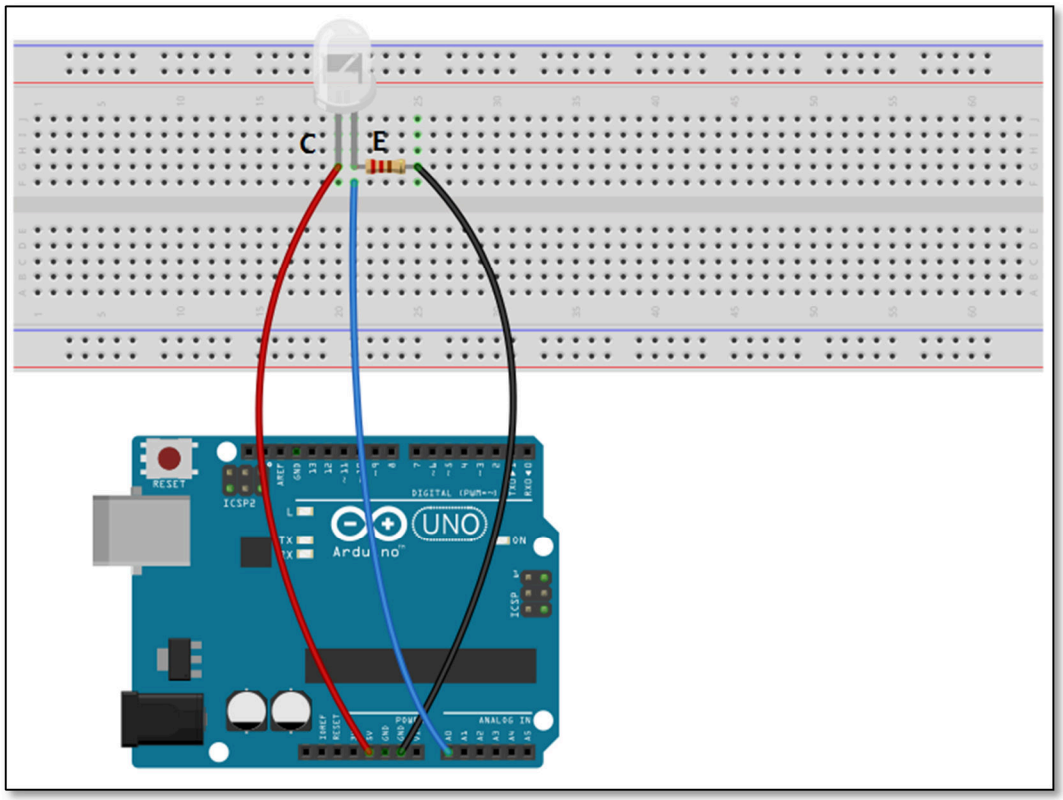
. 아두이노 우노(호환보드) X 1

. Flame sensor X 1

. 10K 저항 X 1

. 점퍼와이어

- **연결도**



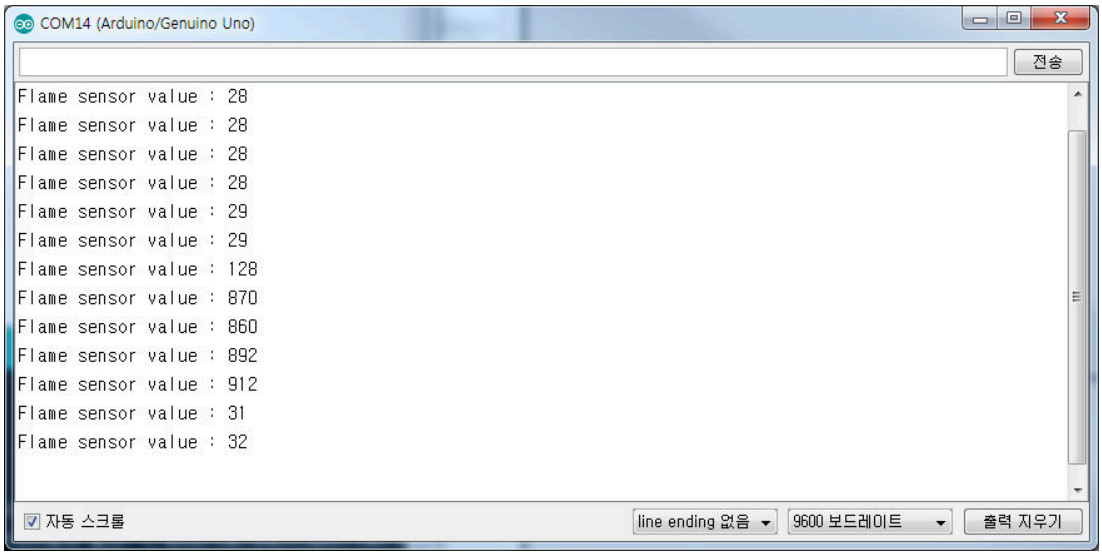
- **코딩**

```
//*************************************************************************
// Project : flame sensor  in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
#define SensorIn A0
void setup()
{
    Serial.begin(9600);
```

```
}
void loop()
{
    int R = analogRead(SensorIn); //Read Sensor 값
    Serial.print("Frame sensor value : ");
    Serial.println(R);    //시리얼 모니터에 출력
    delay(1000);
}
```

- **결과**

불꽃이 인식되면 값이 상승하는 것을 확인할 수 있습니다.

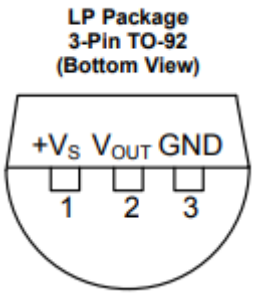


○ LM35 온도센서

- 개요

LM35 온도센서에 대해서 알아 보겠습니다.





LM35 온도센서는 온도를 측정하여 아날로그 신호를 출력하는 센서입니다.

측정할 수 있는 온도범위는 -55 에서 150 도까지이며 0.5 도의 정확성을 보증하면서 1 도당 10mV 전압 변동을 가집니다.

Table 1. Design Parameters

PARAMETER	VALUE
Accuracy at 25°C	±0.5°C
Accuracy from –55 °C to 150°C	±1°C
Temperature Slope	10 mV/°C

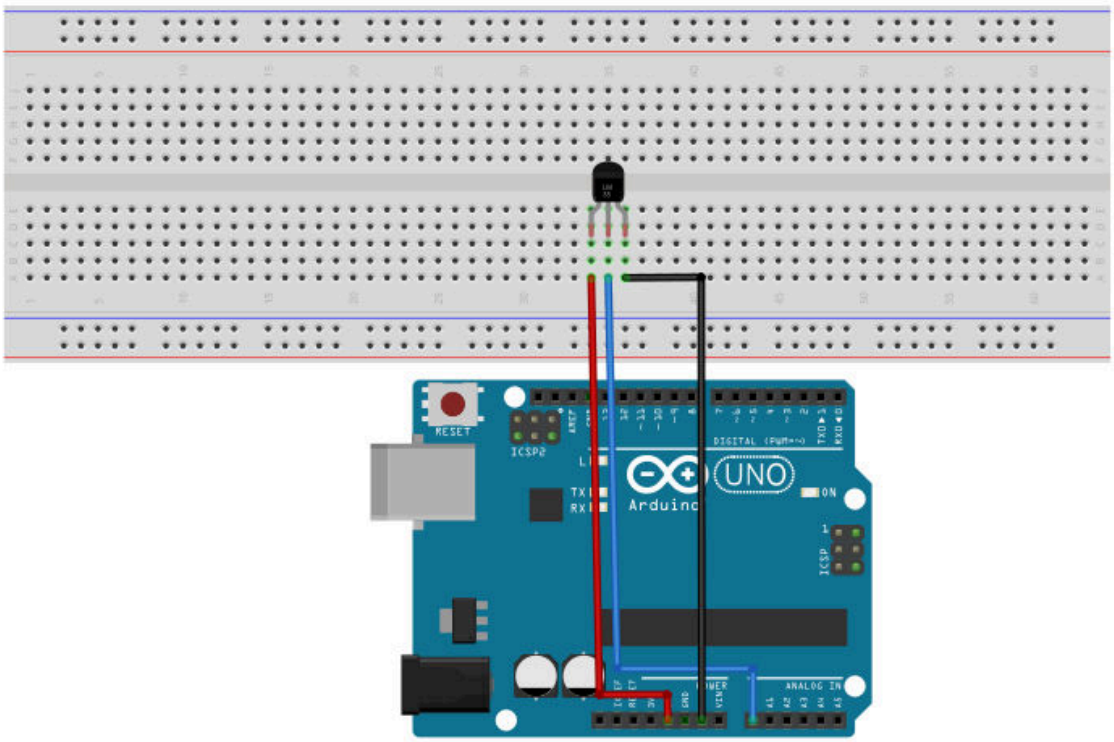
- 준비물

. 아두이노 우노(호환보드) X 1

. LM35 X 1

. 점퍼와이어

- 연결도

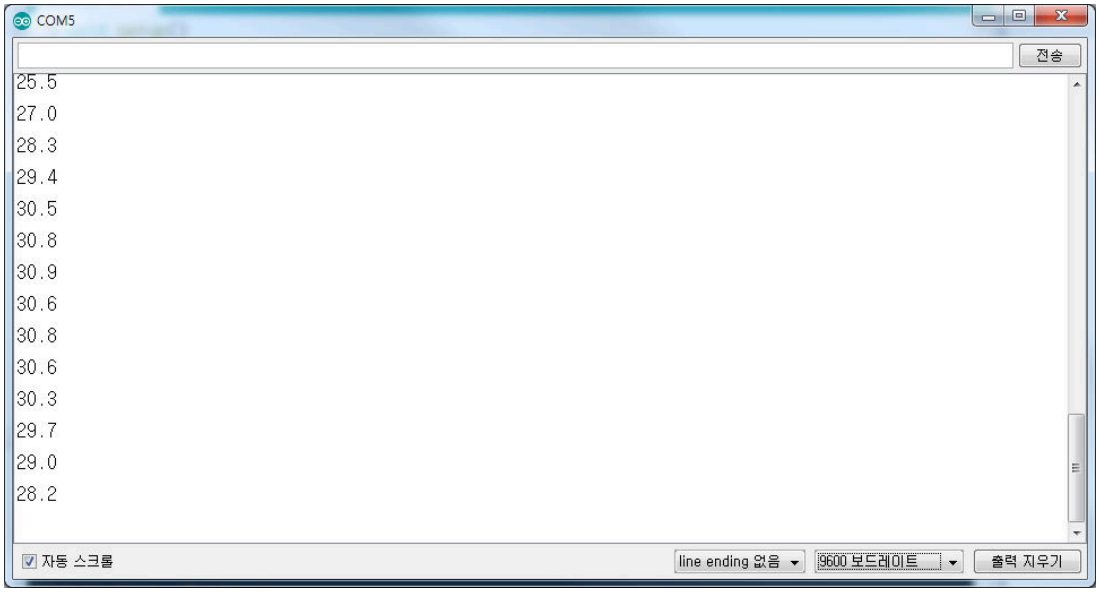


- 코딩

```
//********************************************************************************
// Project : LM35  sensor  in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//********************************************************************************
int analogPin = 0; // Analog pin
int val = 0; // variable
float temp =0;
void setup()
{
  Serial.begin(9600);        // setup serial
  analogReference(INTERNAL); // Internal voltage reference
}
void loop()
{
  val = analogRead(analogPin); // Read ADC
  val = map(val,0,1023,0,1100);
  temp=(float)val/10;
  Serial.println(temp,1); // Output
  delay(500);
}
```

- 결과

시리얼 모니터로 온도 변화를 확인할 수 있습니다.



○ NTC 서미스터

- 개요

NTC 서미스터 온도센서에 대해서 알아 보겠습니다.



서미스터(thermistor)란 저항의 종류로, 온도에 따라 물질의 저항 성질이 변화하는 전자 부품입니다. 또는 열가변저항기라고도 합니다. 주요 용도는 가정용으로는 일반 온도계, 산업용으로는 회로의 전류가 일정 이상으로 오르는 것을 방지하거나, 회로의 온도를 감지하는 센서로도 이용할 수 있습니다. 출력되는 신호는 아날로그 신호 형태입니다.

만약 NTC 서미스터의 파트넘버가 mf5a-3-10k 일 경우, 10K 라는 의미는 25'c 상온에서 저항값이 10K 값을 가진다는 것을 의미합니다. 그러므로 이전에 설명 드린 조도센서에서의 저항 분배방식을 사용해서 온도 값을 확인할 수 있습니다.

다음 표는 온도에 따른 NTC 서미스터 저항값의 변화입니다. 온도가 낮아지면 서미스터 저항 값을 하락하는 것을 확인할 수 있습니다.

**키트에 포함되어있는 서미스터의 R25 값은 10K 또는 47K 입니다.*

아래 표는 10k 와 47K NTC 저항의 온도에 따른 기준 표입니다.

10K-3 Thermistor Output Table

°F	°C	Ohms	°F	°C	Ohms	°F	°C	Ohms
-39	-39.44	232032	37	2.78	25948	113	45.00	4656
-37	-38.33	217394	39	3.89	24670	115	46.11	4473
-35	-37.22	203774	41	5.00	23462	117	47.22	4298
-33	-36.11	191093	43	6.11	22320	119	48.33	4131
-31	-35.00	179281	45	7.22	21241	121	49.44	3971
-29	-33.89	168275	47	8.33	20220	123	50.56	3817
-27	-32.78	158013	49	9.44	19254	125	51.67	3671
-25	-31.67	148442	51	10.56	18332	127	52.78	3532
-23	-30.56	139511	53	11.67	17467	129	53.89	3398
-21	-29.44	131100	55	12.78	16648	131	55.00	3271
-19	-28.33	123317	57	13.89	15872	133	56.11	3149
-17	-27.22	116045	59	15.00	15136	135	57.22	3032
-15	-26.11	109247	61	16.11	14439	137	58.33	2920
-13	-25.00	102889	63	17.22	13778	139	59.44	2812
-11	-23.89	96941	65	18.33	13151	141	60.56	2709
-9	-22.78	91374	67	19.44	12556	143	61.67	2610
-7	-21.67	86160	69	20.56	11987	145	62.78	2516
-5	-20.56	81276	71	21.67	11451	147	63.89	2425
-3	-19.44	76659	73	22.78	10942	149	65.00	2339
-1	-18.33	72371	75	23.89	10459	151	66.11	2256
1	-17.22	68348	77	25.00	10000	153	67.22	2176
3	-16.11	64574	79	26.11	9564	155	68.33	2099
5	-15.00	61031	81	27.22	9149	157	69.44	2026
7	-13.89	57703	83	28.33	8754	159	70.56	1955
9	-12.78	54578	85	29.44	8379	161	71.67	1887
11	-11.67	51641	87	30.56	8019	163	72.78	1822
13	-10.56	48879	89	31.67	7679	165	73.89	1760
15	-9.44	46259	91	32.78	7355	167	75.00	1700
17	-8.33	43817	93	33.89	7047	169	76.11	1642
19	-7.22	41519	95	35.00	6754	171	77.22	1587
21	-6.11	39354	97	36.11	6474	173	78.33	1534
23	-5.00	37316	99	37.22	6208	175	79.44	1483
25	-3.89	35395	101	38.33	5954	177	80.56	1433
27	-2.78	33585	103	39.44	5712	179	81.67	1386
29	-1.67	31878	105	40.56	5479	181	82.78	1341
31	-0.56	30267	107	41.67	5258	183	83.89	1297
33	0.56	28735	109	42.78	5048	185	85.00	1255
35	1.67	27302	111	43.89	4847	187	86.11	1214

47K Thermistor Output Table

°F	°C	Ohms
-39	-39.44	1531956
-37	-38.33	1425406
-35	-37.22	1326920
-33	-36.11	1235843
-31	-35.00	1151576
-29	-33.89	1073571
-27	-32.78	1001327
-25	-31.67	934386
-23	-30.56	872330
-21	-29.44	814274
-19	-28.33	760902
-17	-27.22	711353
-15	-26.11	665333
-13	-25.00	622569
-11	-23.89	582814
-9	-22.78	545838
-7	-21.67	511432
-5	-20.56	479403
-3	-19.44	449314
-1	-18.33	421538
1	-17.22	395646
3	-16.11	371501
5	-15.00	348974
7	-13.89	327948
9	-12.78	308315
11	-11.67	289975
13	-10.56	272835
15	-9.44	256671
17	-8.33	241692
19	-7.22	227677
21	-6.11	214557
23	-5.00	202271
25	-3.89	190762
27	-2.78	179975
29	-1.67	169862
31	-0.56	160377
33	0.56	151399
35	1.67	143050
37	2.78	135211
39	3.89	127847
41	5.00	120927
43	6.11	114422
45	7.22	108304
47	8.33	102549
49	9.44	97134
51	10.56	91991
53	11.67	87192
55	12.78	82672
57	13.89	78412
59	15.00	74395
61	16.11	70608
63	17.22	67035
65	18.33	63663
67	19.44	60480
69	20.56	57448
71	21.67	54611
73	22.78	51929
75	23.89	49395
77	25.00	46998
79	26.11	44731
81	27.22	42586
83	28.33	40557
85	29.44	38635
87	30.56	36799
89	31.67	35076
91	32.78	33443
93	33.89	31895
95	35.00	30428
97	36.11	29036
99	37.22	27715
101	38.33	26462
103	39.44	25272
105	40.56	24133
107	41.67	23061
109	42.78	22042
111	43.89	21073
113	45.00	20153
115	46.11	19278
117	47.22	18445
119	48.33	17653
121	49.44	16899
123	50.56	16175
125	51.67	15493
127	52.78	14842
129	53.89	14223
131	55.00	13632
133	56.11	13070
135	57.22	12533
137	58.33	12021
139	59.44	11533
141	60.56	11064
143	61.67	10620
145	62.78	10196
147	63.89	9791
149	65.00	9404
151	66.11	9035
153	67.22	8682
155	68.33	8345
157	69.44	8022
159	70.56	7711
161	71.67	7416
163	72.78	7135
165	73.89	6865
167	75.00	6607
169	76.11	6359
171	77.22	6123
173	78.33	5896
175	79.44	5679
177	80.56	5469
179	81.67	5270
181	82.78	5079
183	83.89	4896
185	85.00	4721
187	86.11	4552

- 준비물

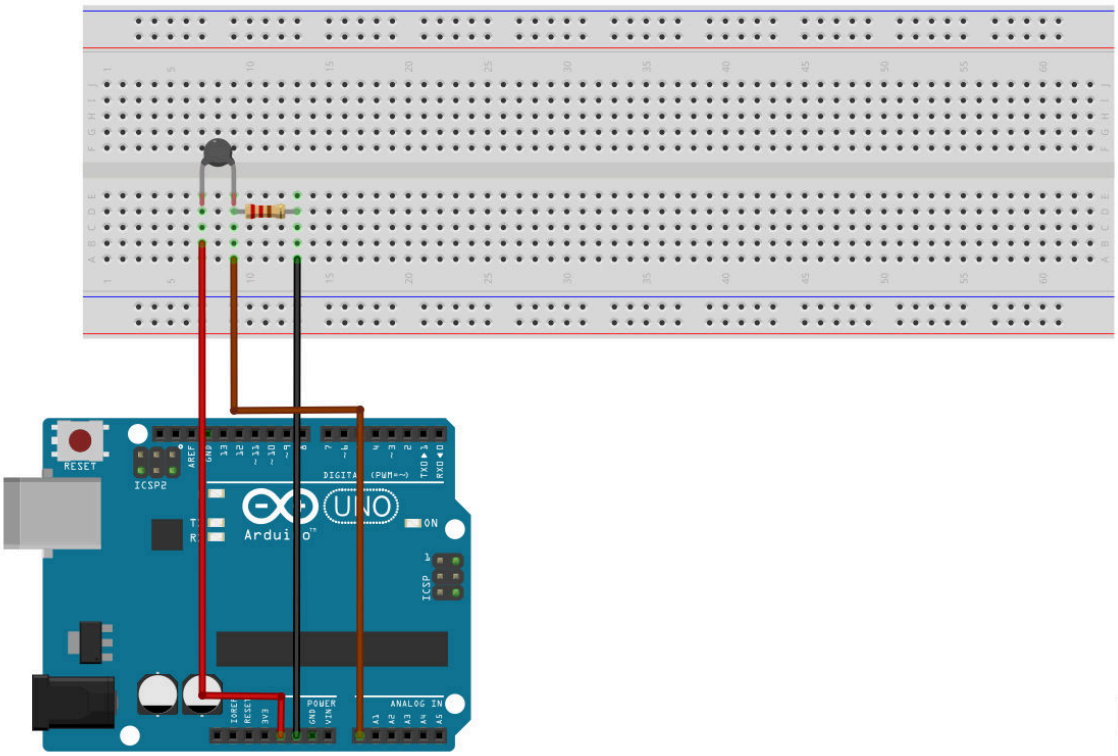
. 아두이노 우노(호환보드) X 1

. NTC 서미스터 X 1

. 10K 막대저항 X 1

. 점퍼와이어

- 연결도



- 코딩

```
//*************************************************************************
// Project : 26_NTC in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
```

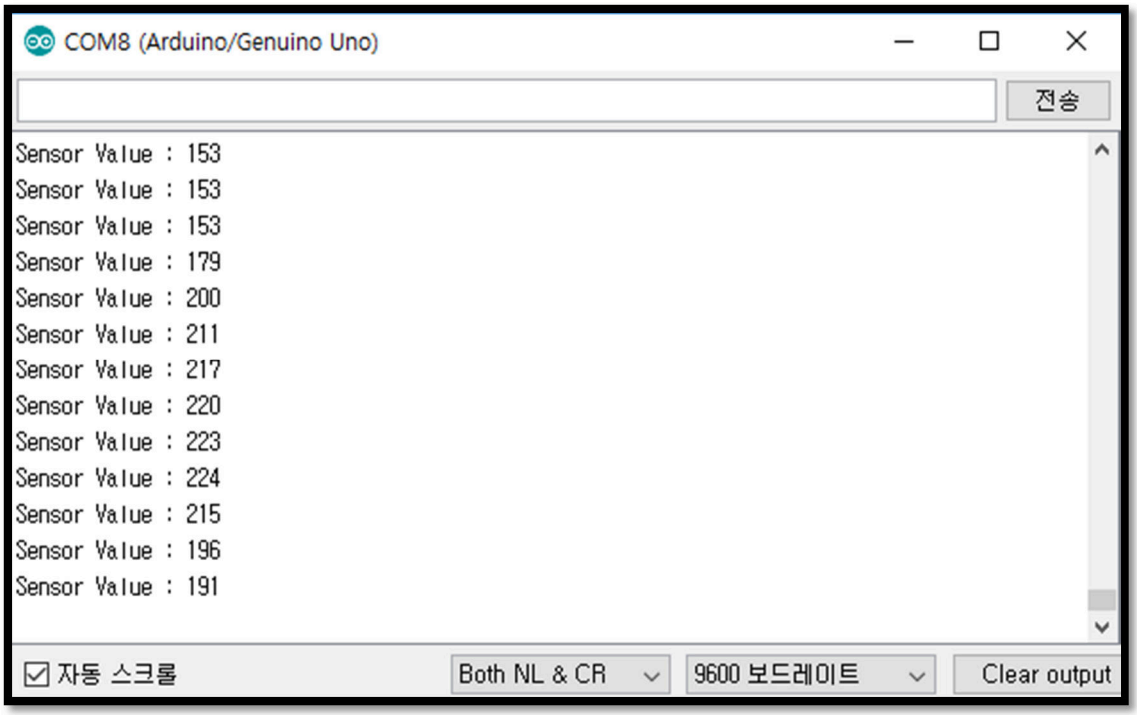
```
{
  pinMode (LedPin, OUTPUT);
//  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
void loop ()
{
  gValue = analogRead(AnalogPin);

  if (gValue>180)   {
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

- **결과**

시리얼 모니터로 온도에 따른 아날로그 변화를 확인할 수 있습니다.

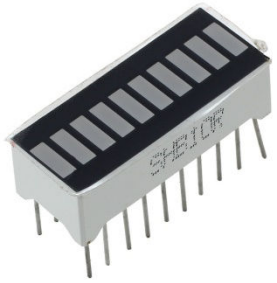
출력되는 아날로그값을 온도로 변화하기 위해서는 전압분배 방식에 의한 계산과 위에서 말씀 NTC 저항 Table 을 비교하는 알고리즘을 추가하셔야 합니다.



○ LED BAR 세그먼트

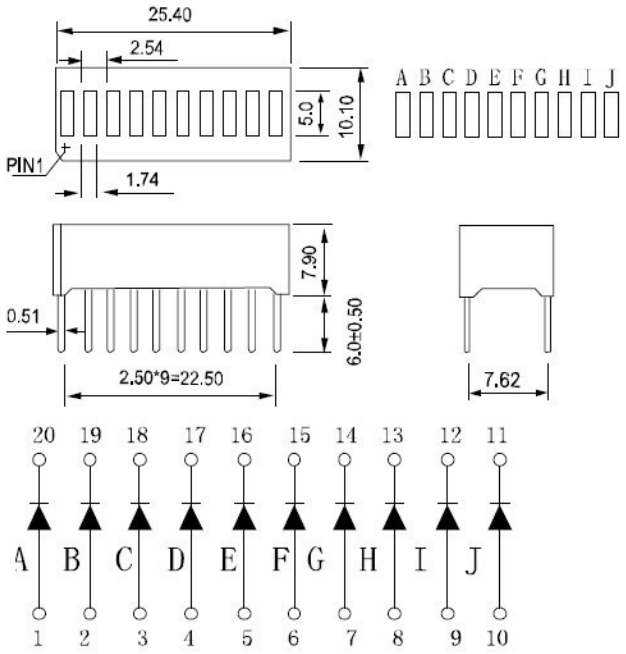
- 개요

10 bar 세그먼트 LED 대해서 알아 보겠습니다.



10 bar segment LED 는 10 개의 LED 를 Bar 형태로 합친 표시장치입니다. 주요 활용 용도는 배터리 잔량 표시, 전력량계 등 다양한 그래픽 형태의 값을 표시할 수 있습니다. (조도센서를 활용한 밝기값, 미세먼지 수치값 등)

내부 구조는 다음 그림과 같이 되어있습니다.



- **준비물**

. 아두이노 우노(호환보드) X 1

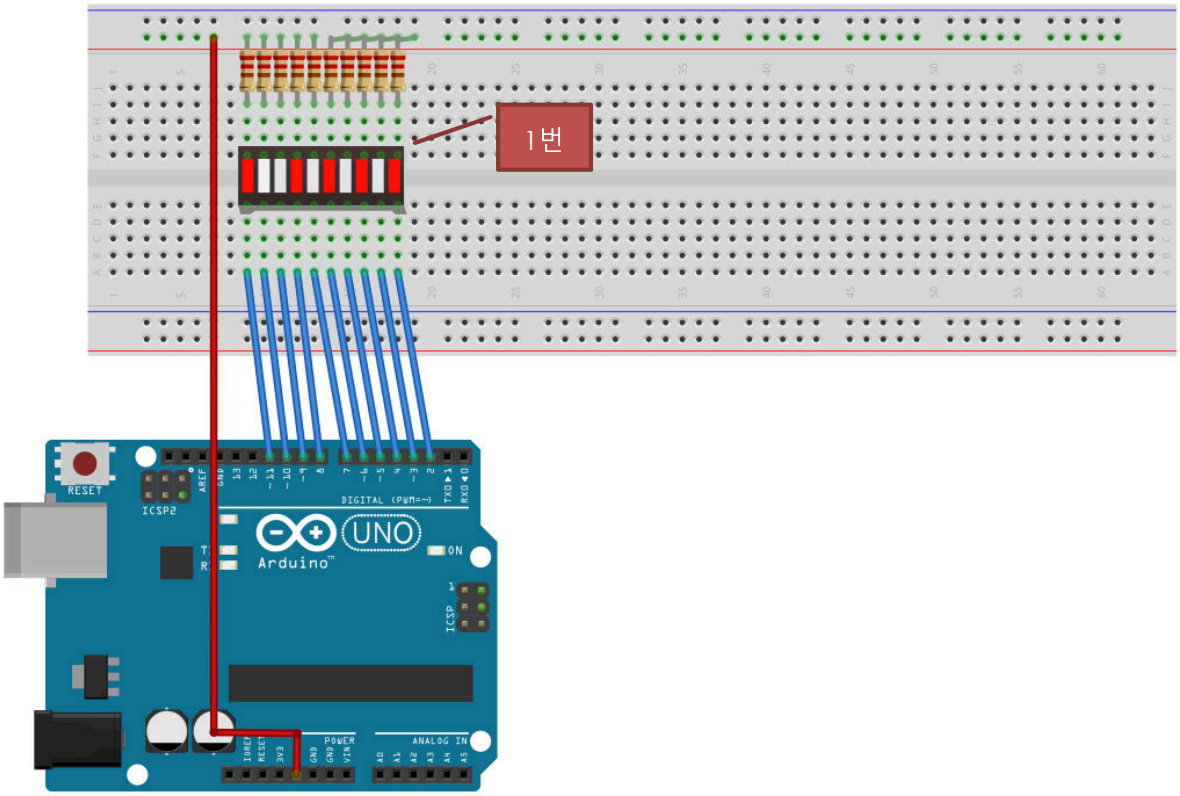
. BAR LED X 1

. 220 옴 막대저항 X 10

. 점퍼와이어

- **연결도**

* 위 사진의 내부그림과 같이 1 번 핀은 작은 홈이 표시 되어있습니다.



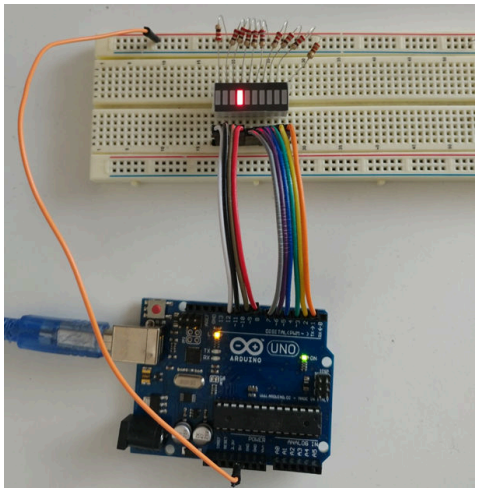
- **코딩**

```
//*************************************************************************
```

```
// Project : 27_LED bar in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin2 = 2;     // select the pin for the LED
int LedPin3 = 3;     // select the pin for the LED
int LedPin4 = 4;     // select the pin for the LED
int LedPin5 = 5;     // select the pin for the LED
int LedPin6 = 6;     // select the pin for the LED
int LedPin7 = 7;     // select the pin for the LED
int LedPin8 = 8;     // select the pin for the LED
int LedPin9 = 9;     // select the pin for the LED
int LedPin10 = 10;     // select the pin for the LED
int LedPin11 = 11;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  int i=0;
  for(i=0 ; i<10 ; i++)
    pinMode (LedPin2+i, OUTPUT);
  Serial.begin (9600);
}
void loop ()
{
  int i=0;
  static int j=2;
  // clear all led
  for(i=0;i<10;i++)
    digitalWrite (LedPin2+i, HIGH);
  // turn on, shift led
  digitalWrite (j, LOW);
  j++;
  if(j>11)  j=0;
  delay(500);
}
```

- 결과

LED 가 하나씩 Shift 되면서 켜지는 것을 확인할 수 있습니다.



○ TCRT5000 근접센서

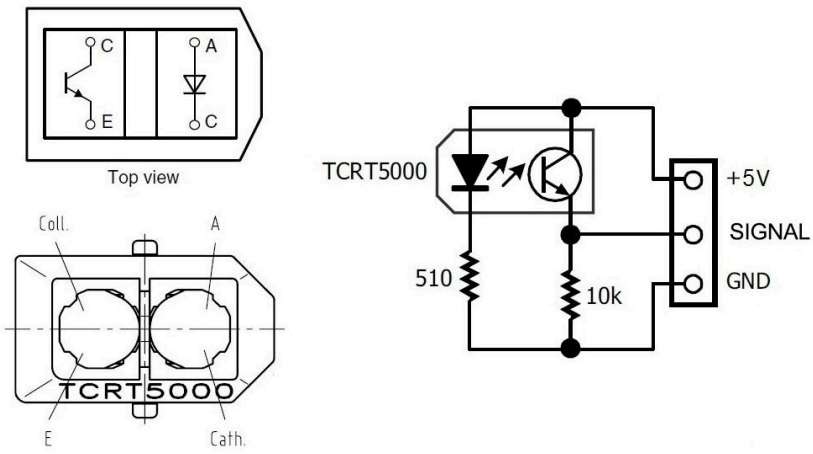
- 개요

TCRT5000 근접센서에 대해서 알아 보겠습니다.



TCRT5000 는 적외선 수/발광 센서가 내장된 적외선 센서입니다. 적외선 수신부와 발신부를 통해 약 2cm 이내의 근접 물체를 감지 하거나 색상을 구분할 수 있으며, 주로 라인트레이서를 만드는데 사용할 수 있습니다. 구동 원리는 적외선을 발신하고 적외선을 흡수하는 색과 흡수하지 않는 색으로 구분하는 것입니다.

내부 구조는 다음 그림과 같이 되어있습니다.



- 준비물

. 아두이노 우노(호환보드) X 1

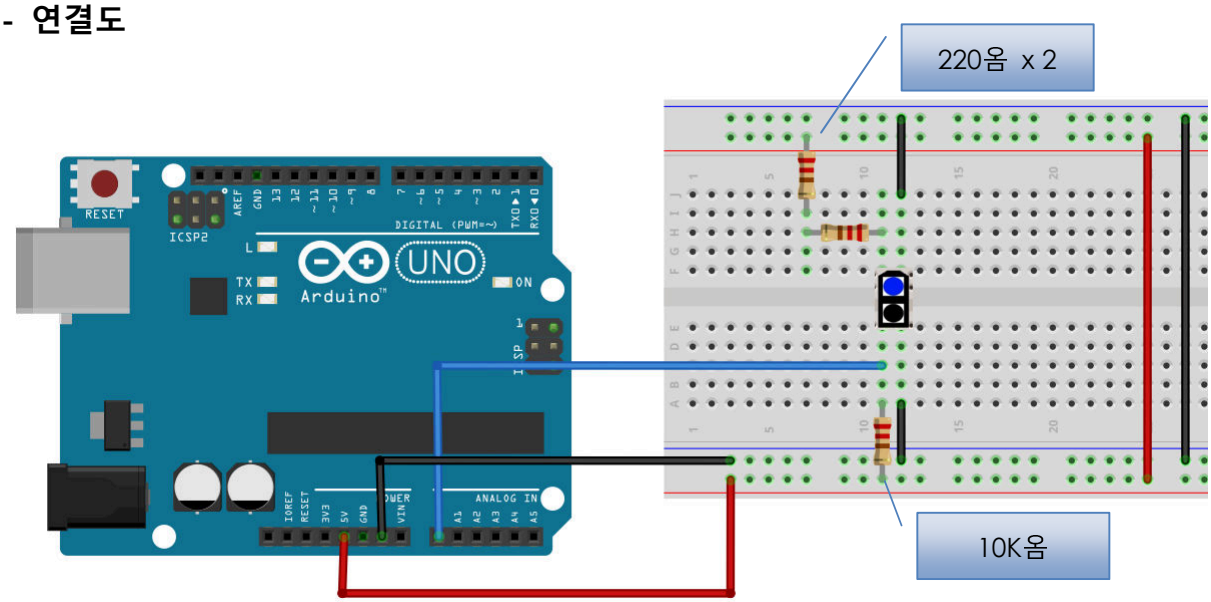
. TCRT5000 X 1

. 220 옴 막대저항 X 2

. 10K 옴 막대저항 X 1

. 점퍼와이어

- 연결도



- 코딩

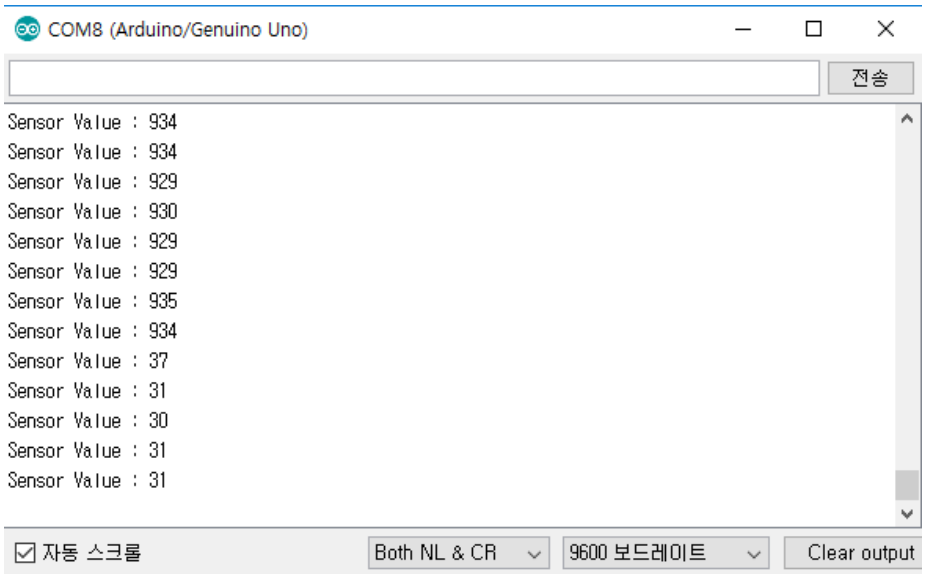
```
//*****************************************************************************
// Project : 28_TCRT5000 in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
//  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
void loop ()
```

```
{
  gValue = analogRead(AnalogPin);

  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

- **결과**

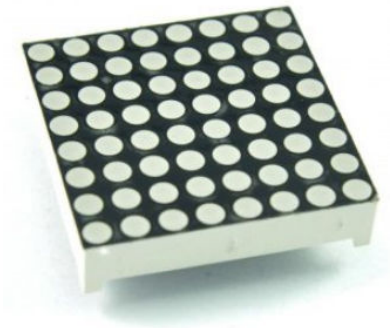
같은 색상의 거리에 따라 또는 색상에 따라 값이 다르게 출력되는 것을 확인할 수 있습니다.



○ 도트메트릭스

- 개요

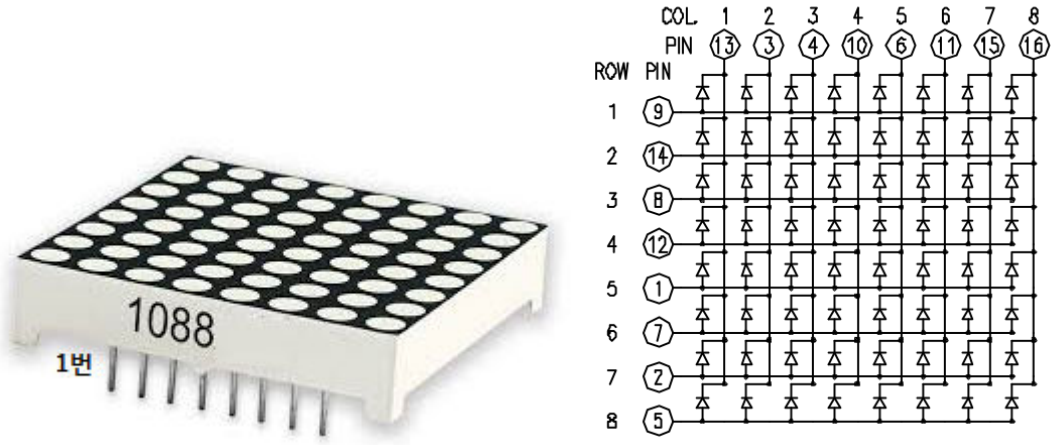
도트매트릭스에 대해서 알아 보도록 하겠습니다.



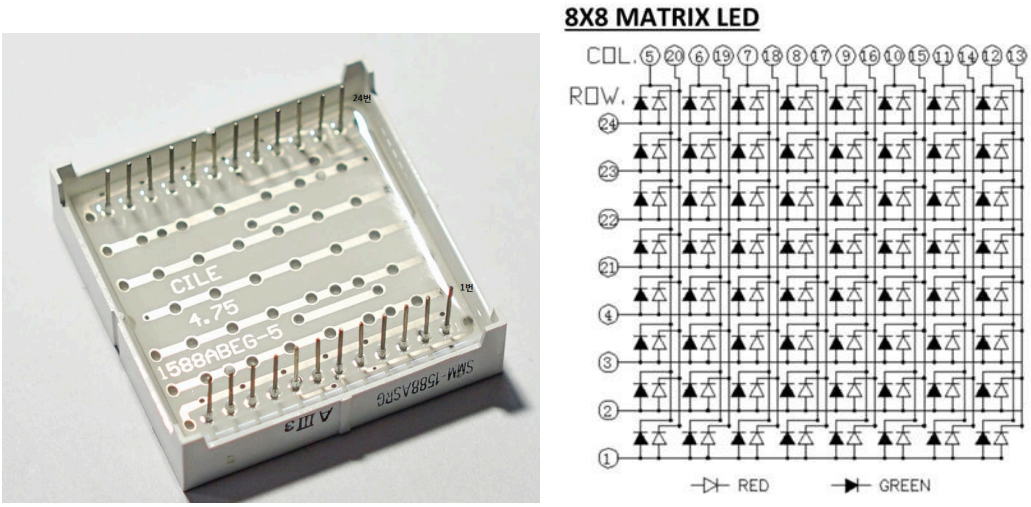
도트매트릭스란 LED 의 집합형태로 다양한 문자와 패턴 등을 표시해는 장치입니다. 8x8, 16x16 여러 모양이 있으며 광고/안내판 등에 널리 사용되고 이는 장치입니다.

내부는 4x4 키패드와 비슷한 구조로 LED 8x8 총 64 개(또는 128 개)의 LED 들이 matrix 형태로 연결이 되어있습니다.

Shift IC(74HC595) 없이 직접 사용하기 위해서는 한 개 컬러 도트매트릭스의 경우, 총 16 개의 디지털 출력이 필요하고 두 개의 컬러 도트매트릭스를 사용하기 위해서는 총 24 개의 디지털 출력이 필요합니다.



[One Color 8x8 dot matrix]



[Two Color(RED/GREEN) 8x8 dot matrix]

- **준비물**

. 아두이노 우노(호환보드) X 1

. 8x8 도트매트릭스 x 1

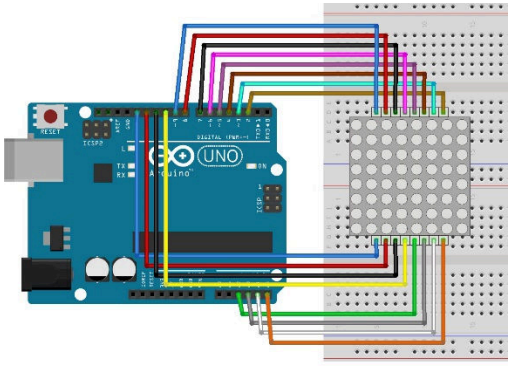
. 브래드보드 x 1

. 점퍼와이어

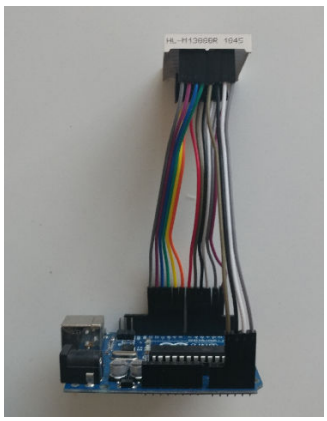
- **연결도**

도트매트릭스의 1 번핀은 실크인쇄 된 면에 좌측 첫 번째 핀입니다.

연결 방법은 아래와 같이 브래드보드를 사용하시거나 점퍼연결선을 직접 연결하시면 됩니다.(핀맵은 동일)



[브래드보드 활용 예]



[직접 연결 예]

- 코딩

아래 코드는 도트매트릭스와 연결된 아두이노 핀맵 정보를 R[]와 C[] 행렬에 저장을 하고, 화면에 표시할 데이터를 biglove 또는 smalllvoe 에 저장한 다음 Display 함수에서 빠르게 LED 를 On/Off 하는 것입니다. (잔상효과)

```
//*************************************************************************
// Project : Dot matrix  in Learing Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
// 2-dimensional array of row pin numbers:
int R[] = {2,7,A5,5,13,A4,12,A2};
// 2-dimensional array of column pin numbers:
int C[] = {6,11,10,3,A3,4,8,9};

unsigned char biglove[8][8] =     //the big "heart"
{
  0,0,0,0,0,0,0,0,
  0,1,1,0,0,1,1,0,
  1,1,1,1,1,1,1,1,
  1,1,1,1,1,1,1,1,
  1,1,1,1,1,1,1,1,
  0,1,1,1,1,1,1,0,
  0,0,1,1,1,1,0,0,
  0,0,0,1,1,0,0,0,
};

unsigned char smalllove[8][8] =      //the small "heart"
{
  0,0,0,0,0,0,0,0,
  0,0,0,0,0,0,0,0,
  0,0,1,0,0,1,0,0,
  0,1,1,1,1,1,1,0,
  0,1,1,1,1,1,1,0,
  0,0,1,1,1,1,0,0,
  0,0,0,1,1,0,0,0,
  0,0,0,0,0,0,0,0,
};

void setup()
{
   // iterate over the pins:
  for(int i = 0;i<8;i++)
  // initialize the output pins:
  {
    pinMode(R[i],OUTPUT);
    pinMode(C[i],OUTPUT);
  }
}

void loop()
{
  for(int i = 0 ; i < 100 ; i++)       //Loop display 100 times
  {
    Display(biglove);                  //Display the "Big Heart"
  }
  for(int i = 0 ; i < 50 ; i++)        //Loop display 50 times
  {
    Display(smalllove);                //Display the "small Heart"
```

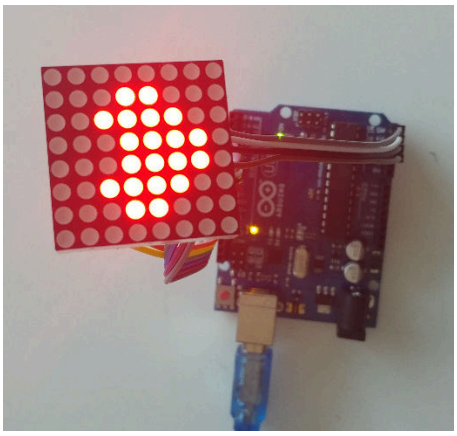
```
  }
}

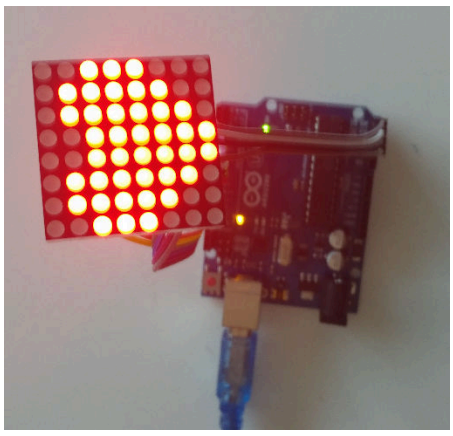
void Display(unsigned char dat[8][8])
{
  for(int c = 0; c<8;c++)
  {
    digitalWrite(C[c],LOW);//use thr column
    //loop
    for(int r = 0;r<8;r++)
    {
      digitalWrite(R[r],dat[r][c]);
    }
    delay(1);
    Clear();  //Remove empty display light
  }
}

void Clear()                          // Display Clear
{
  for(int i = 0;i<8;i++)
  {
    digitalWrite(R[i],LOW);
    digitalWrite(C[i],HIGH);
  }
}
```

- **결과**

큰 하트 모양과 작은 하트 모양이 교대로 표시되는 것을 확인 할 수 있습니다.





- 8x8 도트 매트릭스 아두이노 사용법

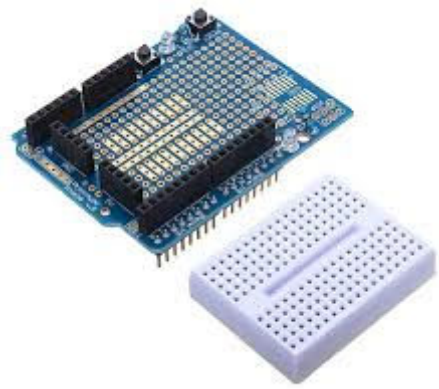
https://www.arduino.cc/en/Tutorial/RowColumnScanning

http://osoyoo.com/2017/07/15/arduino-lesson-8x8-led-matrix/

○ 아두이노 프로토 쉴드

- 개요

아두이노 프로토 쉴드(Arduino proto shield)에 대해서 알아 보겠습니다.



[프로토 쉴드]



[아두이노 연결]

아두이노 프로토 쉴드는 프로토타이핑을 용이하게 하기 위해서 디자인 된 제품입니다. 외부 크기와 커넥터 위치가 아두이노 우노와 동일하므로 위/아래로 쉽게 연결해서 사용할 수 있습니다. 간단히 부품을 납땜 할 수도 있고, 미니 브래드 보드를 사용할 수도 있습니다. 다만 프로토 쉴드 위에 직접 납땜을 할 경우 재사용이 어려울 수도 있습니다.

특징

- Arduino 우노 Pinout 호환(우노보드와 연결가능)

- 2 x LEDs (wired-up to 2 resistors)

- 1 x Button, 1 x Reset

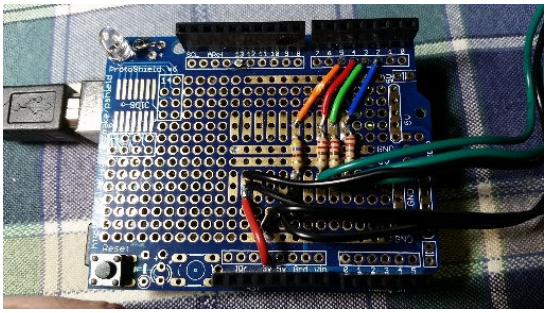
- 5 x extra 5V female pins, 5 x extra GND female pins

- 1 x 170 pin breadboard (separate piece)

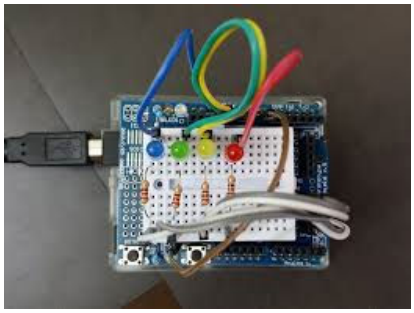
- 1 ICSP Connector footprint

- SPI signals pads

- 24 pin SMD footprint (50 mils pitch)



[납땜]



[미니브래드보드]

○ 초음파 센서

- 개요

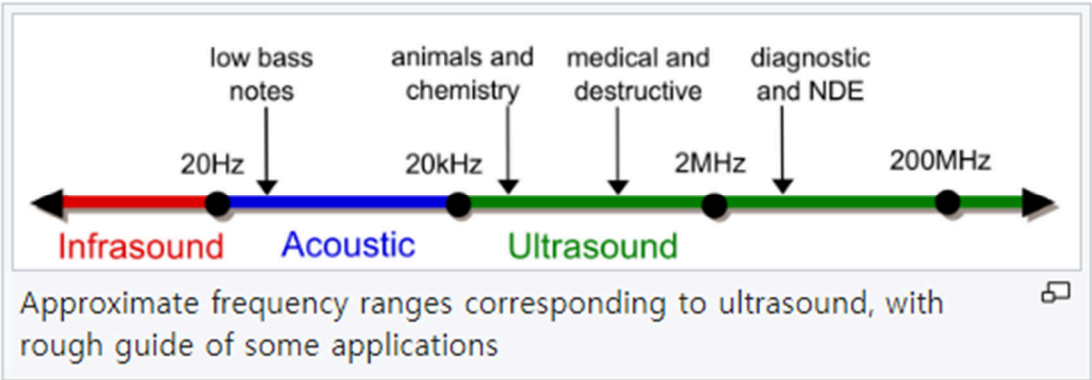


초음파는 영어로 Ultra Sonic 입니다.
위키백과의 내용을 보면 아래와 같이 설명합니다.

"초음파(超音波, 영어: ultrasound(s), ultrasonic wave)는 인간이 들을 수 있는 가청 최대 한계 범위를 넘어서는 주파수를 갖는 주기적인 '음압'(音壓, sound pressure)을 의미한다. 비록, 이러한 한계값이 사람마다 다르지만, 건강한 젊은 사람의 경우 이 값은 약 20kHz(20,000 Hz)이고 따라서 20kHz는 초음파를 설명하는 데 있어 유용한 하한(下限)이 된다."

인간의 가청 주파수 (들을 수 있는 소리)는 건강한 사람의 기준으로 20Khz 까지 입니다. 다시 말해 초음파는 가청 주파수 20Khz 를 이상 되고 주파수를 가지는 주기적인 음압을 의미합니다. 인간의 가청주파수는 20Khz 이지만 일부 동물들인 돌고래, 박쥐 등은 초음파를 발생시키기도 하고 들을 수도 있습니다. 특히 박쥐는 100khz 에서 200khz 까지도 감지가 가능합니다.

다음 그림은 일부 어플리케이션에 따른 주파수 범위 입니다.



Approximate frequency ranges corresponding to ultrasound, with rough guide of some applications

* 초음파 센서 거리 측정공식

초음파센서에서 거리를 측정하기 위한 파라미터 아래와 같습니다.

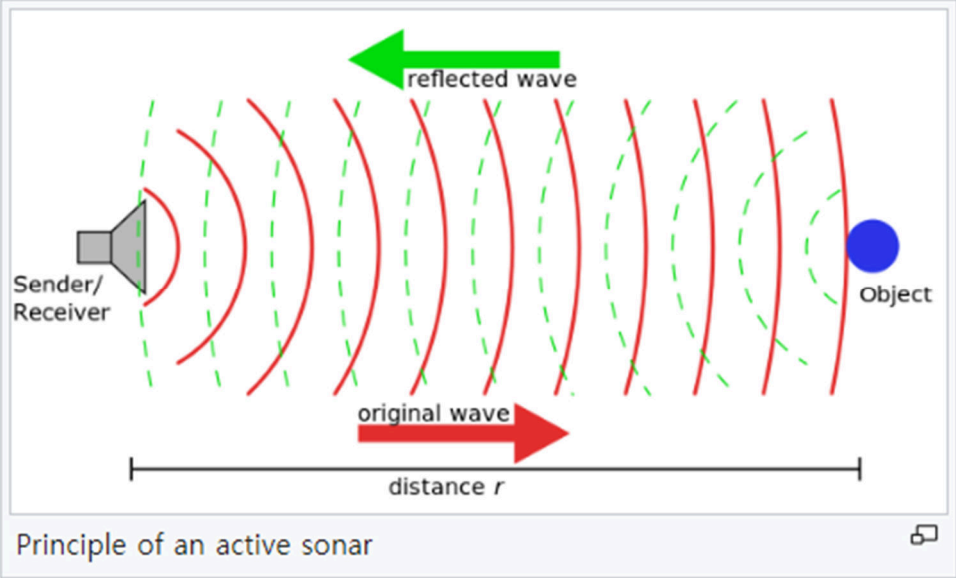
- 대상 물체까지의 거리 : L [m]
- 측정된 시간 : T [s]
- 음속 : V [m/s]
- 기온 : t [C]

거리는 L = (T * V) [m]

초음파는 소리와 같은 속도를 가집니다. 그러므로 초당 340m 를 전파할 수 있습니다. 하지만 반사되어 돌아오는 초음파의 파형은 상대 물질의 표면이나 종류에 따라 다를 수도 있고, 공기 온도에도 영향을 받을 수 있습니다.

아래 그림을 보시면 초음파센서 동작을 이해하기 쉽습니다.

즉, 우선 초음파를 보내고 되돌아 오는 초음파를 인식해서 시간을 거리로 환산하는 것입니다.

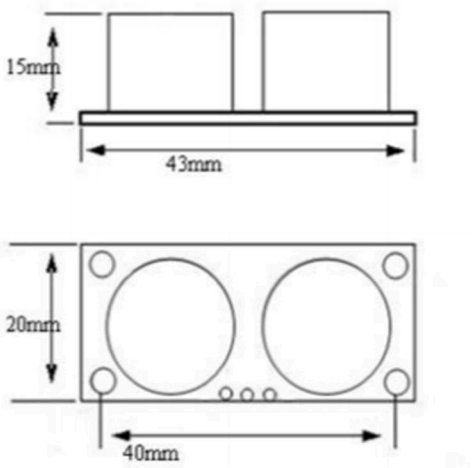


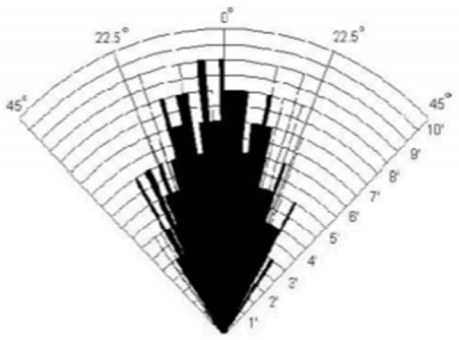
Principle of an active sonar

(출처 : https://en.wikipedia.org/wiki/Ultrasound)

초음파는 빛이 아니므로 소리와 동일하게 파장으로 퍼서 나갑니다.

HC-SR04 데이터시트를 보면, 아래와 같이 측정할 수 있는 최적의 각도를 30 도로 규정하고 있습니다.



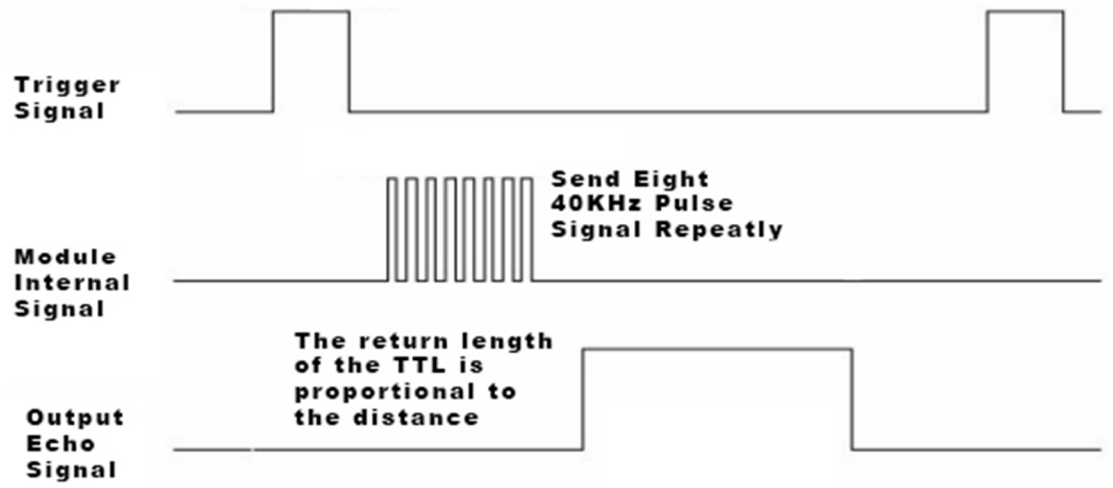


Practical test of performance,
Best in 30 degree angle

다음 그림은 인터페이스 신호들입니다.
Sensing 하는 순서는 위에서 아래로 생각하시면 됩니다.
즉, 아두이노에서 시작 Trigger 신호를 10ms 간격으로 센서로 전달하면,

초음파센서는 초음파를 발생시킵니다.

초음파센서는 물체의 표면에서 반사되어 돌아오는 초음파 응답 시간을 TTL 레벨 Echo 신호로 다시 아두이노로 전달합니다.
아두이노에서는 마지막 Echo 신호의 High 구간의 시간을 체크하면 됩니다.



SR 04 기본적인 사양은 다음과 같습니다.

1. 모듈명 : [CATALEX] Ultrasonic Ranger
2. 입력 전압 : 5V
3. 소비 전류 : 2mA
4. 측정 거리 : 2cm ~ 450cm
5. 측정 각도 : 최대 15° 이하

- 준비물

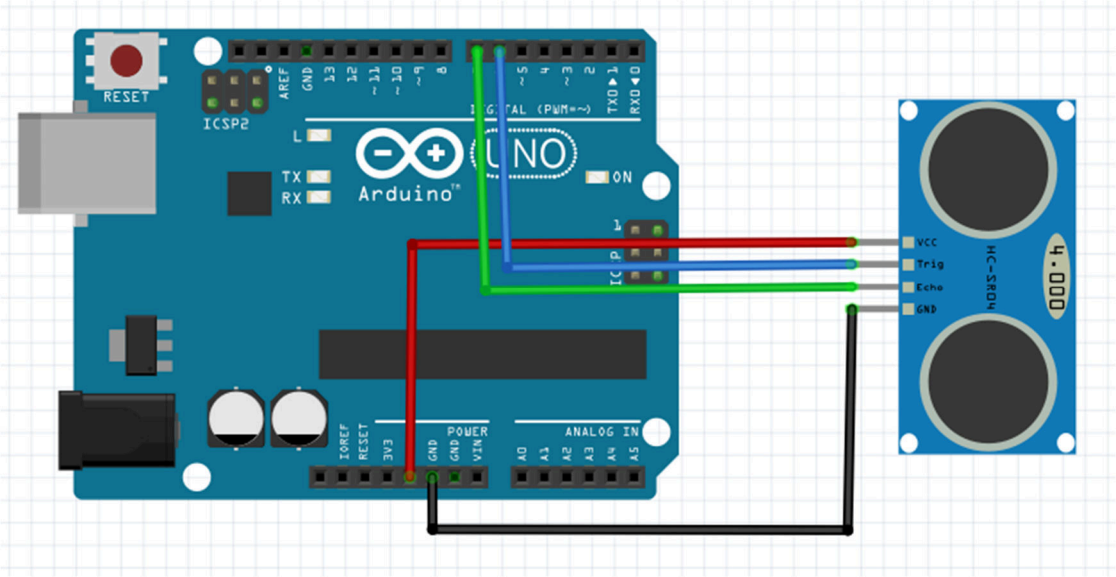
. 아두이노 우노(호환보드) X 1

. 초음파센서 X 1

. 점퍼와이어

- **연결도**

Trigger 와 Echo 를 디지털 핀에 연결합니다.



- **코딩**

다음 코드는 Setup()에서 입출력포트를 설정하고 Loop 에서 1 초 단위로 거리를 측정하는 코드입니다.

```
//*********************************************************************************
// Project : Arduino Uno R3 Ultra Sonic in Advanced set
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************

int echo = 7;
int trig = 6;
void setup() {
  Serial.begin(115200);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
}
void loop() {
  float Length, distance;
  digitalWrite(trig, HIGH);
  delay(10);
  digitalWrite(trig, LOW);

  Length = pulseIn(echo, HIGH);
  distance = ((float)(340 * Length) / 10000) / 2;
```

```
  Serial.print(distance);
  Serial.println(" Cm");
  delay(1000);
}
```

추가로 코드를 설명을 드리면

Setup 에서 Trigger 와 Echo 포트의 입출력을 설정합니다.
Loop 에서 Trigger 신호 10us 신호를 보내서 초음파센서의 초음파를 발생합니다. 초음파가 피드백되어 Puseln() 함수를 통해서 echo 가 high 가 되는 시간을 측정합니다.

* 잠시 pulsein() 함수에 대해서 설명을 드리면

pulsein()는 아두이노 기본 함수로 입력신호가 High 또는 Low 가 되는 시간을 측정해주는 함수입니다.

입력 파라미터를 아래와 같이 2 개 또는 3 개를 사용할 수 있습니다.

pulseIn(pin, value);
pulseIn(pin, value, timeout);

파라미터를 두 개를 사용할 경우 입력핀과 pulse 상태(High, Low) 를 입력합니다. 파라미터를 세 개를 사용할 경우 Timeout 이 추가됩니다. value 가 high 라면 high 되는 것을 기다리는 시간입니다.

만약 timeout 시간 동안 high 가 안 된다면 "0"을 return 합니다.

출력(Return)은 1us 단위로 범위는 10us 에서 3 분까지입니다.

Pulsein()에서 입력 받은 값은 거리로 계산하기 위해서
(340 * Length) / 10000) / 2 이렇게 사용합니다.

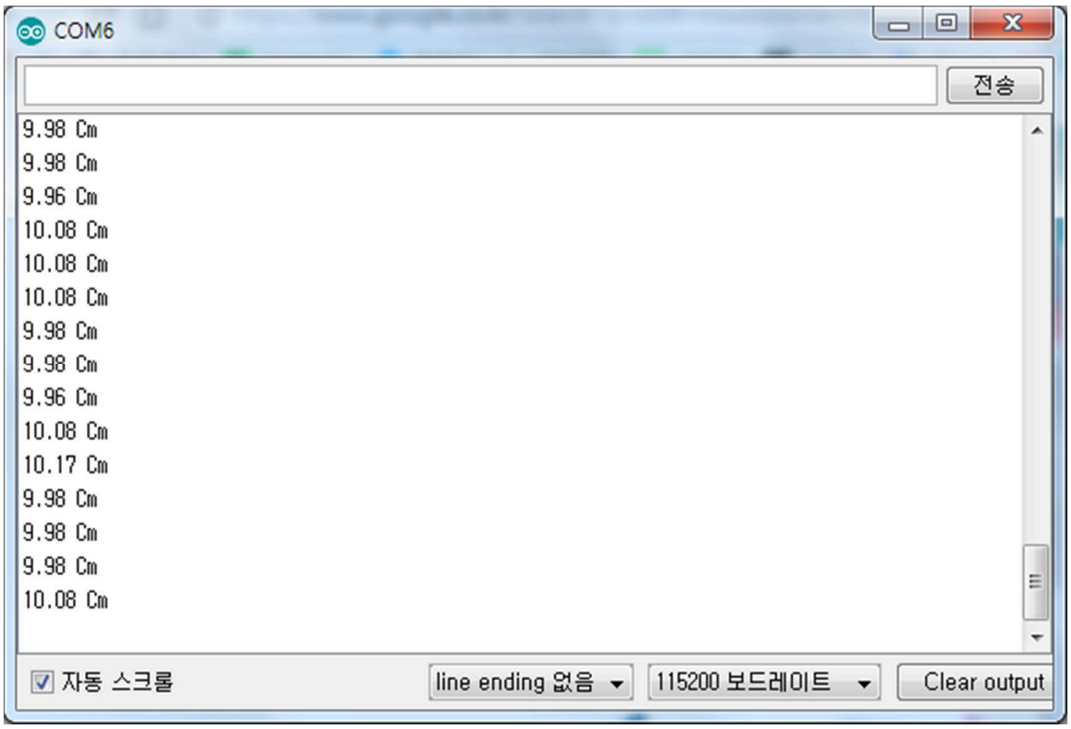
이전에 말씀 드린 것처럼 초음파는 소리와 동일하게 1 초에 340m 가 전파됩니다. 그리고 10000 으로 나눈 것을 "Cm"로 단위 환산한 것입니다.

마지막으로 나누기 2 는 목표 물체까지 거리만 측정하기 위해서입니다.

/2 를 하지 않는다면 초음파센서부터 물체 반사되어 돌아오는 전체 거리가 측정하게 됩니다.

- 결과

초음파 센서 10Cm 앞에 물체를 인식하는 결과 입니다.



○ 인체감제센서

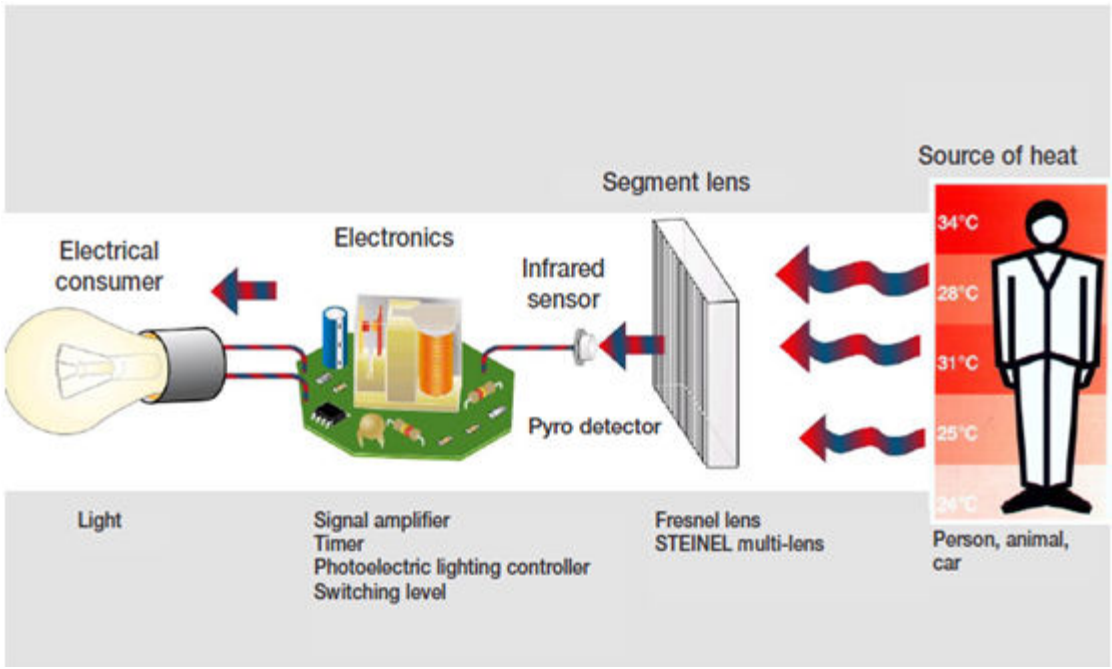
- 개요



인체감지센서 PIR(Passive Infrared Sensor)는 적외선 감지 센서입니다.

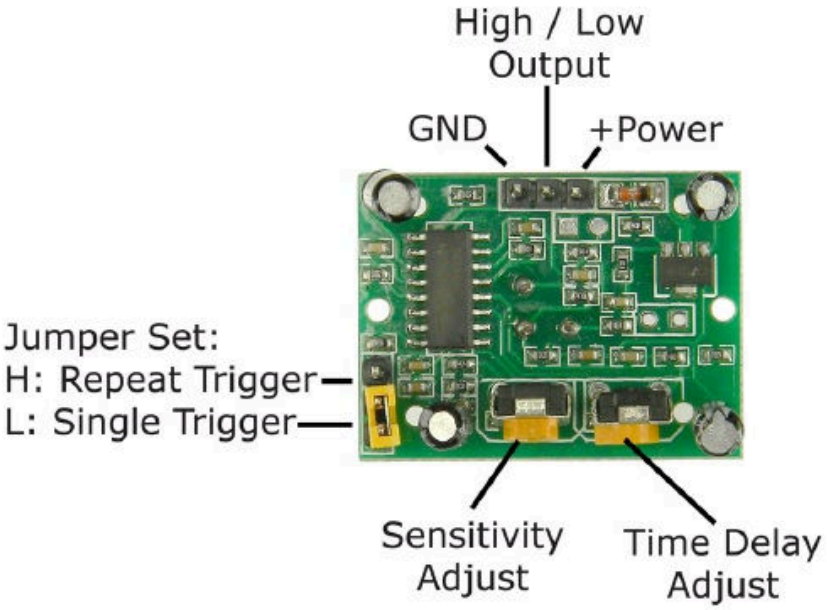
우리 주변에서 흔히 볼 수 있는 아파트 현관/계단/복도에서 사람이 인식되면 자동으로 형광등을 켜주는 장치에 부착되어있는 센서모듈입니다.

인체감시센서의 원리는 사람의 인체에서 36.5 도의 열에서 방출되는 열파장을 적외선센서에서 감지하는 원리입니다.



PIR 의 원리 (출처: https://pirtechnology.wordpress.com/2011/09/09/hello-world/)

모듈의 외부 인터페이스를 살펴보겠습니다.



VCC, GND : 입력 전압(5V) 및 GND

Output : 인체를 감지하면 HIGH, 아니면 LOW 을 출력

Jumper Set : 반복적으로 인체를 감지할 것인지 결정하는 점퍼

Sensitivity Adjust : 인체를 인식하는 감도 조정

Time Delay Adjust : 출력신호의 시간을 조정(5~200 sec)

SR501 의 주요 사양은 다음과 같습니다.

- working voltage range: DC 4.5 - 20V
- Quiescent Current : 50uA
- high output level 3.3V / Low 0V
- Trigger L trigger can not be repeated / H repeated trigger
- circuit board dimensions: 32*24 mm
- maxium 110º angle sensor

- 7m maximum sensing distance

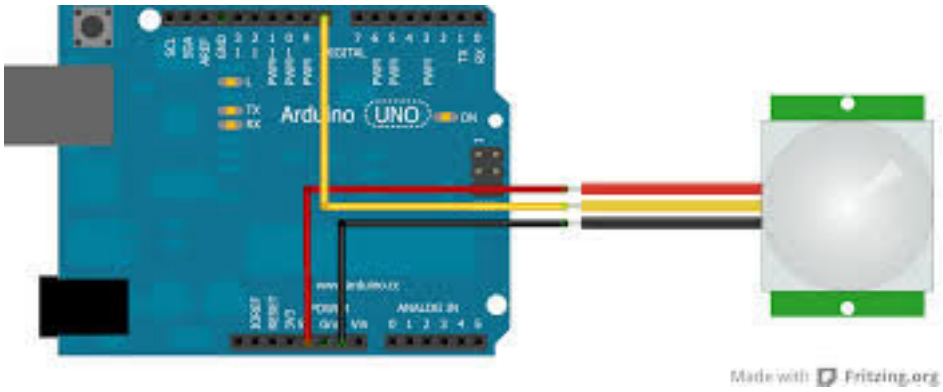
- 준비물

. 아두이노 우노(호환보드) X 1

. SR501 인체감지센서 모듈 X 1

. 점퍼와이어

- 연결도



- 코딩

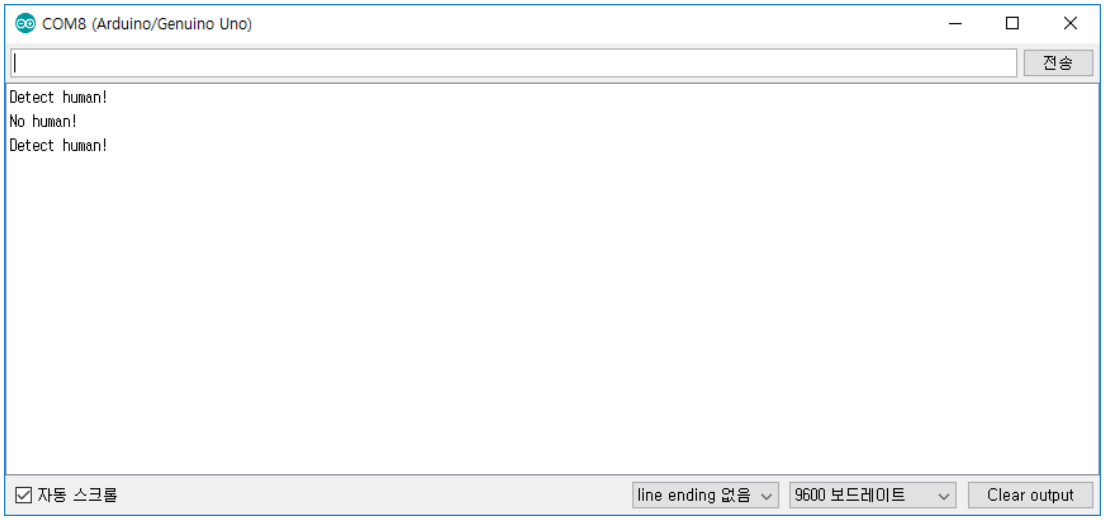
코드는 사람(인체)가 감지되면 디지털신호로 출력하기 때문에 매우 간단합니다. 만약 인식감도 및 출력시간를 변경하고자 한다면 모듈 뒷면에 있는 가변저장을 조정해서 사용하시면 됩니다.

```
//*****************************************************************************
// Project : SR501 human detect in learing set
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
#define  DETECT   8
void setup() {
  // initialize the digital pin as an output.
  pinMode(DETECT, INPUT_PULLUP);
  Serial.begin(9600);
}
void loop() {
  int sensorValue = digitalRead(DETECT);
  static int flag=0;
```

```
  if(sensorValue == 1 && flag==0 ){
     digitalWrite(13, HIGH);   // set the LED on
     Serial.println("Detect human!");
     flag=1;
  }else if(sensorValue == 0 && flag==1){
     Serial.println("No human!");
    flag=0;
  }
}
```

- **결과**

인체가 감지되면 Detect 신호를 보냅니다.



○ 터치센서

- 개요

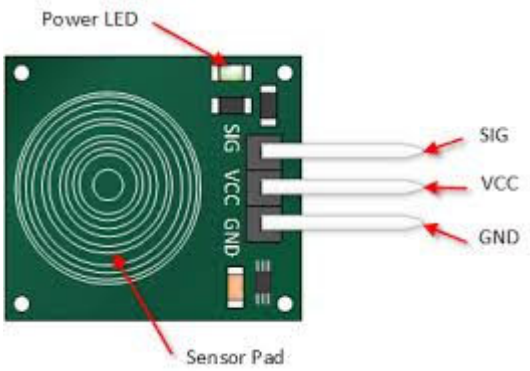


터치센서는 TTP223B 를 탑재한 정전식 터치센서 모듈입니다.

정전식이란 인체에 미세하게 흐르는 전류를 감지해서 터치한 것을 감지하는 것을 뜻합니다. 최근 스마트폰, 태블랫 터치패널로 많이 사용하고 있습니다.

TTP223B 역시 인체의 전기를 감지해서 터치 유무에 대한 정보를 디지털 신호로 출력을 합니다.

모듈의 외부 인터페이스를 살펴보겠습니다.



Power LED : 전원이 공급되면 LED 가 ON

Sensor pad : 손가락으로 터치를 하는 곳

SIG : Sensor pad 의 터치 유무를 디지털 신호로 출력

VCC : 5V 전압, GND : Ground

TTP223B 의 주요 사양은 다음과 같습니다.

Low power consumption Power supply for 2 ~ 5.5V DC Can replace the traditional touch of a button Operating voltage: DC 2.0V ~ 5.5V VOH: VCC 0.8V VOL: VCC 0.3V Sink current: 8mA @ VCC = 3V, VOL = 0.6V Source current: -4mA @ VCC = 3V, VOH = 2.4V Response time: 220ms (at mode of low power consumption) and 60ms (at quick speed mode) Size: 24 x 24 x 7.2mm

- 준비물

. 아두이노 우노(호환보드) X 1

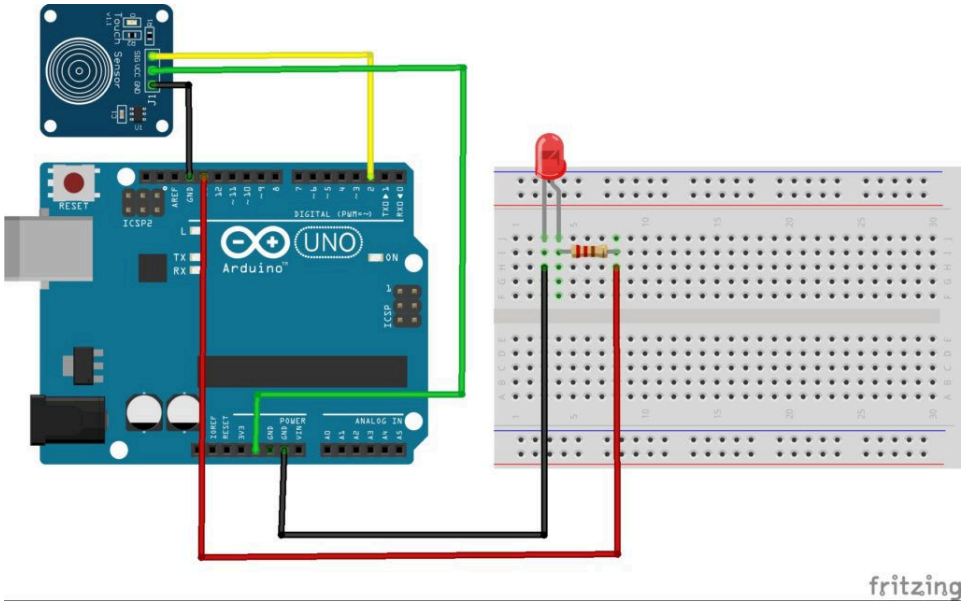
. TTP223B 터치센서 모듈 X 1

. LED X 1

. 220 옴 저항 x 1

. 점퍼와이어

- 연결도



- 코딩

코드는 사람(인체)가 감지되면 LED 를 제어하는 코드입니다.

```
//*********************************************************************************
// Project : TTP223B Touch Sensor human detect in learning set
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************

// When Sig Output is high, touch sensor is being pressed

int touchPin = 2; // pin for the touch sensor
int ledPin = 13; // pin for the LED

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);  // set the Led Pin to output
  pinMode(touchPin, INPUT); // set the Touch Pin to input
}

void loop() {
  int touchValue = digitalRead(touchPin); // read the Touch Pin
  if (touchValue == HIGH) { //check if the sensor is touched
    digitalWrite(ledPin, HIGH);
    Serial.println("TOUCHED");
  }
  else {
    digitalWrite(ledPin, LOW);
    Serial.println("not touched");
  }
  delay(100);

}
```

- 결과

LED 및 시리얼 모니터에서 터치 상태를 확인할 수 있습니다.

◻ 참고자료 출처 및 기타 정보

https://www.arduino.cc/en/Main/Education

https://www.elegoo.com/

http://fritzing.org/download/

http://GOOGLE.COM/

전자파영향성

키트플러스 아두이노 키트는 KC 전자파적합성평가가 완료된 키트입니다.

본 아두이노 키트는 외부 차폐된 하우징(케이스)가 있어야 외부기자재(전자장비)에 전자파영향을 미치지 않습니다.

외부 전원어댑터 사용

아두이노 우노 보드는 USB 전원으로 사용(동작)이 가능하지만,

별도의 외부 전원어댑터 사용하시는 것을 권장 드립니다.

* 본 문서의 저작권은 출처를 밝히고, 비영리 목적으로 배포/활용이 가능합니다.

아두이노 IOT / 고급자키트 매뉴얼



Last Update : 19/03/10

By : 키트플러스

시작하며

본 매뉴얼은 아두이노 IOT/고급자 키트에 관한 사용자 매뉴얼 입니다.

아두이노 IOT/고급자 키트는 중급자 키트 이후 더욱 다양하고 깊이 있는 디바이스를 공부하시고 싶으신 분 들을 위한 키트 입니다. 초/중급자 키트를 마치신 분들과 LCD 와 같은 디스플레이 장치들을 제어하고 싶으신 분, IOT(사물인터넷:Internet Of Things)을 공부하시고 싶으신 분 그리고 특별한 아두이노 만들고 싶으신 분 들에게 매우 적합한 세트입니다.

특히 IOT(사물인터넷:Internet of Things)은 4 차산업의 핵심 주제로 각종 사물에 센서와 통신 기능을 내장하여 인터넷에 연결하는 기술을 말합니다. 무선 통신을 통해 각종 사물을 연결하는 기술도 포함됩니다. 인터넷으로 연결된 사물들이 데이터를 주고받아 스스로 분석하고 학습한 정보를 사용자에게 제공하거나 사용자가 이를 원격 조정할 수 있는 인공지능 기술과 함께 모든 사물이 연결 것을 의미합니다. 여기서 사물의 뜻은 가전제품, 모바일 장비, 웨어러블 디바이스 등 모든 임베디드 시스템을 말합니다. 사물인터넷에 연결되는 사물들은 자신을 구별할 수 있는 유일한 IP 를 가지고 인터넷으로 연결되어야 하며, 외부 환경으로부터의 데이터 취득을 위해 무선통신 기능를 포함되어야 합니다.

IOT/고급자 키트에서 제공되는 디바이스들은 0.96 인치 I2C OLED, 인체감지센서 SR501, L298n 모터제어모듈, DC 모터, MAX7219 매트릭스, 3 축 지자계센서, 2.4 인치 TFT LCD, HC-06 블루투스 슬레이브 모듈, NRF24L10 2.4G RF T/RX 모듈, ENC28J60 이더넷모듈, ESP8266 EP01 WIFI 모듈 들이 있습니다.

중급자키트에 이어 고급자 세트까지 마치셨다면 아두이노와 관련된 학습은 대부분 마무리가 된 것 입니다. 학습자 분께서 임베디드와 관련된 학습이 더 필요하시다면 고민이 되신다면, 이제부터 라즈베리파이 / 바나나파이 / 오렌지파이와 같은 OS 기반의 보드를 고려해 보실 수 있습니다. 위와 같은 보드들은 기본적으로 리눅스 OS 가 탑재되어있고, 활용범위는 무한합니다. 스마트폰, 타블랫 PC, 오락실 레트로 게임기, 서버, 영상처리, 안드로이드 노트북 등 모든 것을 만들 수 있는 보드입니다

본 문서가 코딩교육이 필요하신 분들에게 많은 도움이 되었으면 합니다.

** 제조사 사정으로 인해 본 문서에서 사용되는 일부 부품 구성이 다를 수 있습니다.*

** 고급자 키트에는 아두이노가 포함되어있지 않습니다.*

"나는 나의 의지대로 된다."

-쇼펜하우어

목 차

아두이노란 4

하드웨어 구성 5

아두이노 우노 R3 보드 5

IOT/고급자키트 구성 7

브래드보드 8

아두이노 보드 PC 연결 9

소프트웨어 시작하기 10

아두이노 IDE 다운로드 10

아두이노 IDE 설치 11

아두이노 IDE 및 소프트웨어 구조 15

아두이노 라이브러리 추가 16

아두이노 보드 선택 17

컴파일 및 업로드 18

시리얼 모니터 사용법 20

아두이노 코딩절차 21

예제 학습하기 22

DC 모터제어 26

MAX7219 매트릭스 32

3 축 자이로센서 39

OLED 0.96 인치 50

2.4 인치 TFT LCD 59

2.4Ghz 무선송수신 66

블루투스 슬레이브 71

이더넷모듈 78

무선 WIFI 모듈 83

참고자료 95

What is ARDUINO ?

*** 고급자 키트에는 아두이노 우노 보드가 포함되어있지 않습니다.**



아두이노는 2005년 이탈리아에서 개발된 오픈 소스 기반을 한 단일 보드를 뜻합니다.

아두이노가 세상에 나오기 전에는 마이크로컨트롤러라는 것을 비전공자가 공부하기 어려웠습니다. 일반인이 자료를 구하는 것도 쉽지 않았고, 개발환경을 구축하는 것도 복잡한 과정이 필요했습니다. 그리고 하드웨어를 구입해야 하는 비용도 저렴하지 않았습니다.

하지만 아두이노는 저렴한 보드와 다양한 부품 그리고 강력한 소프트웨어의 지원으로 하드웨어에 익숙지 않는 학생 그리고 비전공자들도 자신만의 프로젝트를 디자인/개발 할 수 있도록 도와 줍니다.

즉, 아두이노는 임베디드시스템을 누구나 쉽게 개발할 수 있는 환경을 제공합니다.

현재도 아두이노와 관련된 수많은 센서와 장치들이 새롭게 등장하고 있습니다.

하드웨어 장치와 더불어 누구나 사용이 가능한 수많은 라이브러리도 계속 업데이트 되고 있습니다.

아두이노는 오픈하드웨어/오픈소스 기반으로 하고 있기 때문에 웹에서 기술 자료를 찾는 것이 어렵지 않습니다. 그러므로 누구나 상상하거나 만들고 싶다는 의지만 있으면 자신만의 프로젝트를 만들 수 있습니다. 로봇, IOT, 심지어 휴대폰까지도 아두이노를 활용해서 설계/제작이 가능합니다.

다양한 아두이노 보드가 있지만 본 매뉴얼에서는 "아두이노 우노 R3"를 사용한 Learning 키트 사용법에 대해서 설명을 드리겠습니다.

□ 하드웨어 구성

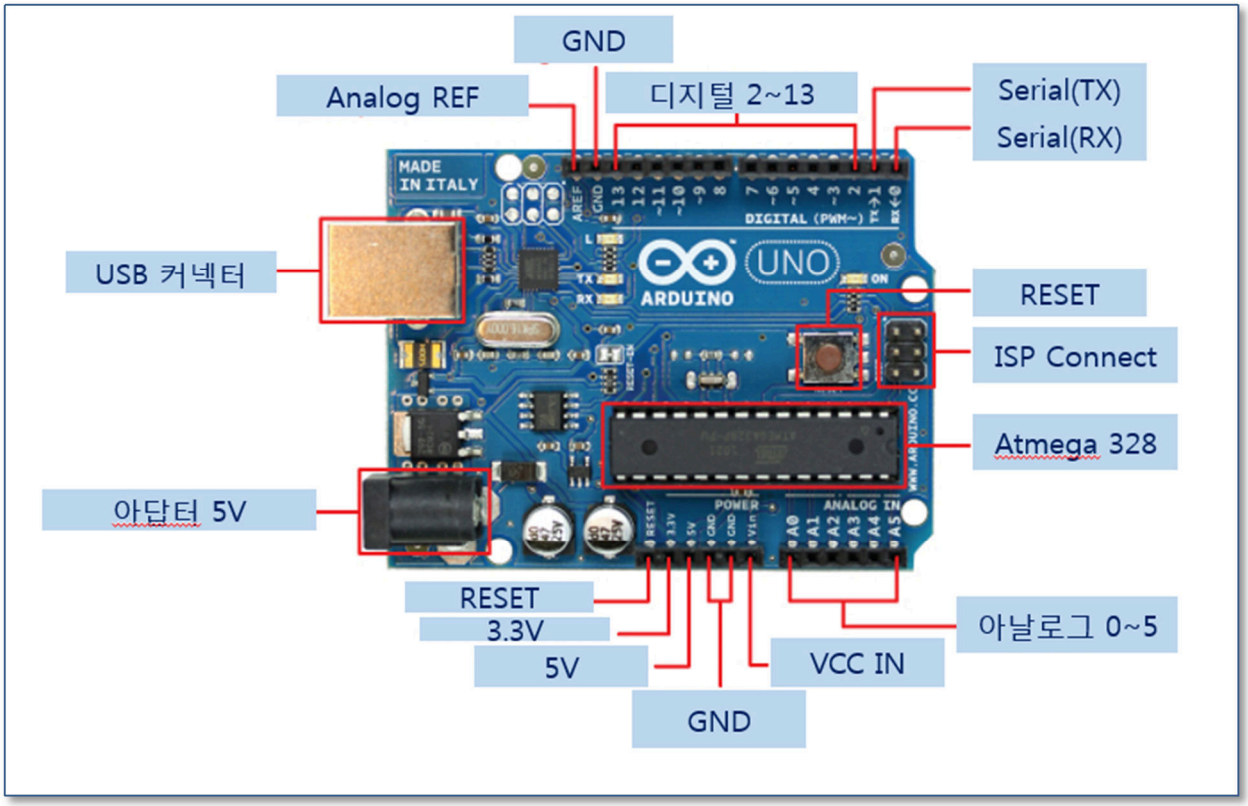
○ 아두이노 우노 R3보드

아두이노 우노 보드는 ATmega328 기반의 마이크로 컨트롤러 보드입니다.

14개의 디지털 I/O핀(14개 중 6개의 핀은 PWM 출력으로 사용 가능)과 6개의 아날로그 입력, 16MHz ceramic resonator로 구성되며, 외부 USB 커넥터, 파워 커넥터, ISP 헤더, 리셋버튼이 내장되어있습니다.

쉽게 컴퓨터와 USB 케이블로 연결할 수 해서 사용할 수 있습니다. 전력소모가 많은 디바이스(ex. 모터)를 사용할 경우에는 추가로 아답터가 필요할 수도 있습니다.

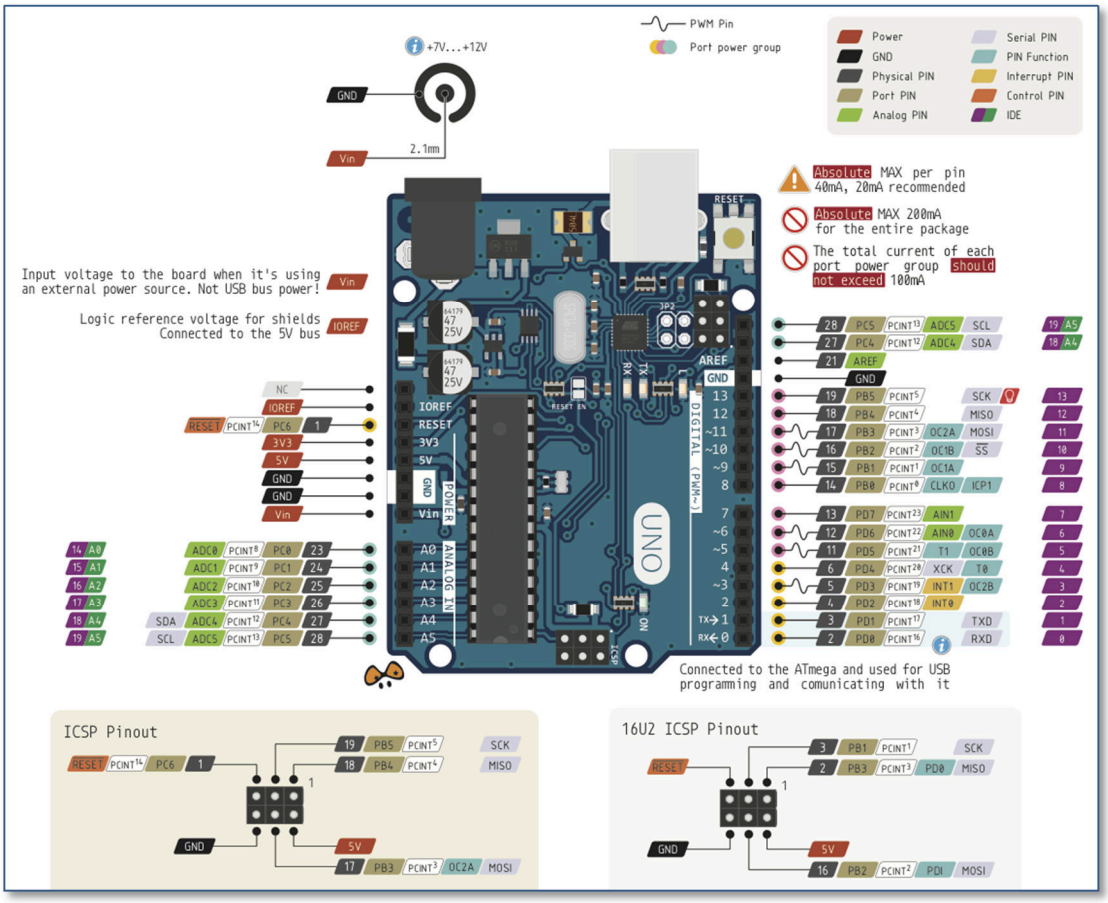
아두이노 우노 정품보드는 USB-to-Serial는 Atmega16U2로구성되어있고, 호환보드는 FT232 또는 CH340으로 구성되어있습니다. 만약 CH340 호환보드를 사용하실 경우에는 별도의 드라이버를 설치하셔야 합니다. 외형적인 핀맵 및 기본 구성은 정품/호환 보드 모두 동일합니다.



[정품 아두이노 우노 구성도]

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by boot loader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

[아두이노 우노 R3 주요 사양]



(출처 : http://forum.arduino.cc/index.php?topic=146315.0)

[아두이노 우노 핀맵]

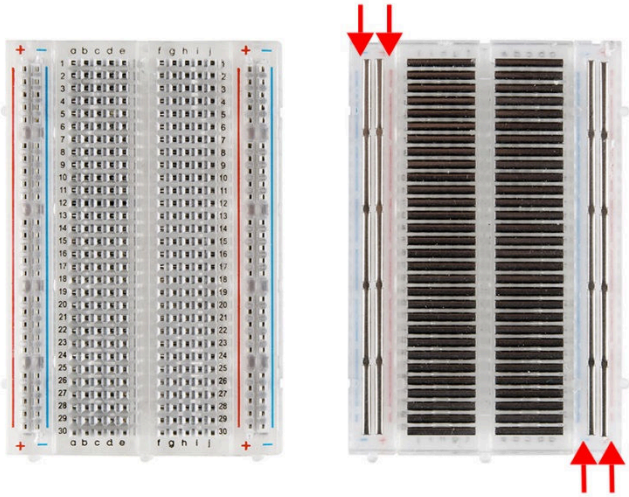
○ IOT/고급자 세트 구성

*** 아두이노 우노 R3 는 포함되어있지 않습니다.**

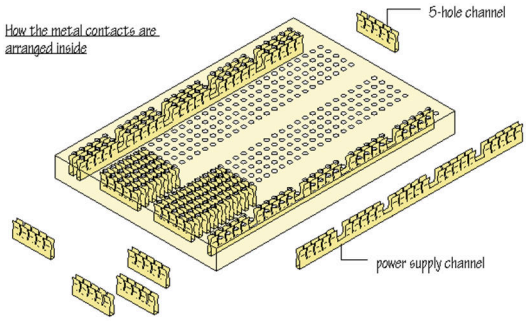
No	Part	한글명	EA
1	RFID 모듈	RFID	1
2	L298n DC motor driver	DC 모터 드라이버 모듈	1
3	DC motor and wheel	DC 모터 및 바퀴	2
4	MAX7219 LED display Drivers	병렬 도트매트릭스	1
5	MPU6050 Accelerometer/Gyroscope	3 축 가속 / 자이로 센서	1
6	0.96 OLED	OLED	1
7	2.4 inch TFT LCD	2.4 인치 TFT LCD	1
8	nRF24L01 Single Chip 2.4GHz Transceiver	무선 송수신 모듈 2.4GHz	2
9	HC06 Slave Bluetooth	블루투스 슬레이브 모듈	1
11	ENC28J60 Ethernet module	이더넷 모듈	1
12	ESP8266 EP-01	WIFI 모듈	1

○ 브래드보드

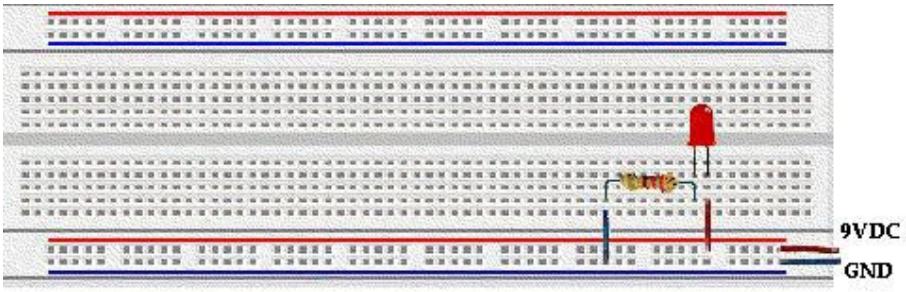
브래드보드란 납땜 작업 없이 전자부품들을 서로 빠르고 쉽게 연결 할 수 있는 보드입니다. 기본 형태는 여러 개의 홀(구멍)이 촘촘히 배치되어있습니다. 홀 내부는 "a,b,c,d,e" 가 서로 연결이 되어있고, +,- 의 전원선도 수직으로 연결 되어있는 구조입니다.



출처 : SparkFun Electronics



출처 http://secte1.elec.uow.edu.au

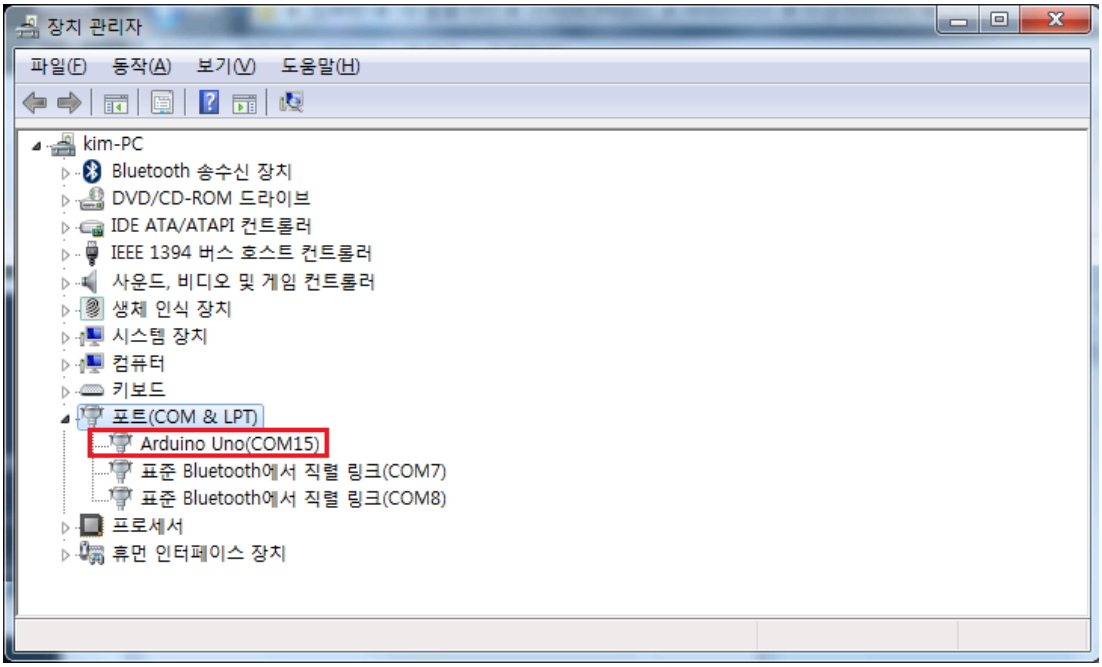


[브래드보드 내부 구조]

○ 아두이노 보드 PC 연결

USB 케이블을 아두이노와 PC 에 연결하고, 장치관리자에서 COM 포트를 확인합니다.





□ 소프트웨어 시작하기

○ 아두이노 IDE 다운로드

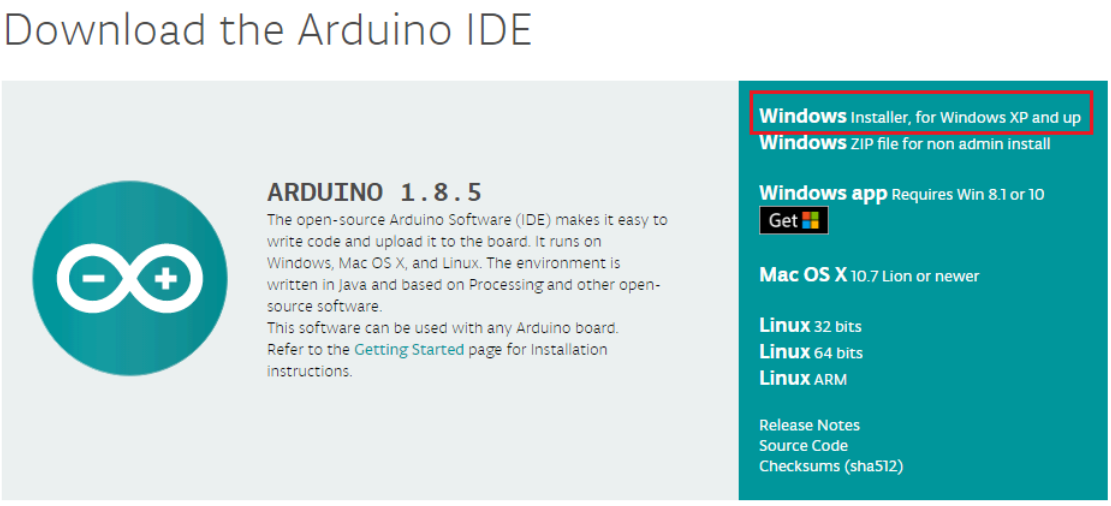
하드웨어 보드가 준비되셨다면, 이제 아두이노 소프트웨어가 필요합니다. 아두이노는 코딩, 컴파일, 업로드가 가능한 IDE를 기본 제공합니다.

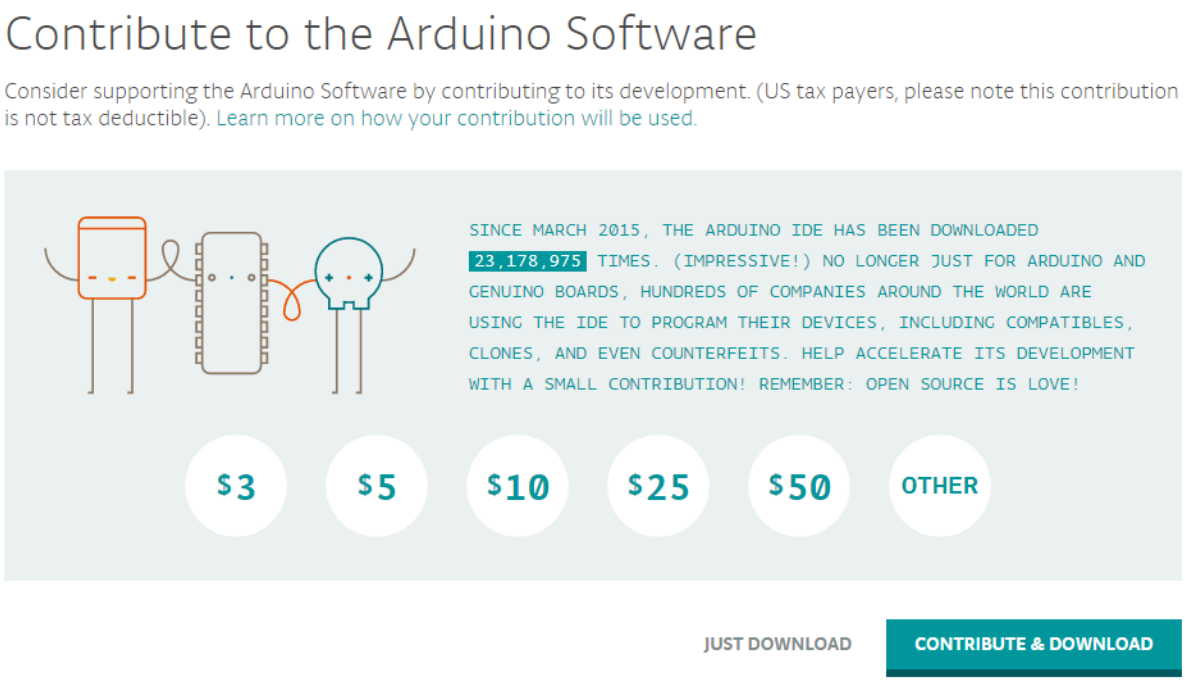
> *"IDE란, 통합 개발 환경(統合開発環境, Integrated Development Environment, IDE)으로 코딩, 디버그, 컴파일, 배포 등 프로그램 개발에 관련된 모든 작업을 하나의 프로그램 안에서 처리하는 환경을 제공하는 소프트웨어를 뜻합니다."*

아두이노 IDE 다운로드 홈페이지를 방문하시면 web을 이용한 아두이노 editor가 있지만, 본 매뉴얼에서는 Windows 기반의 아두이노 IDE에 대해서 설명을 합니다.

가장먼저 아래 링크에서 아두이노 IDE를 다운로드 받습니다.

아두이노 다운로드 링크 : https://www.arduino.cc/en/Main/Software





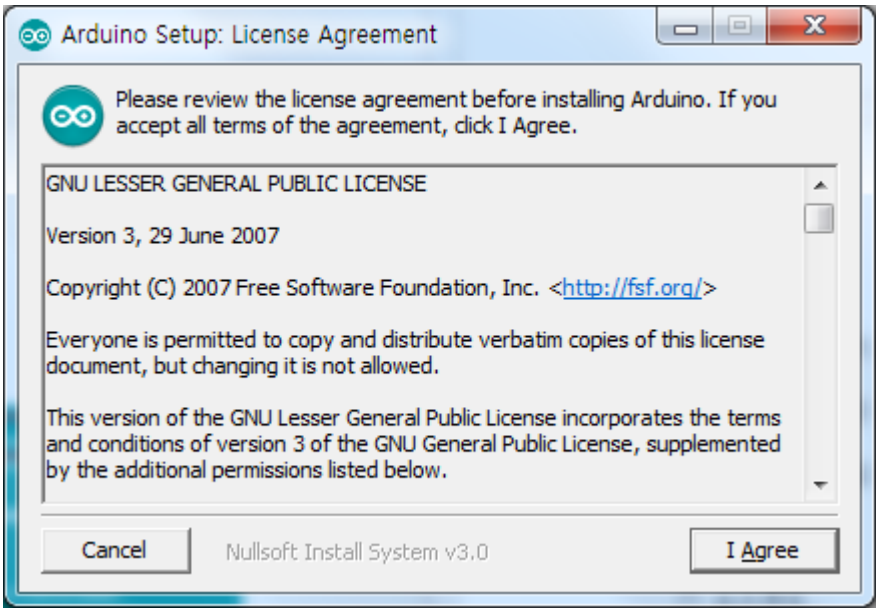
기부를 원하시는 분들은 "Contribute & DOWNLOAD"를 누르시고,

아니시라면 "JUST DOWNLOAD"를 눌러서 다운로드를 진행합니다.

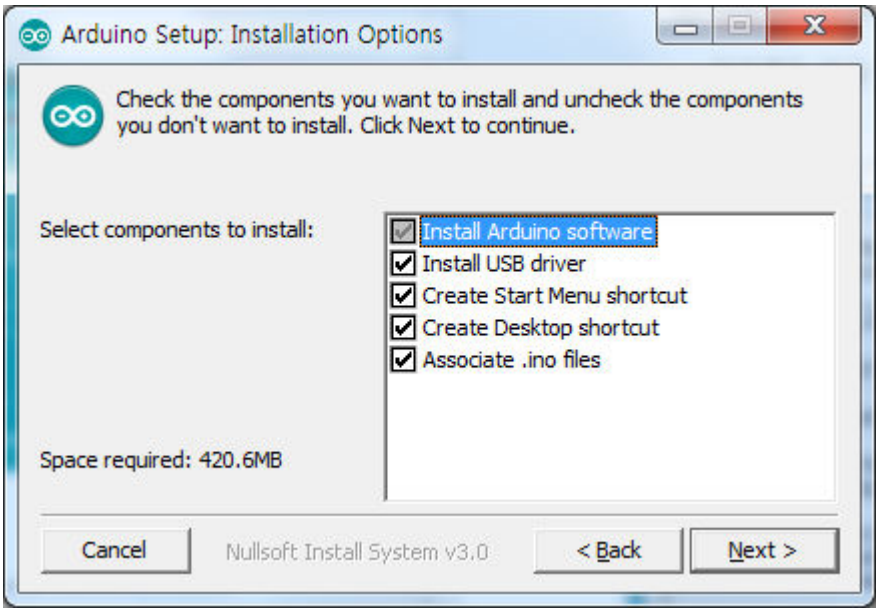
○ 아두이노 IDE 설치

다운로드 받은 "arduino-1.8.5-windows.exe" 파일을 실행합니다.(버전은 변경 될 수 있습니다.)

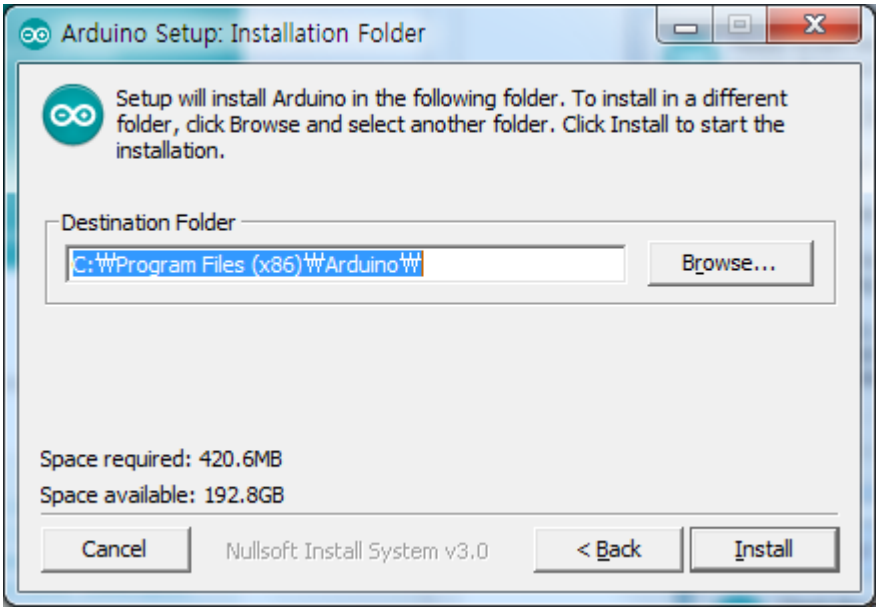
아래 라이선스 정책에 "I Agree"(동의) 버튼을 누릅니다.



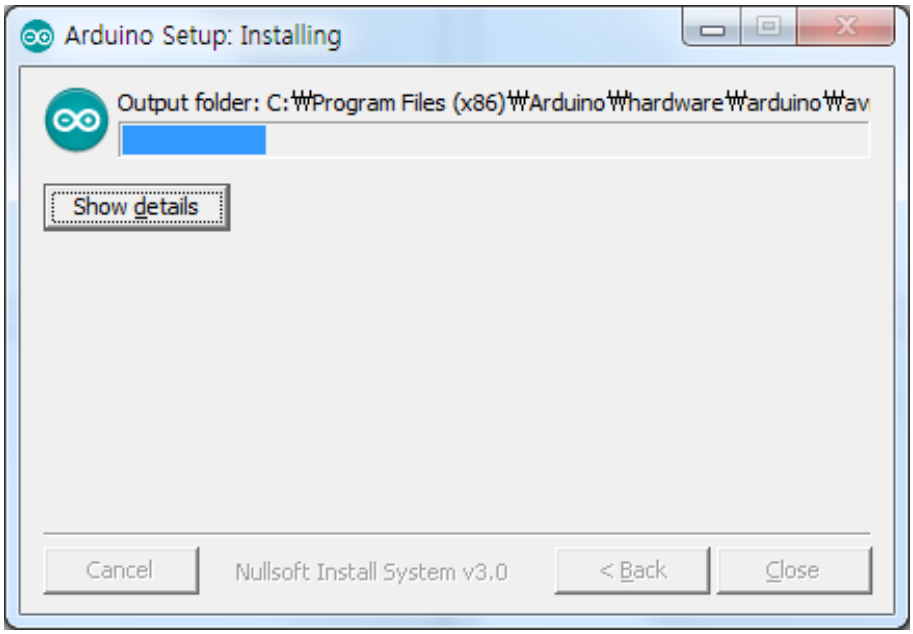
설치 옵션에서 USB Driver를 포함해서 모두 체크하시고 "NEXT>"를 누릅니다.



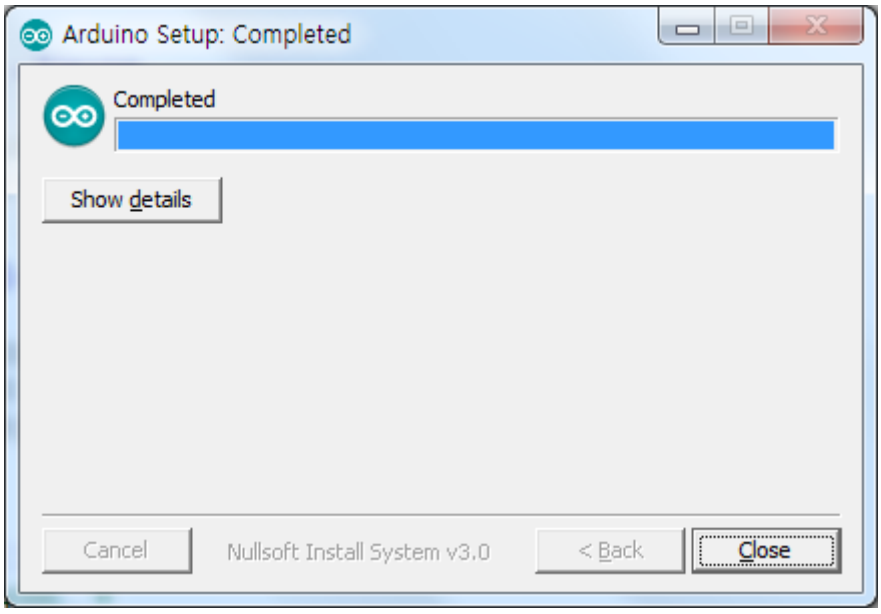
설치 경로를 입력하신 후 "Install"를 누릅니다.



설치를 시작합니다. 2~3분정도의 시간이 필요합니다.



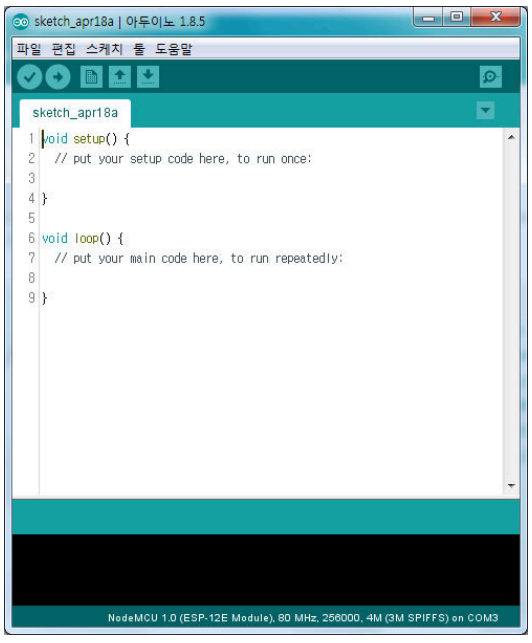
설치가 완료되면 "Close"를 눌러 설치를 종료합니다.



바탕화면의 "Arduino" 아이콘을 두번 클릭해서 아두이노 IDE를 실행합니다.

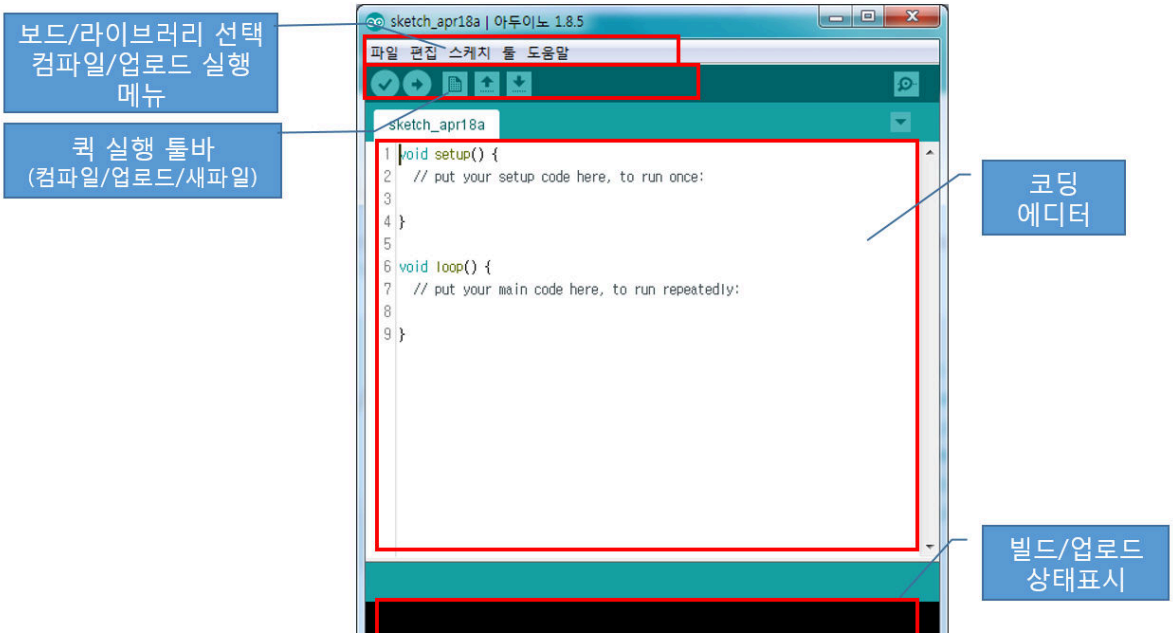






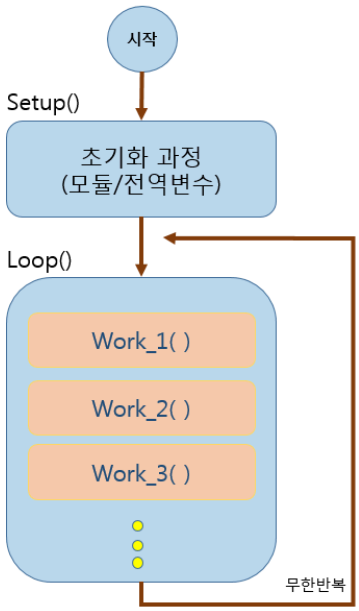
○ 아두이노 IDE 및 소프트웨어 구조

아두이노 IDE는 크게 메뉴와 코딩 에디터, 상태 표시 콘솔로 구성되어있습니다.



아두이노의 소프트웨어 구조는 객체지향 C++ / C 를 사용합니다.

전원인가 이후 RESET이 최초로 실행되는 코드는 Setup() 함수에서 필요한 모듈 및 변수들을 초기화 합니다. 이후 Loop()에서 무한 반복을 하면서 일을 수행합니다.



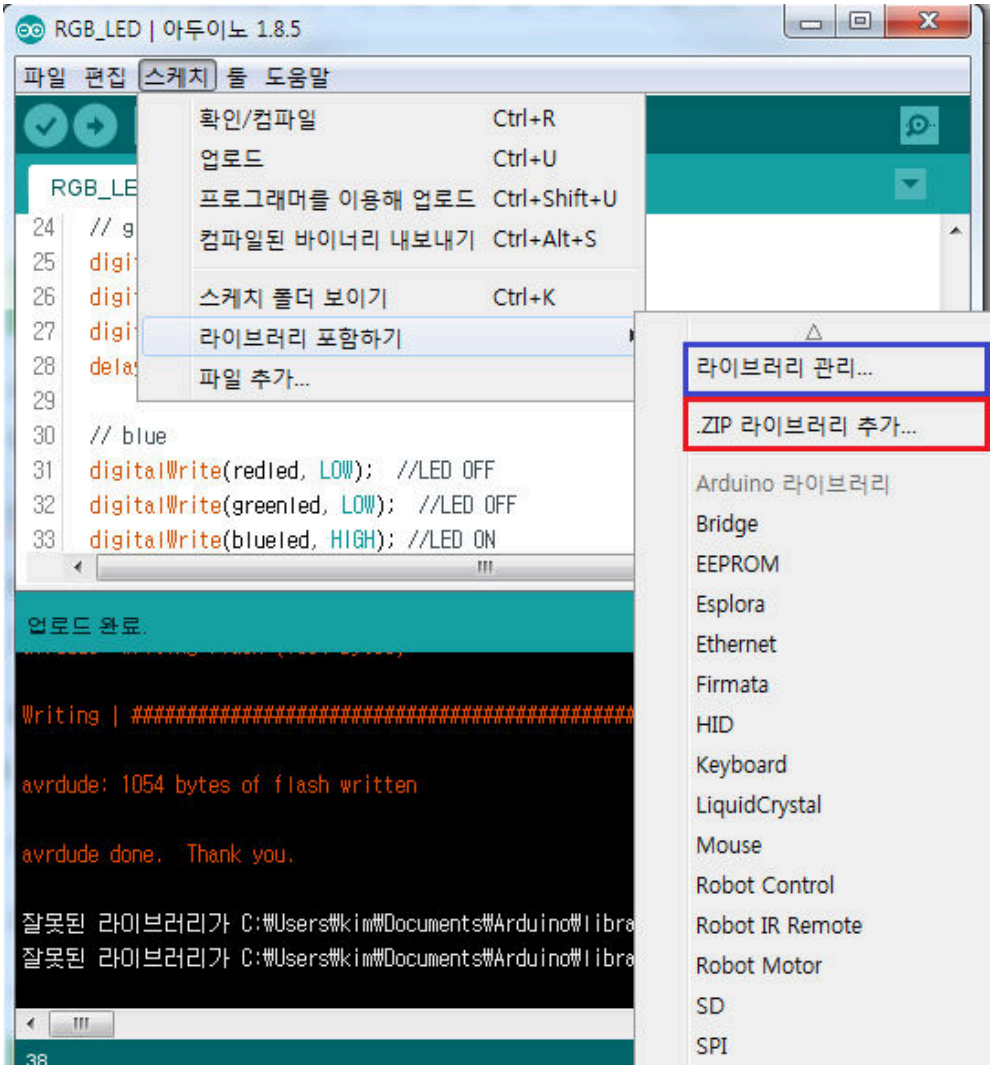
○ 아두이노 라이브러리 추가

소프트웨어 라이브러리란 센서, 디스플레이, 모터 등과 같은 모듈들을 쉽게 사용하기 위한 코드들의 집합체입니다.

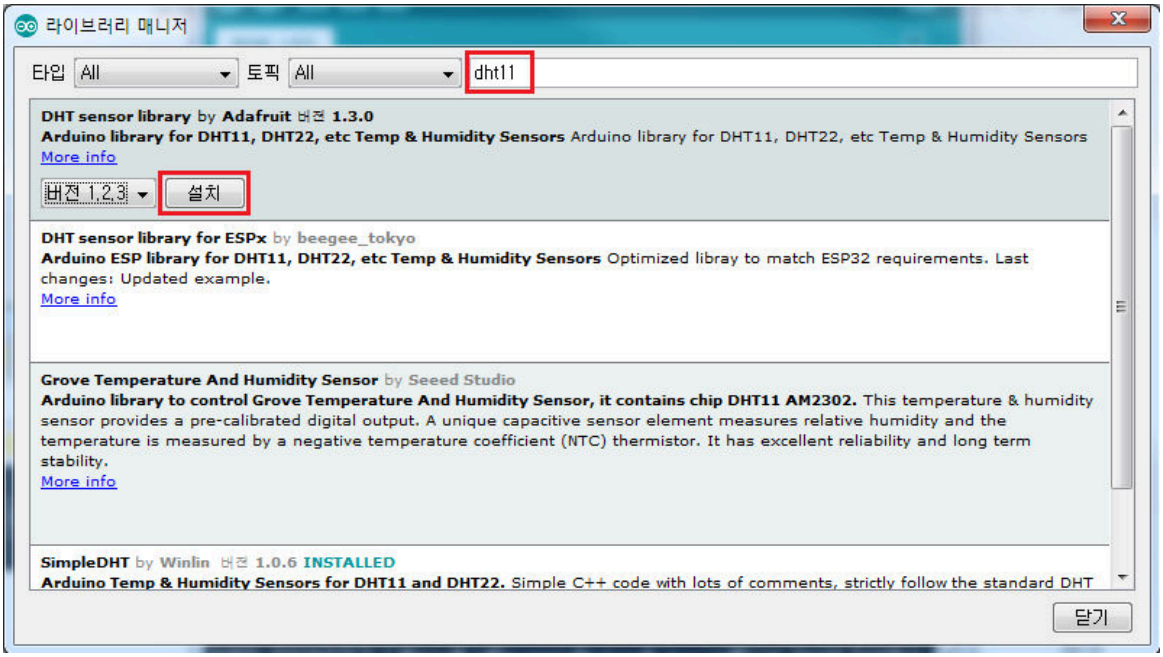
아두이노 IDE에는 기본라이브러리가 포함되어있습니다.

하지만 기본 라이브러리에 없는 새로운 모듈을 사용하기 위해서는 라이브러리를 추가로 설치해야 합니다.

다음 그림은 온습도 센서인 DHT11를 사용하기 위한 라이브러리를 설치하는 과정입니다. 아두이노 온라인을 이용한 라이브러리 추가는 "라이브러리 관리"를 선택하시면 됩니다. 별도의 라이브러리를 ZIP으로 다운로드 받으셨다면 ".ZIP 라이브러리 추가"를 누르시고 탐색창에서 라이브러리 ZIP파일만 선택해주시면 됩니다.



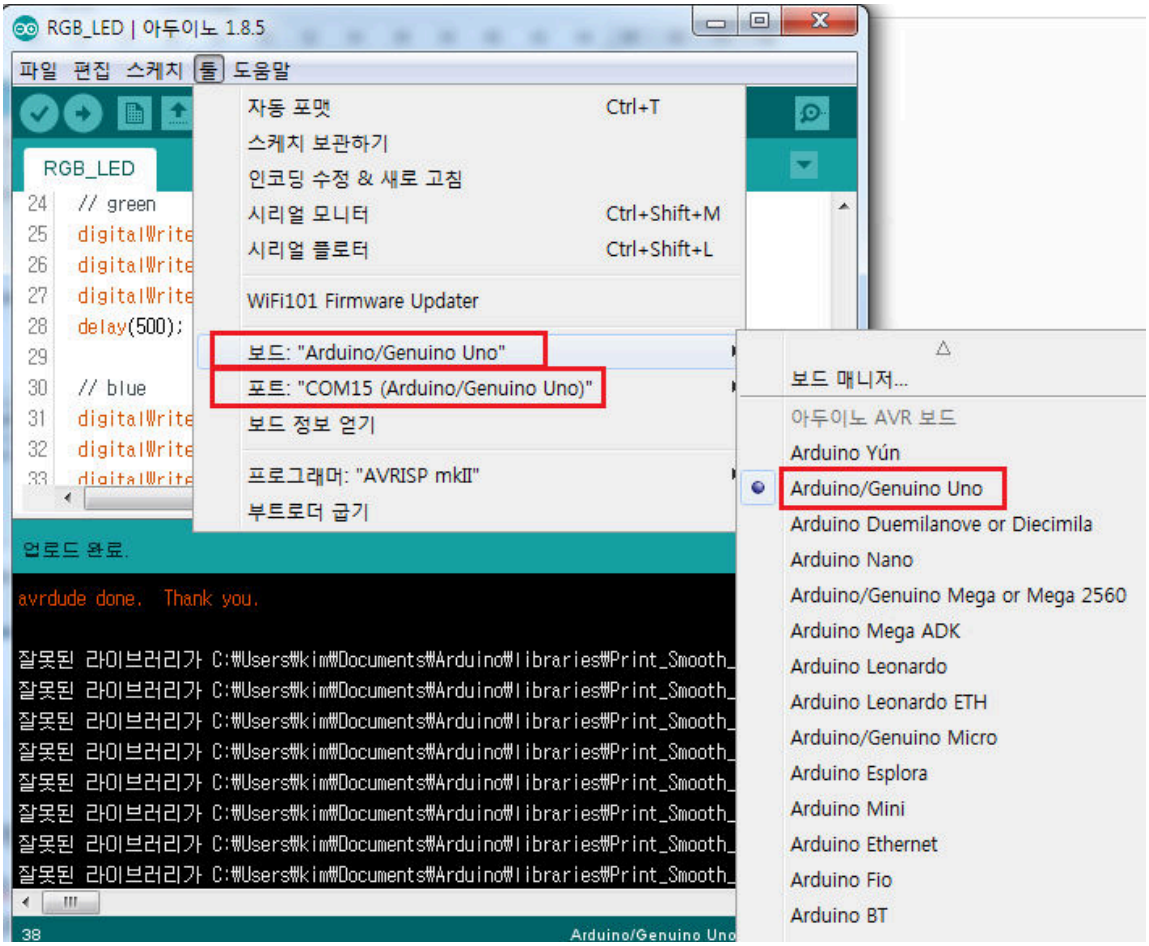
"라이브러리 관리"를 누르시면 아래와 같은 창이 나옵니다. 찾으시는 라이브러리를 검색을 하시면 관련된 라이브러리들이 표시가 됩니다. 설치버튼을 누르시면 라이브러리가 설치가 됩니다.



○ 아두이노 보드 선택

아두이노는 학습자/사용자 요구에 맞는 다양한 보드들을 있습니다. 정식으로 출시하고 있는 아두이노 미니, 아두이노 마이크로, 아두이노 레오나로드, 아두이노 메가 등이 있으며, 3rd 파트 MCU들은 ESP8266, STM32도 지원이 가능합니다.

다음 그림은 앞으로 사용하는 아두이노 우노 보드와 통신포트를 선택 과정입니다.



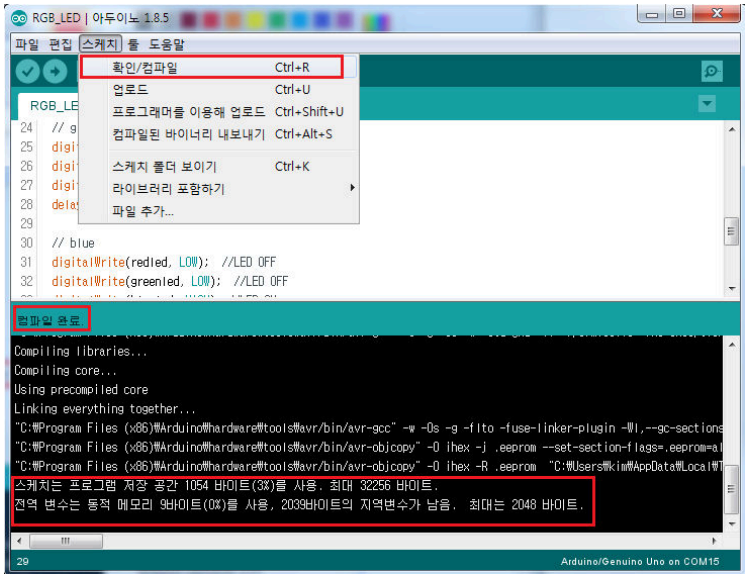
○ 컴파일 및 업로드

코딩을 완료하셨다면, 컴파일 과정을 거쳐 실행파일을 만들어야 합니다. 에러가 없다면 만들어진 실행파일을 아두이노로 업로드를 진행해야 합니다.

"컴파일이란 사람이 작성하는 프로그램(고급언어)을 기계가 인식할 수 있는 기계어로 변환하는 과정을 뜻합니다."

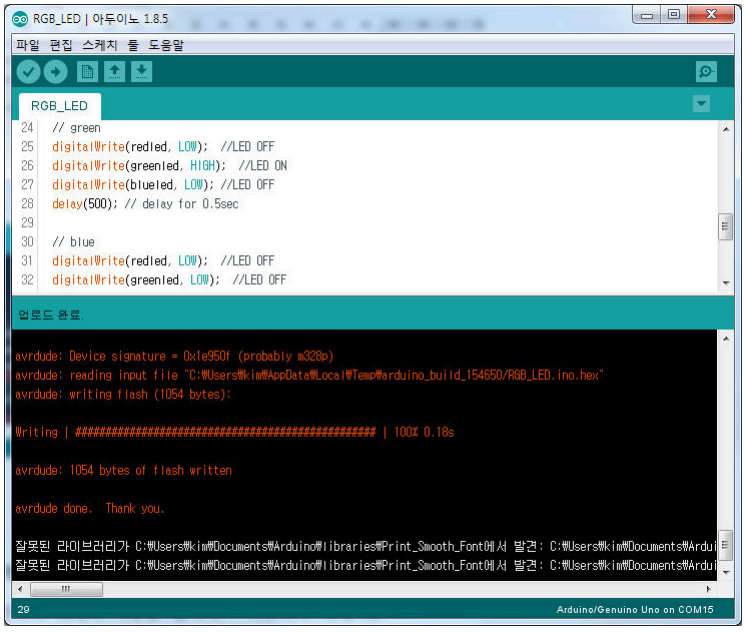
"업로드란 컴파일 과정을 거쳐 생성된 실행파일을 마이크로컨트롤로 내부 또는 외부 메모리로 주입하는 과정을 뜻합니다"

다음 그림은 스케치 -> 확인/컴파일을 실행한 그립니다. 컴파일이 완료되었다면 아래 콘솔창에 사용되는 메모리 사이즈가 표시됩니다. (단축키 Ctrl + R)



다음 그림은 스케치->업로드를 실행한 화면입니다. 주입이 100% 된 것을 확인 하시면 됩니다.

만약 COM port Error가 발생된다면, 장치관리자의 COM port와 아두이노 COM port를 확인하십시오.



단축키 Ctrl + U 를 누르면 컴파일과 업로드가 같이 진행이 됩니다.

○ 시리얼 모니터 사용법

시리얼 모니터는 아두이노가 실행되는 단계를 관찰할 수 있는 도구 입니다.

주로 디버깅이나 결과값을 확인하기 위해서 사용을 합니다. 시리얼모니터를 사용하기 위해서는 Setup에서 Serial speed만 설정하시면 사용이 가능합니다.

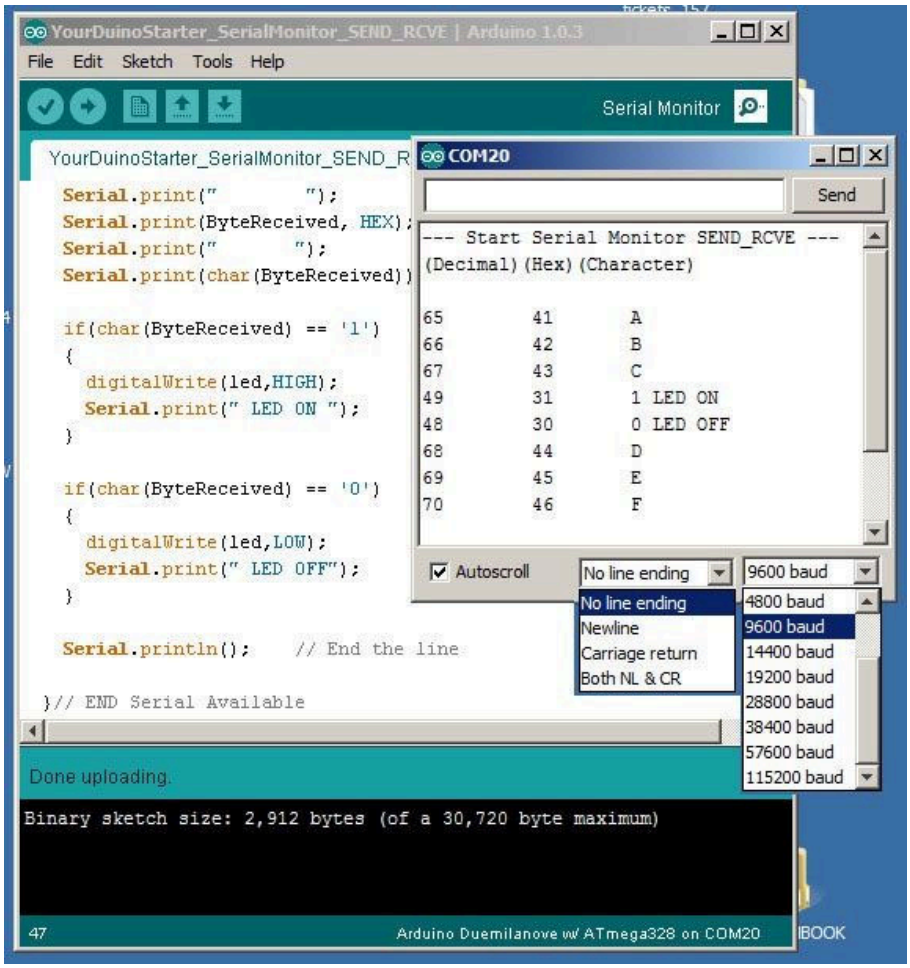
아두이노 우노에서는 Printf()는 기본적으로 지원하지 않고, Serial.print() 또는 Serial.println()을 지원합니다. Printf()를 사용하기 위해서는 별도의 class 소스 작업이 필요합니다.

[기본문법]

```
Serial.print(val);
Serial.print(val, format);
```

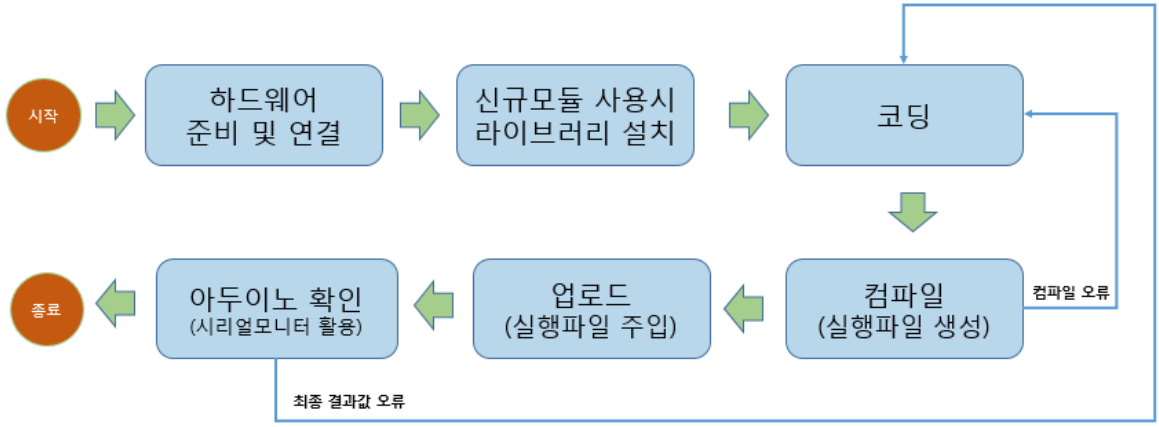
[사용예]

아두이노 IDE에서 툴->시리얼 모니터 실행(단축키 Ctrl+Shfit+M)



○ 아두이노 코딩절차

지금까지 설명 드린 절차를 요약해서 코딩과정을 순서도로 표현하면 아래와 같습니다.



□ 예제 학습하기

○ 인체감제센서

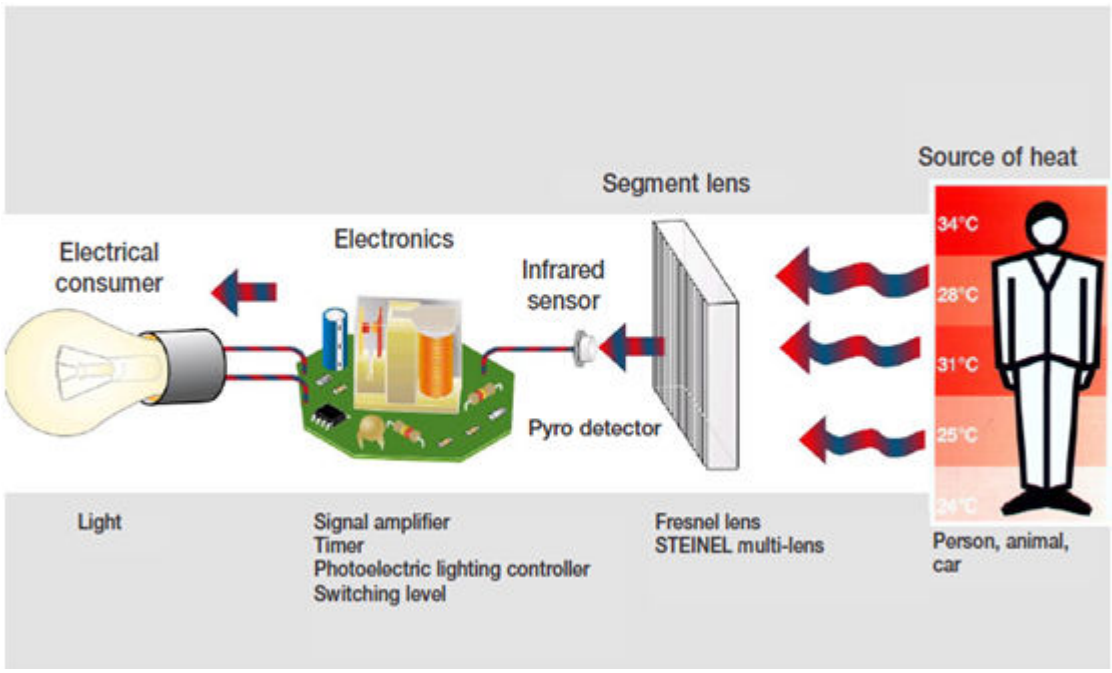
- 개요



인체감지센서 PIR(Passive Infrared Sensor)는 적외선 감지 센서입니다.

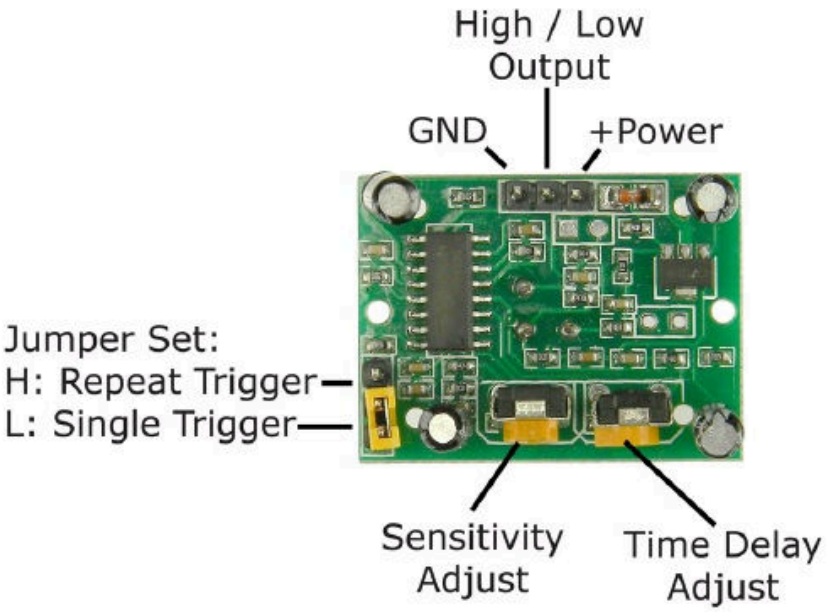
우리 주변에서 흔히 볼 수 있는 아파트 현관/계단/복도에서 사람이 인식되면 자동으로 형광등을 켜주는 장치에 부착되어있는 센서모듈입니다.

인체감시센서의 원리는 사람의 인체에서 36.5 도의 열에서 방출되는 열파장을 적외선센서에서 감지하는 원리입니다.



PIR 의 원리 (출처: https://pirtechnology.wordpress.com/2011/09/09/hello-world/)

모듈의 외부 인터페이스를 살펴보겠습니다.



VCC, GND : 입력 전압(5V) 및 GND

Output : 인체를 감지하면 HIGH, 아니면 LOW 을 출력

Jumper Set : 반복적으로 인체를 감지할 것인지 결정하는 점퍼

Sensitivity Adjust : 인체를 인식하는 감도 조정

Time Delay Adjust : 출력신호의 시간을 조정(5~200 sec)

SR501 의 주요 사양은 다음과 같습니다.

- working voltage range: DC 4.5 - 20V
- Quiescent Current : 50uA
- high output level 3.3V / Low 0V
- Trigger L trigger can not be repeated / H repeated trigger
- circuit board dimensions: 32*24 mm

- maxium 110º angle sensor

- 7m maximum sensing distance

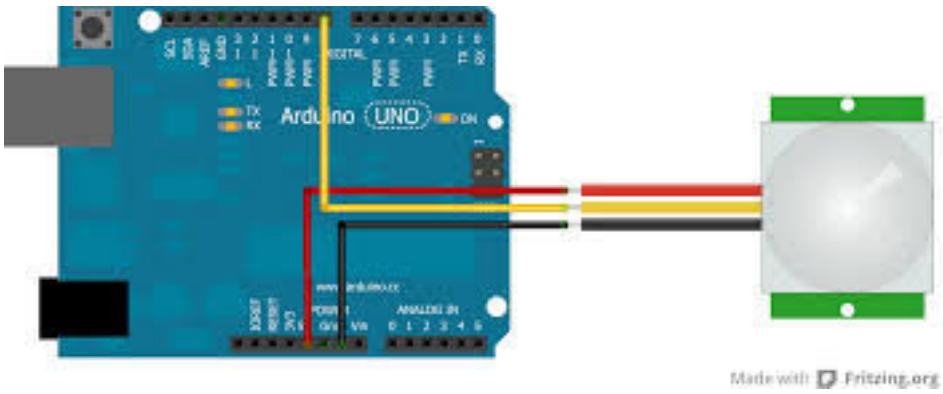
- 준비물

. 아두이노 우노(호환보드) X 1

. SR501 인체감지센서 모듈 X 1

. 점퍼와이어

- 연결도



- 코딩

코드는 사람(인체)가 감지되면 디지털신호로 출력하기 때문에 매우 간단합니다. 만약 인식감도 및 출력시간를 변경하고자 한다면 모듈 뒷면에 있는 가변저장을 조정해서 사용하시면 됩니다.

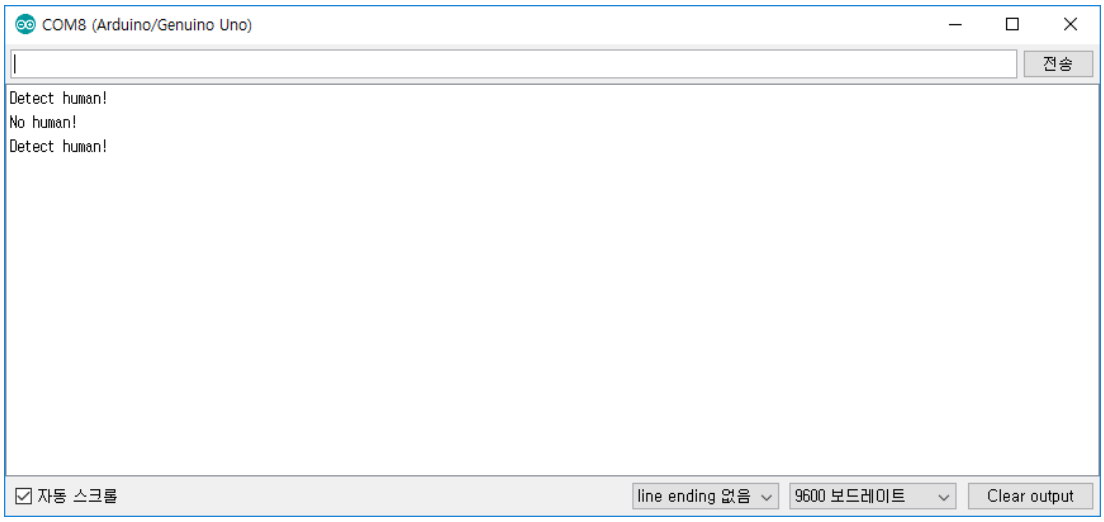
```
//*****************************************************************************
// Project : SR501 human detect in Advance set
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
#define  DETECT   8
void setup() {
  // initialize the digital pin as an output.
  pinMode(DETECT, INPUT_PULLUP);
  Serial.begin(9600);
}
void loop() {
```

```
  int sensorValue = digitalRead(DETECT);
  static int flag=0;

  if(sensorValue == 1 && flag==0 ){
     digitalWrite(13, HIGH);   // set the LED on
     Serial.println("Detect human!");
     flag=1;
  }else if(sensorValue == 0 && flag==1){
     Serial.println("No human!");
    flag=0;
  }
}
```

- **결과**

인체가 감지되면 Detect 신호를 보냅니다.

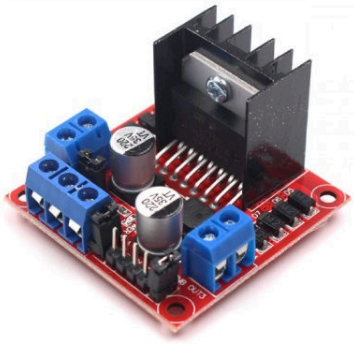


○ DC 모터 동작하기

- 개요



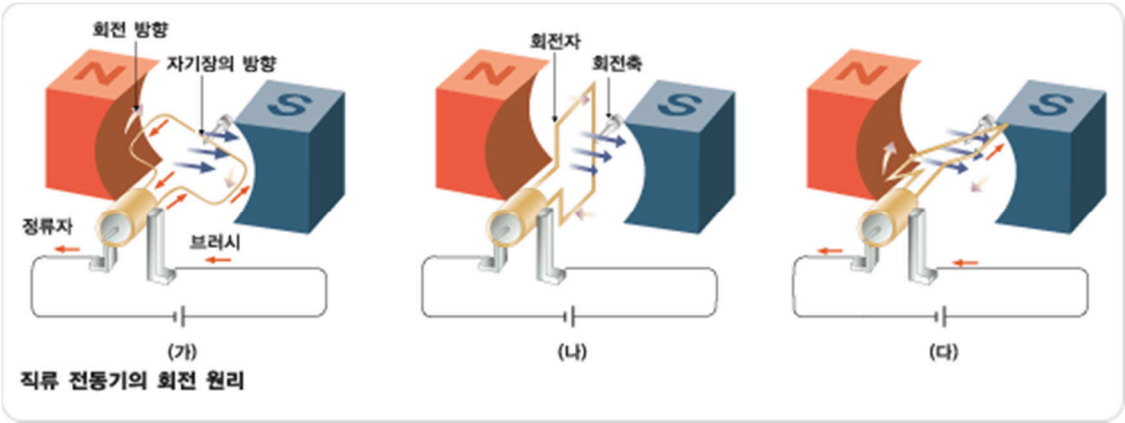
[DC 모터 및 바퀴]

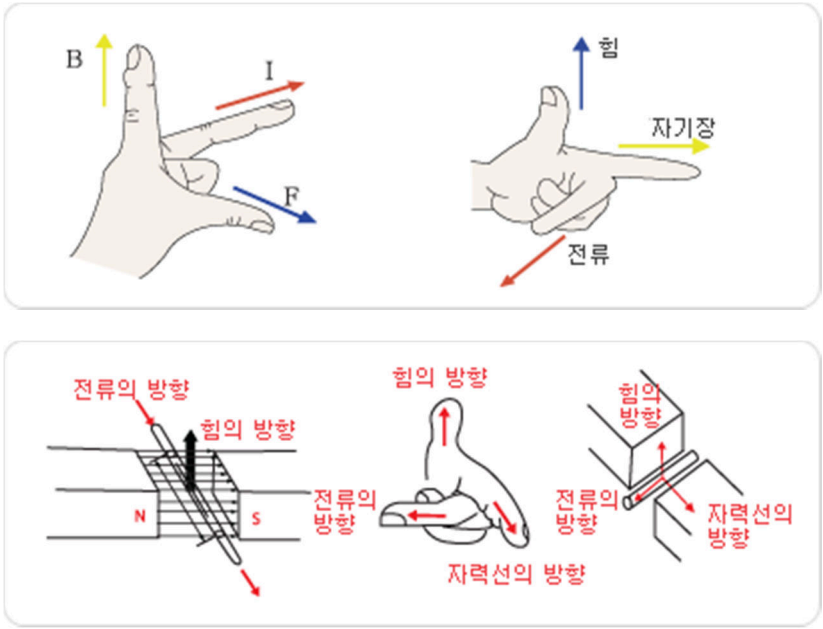


[L298n 모터드라이버]

위 사진 왼쪽의 DC 모터를 먼저 설명 드리면,

일반적인 장난감등에 많이 사용하는 DC(직류)모터 입니다. 교류 모터와 다르게 +극과 –극에 직류전원만 인가하게 되면 내부에 고정된 영구자석과 코일이 자기장의 상호 반발력을(플레밍의 왼손법칙) 이용해서 모터가 회전하도록 되어있습니다. 또한 기어박스가 내장되어있어 회전축이 90 도 방향으로 틀어져 위치하고 있습니다.





(출처 http://kimmy-ro.kimm.re.kr/learn/lea_environment_v10.html)

L298N 모터 드라이버 모듈은 Dual H-Bridge 모터 드라이버 모듈입니다.

쉽게 표현하면 DC 모터의 방향과 속도를 제어하기 위한 모듈입니다. 출력은 최대 2A 까지 가능합니다. 최대 출력을 사용할 경우 발열 높을 수 있습니다.

모듈 사진에서 실제 모터를 제어하는 IC 는 중앙에 방열판이 붙어있는 L298N IC 이고 입니다.

L298N 모듈은 2 개의 DC 모터와 1 개의 스텝모터를 동작할 수 있습니다. DC 모터의 정회전, 역회전을 제어할 수 있으며, ENA, ENB 의 핀을 제거하고 아두이노 PWM 단자에 연결하면 속도제어도 가능합니다. 아두이노에서는 analogWrite 를 사용하여 PWM 을 0~255 단계로 모터의 속도를 변경하시면 됩니다.

L298N 모듈 사양은 다음과 같습니다.

- 드라이버 칩 : L298N 듀얼 H-브리지 드라이버 칩
- 드라이브 부분의 공급전압 VMS : +5 V ~ 35 V-
- 드라이브 부분의 최대전류 : 2A / bridge
- 로직 부분의 공급전압 VSS : 4.5-5 0.5 V

- 로직 부분의 작동 전류 범위 :0 ~ 36mA
- 입력제어신호 전압 범위 : H: 4.5~ 5.5V / L: 0V
- 최대 소비 전력 : 20W
- 보관 온도 : -25 ℃까지 130 ℃
- 드라이버 보드 크기 : 55mm * 60mm * 30mm
- 드라이버 보드 무게 : 33g
- 기타 기능 : 방향 제어 표시 LED, 전원 표시 LED

- 준비물

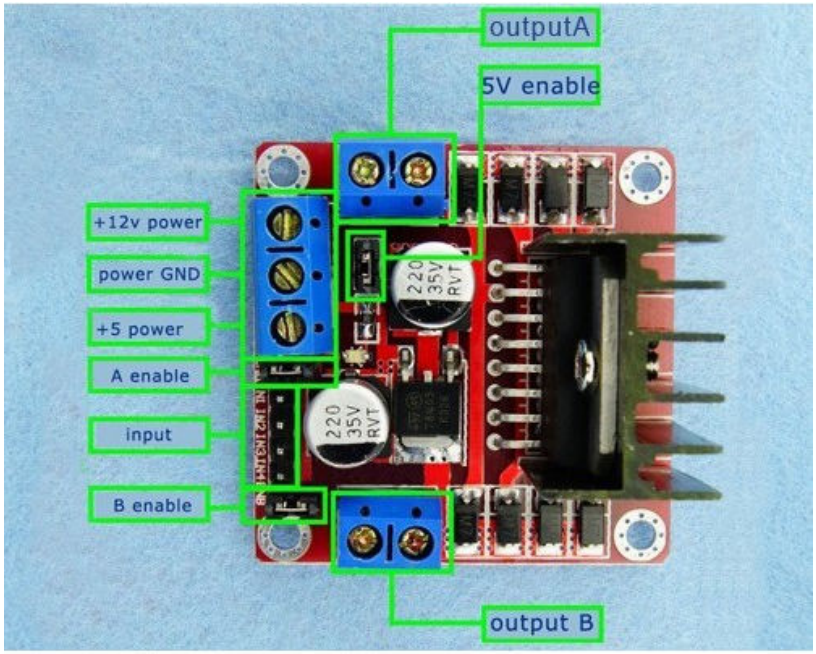
. 아두이노 우노(호환보드) X 1

. L298n DC 모터 드라이버 X 1

. DC 모터 및 바퀴 X 2

. 점퍼와이어

- 연결도



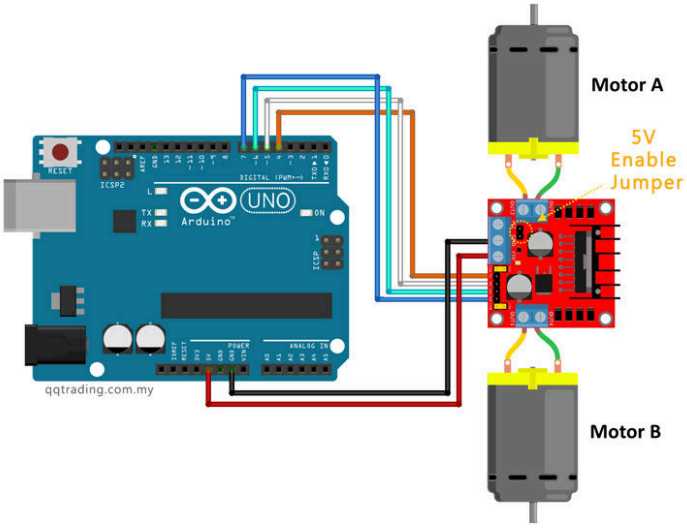
(출처: http://forum.arduino.cc/index.php?topic=210045.0)

다음은 모듈의 외부 인터페이스를 설명한 표입니다.

신호	설명
+12V	12V입력단자(선택)
GND	GND
+5V	5V입력단자(선택)
A_enable	출력 A Enable 탭 제거 후 PWM 입력시 속도조절가능
Input	모터 A,B 방향 제어
B_enable	출력 B Enable 탭 제거 후 PWM 입력시 속도조절가능
5V_enable	5V 전압 사용 설정(탭 없으면 12V 사용)
outputA	모터 A 출력
outputB	모터 B 출력

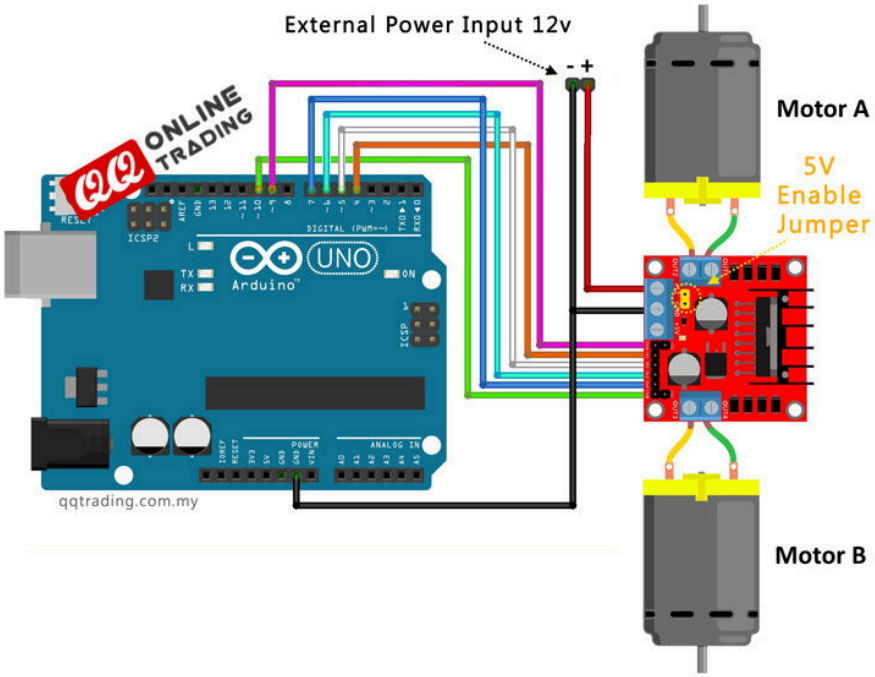
가장먼저 "5V_enable"로 사용할 전압을(5V/12V) 선택해야 합니다. 5V 경우 모터의 토크(힘)가 부족할 수 있습니다.

아래 연결 도는 5V 로 운용하고 속도변화 없이 모터 회전 방향만 제어하는 연결도 입니다.(5V_enable/ A_enable / B_enable 모두 탭이 연결된 상태)



(출처 : http://qqtrading.com.my/stepper-motor-driver-module-L298N)

다음그림의 예제 연결도는 외부 12V 를 입력하고 PWM 으로 속도를 제어할 수 있는 연결도 입니다. (5V_enable/ A_enable / B_enable 모두 탭을 제거한 상태)



- 코딩

다음 예제 코드는 위 두번째 연결도에서 5V 전압을 인가하고, 5V_enable/ A_enable / B_enable 모두 탭을 제거한 상태 입니다.

Setup 에서는 포트의 입출력만 설정하고, loop 에서 2 초단위로 모터의 회전 방향을 변경합니다.

analogWrite(speedPinA, 255); 값을 조정하면 모터 회전 속도가 변화하는 것을 확인할 수 있습니다.

```
//*********************************************************************************
// Project : DC Motor control l298n in advanced set
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
// Define Pins
int IN1 = 4;
int IN2 = 5;
int speedPinA = 9; // Needs to be a PWM pin to be able to control motor speed
int IN3 = 6;
int IN4 = 7;
int speedPinB = 10; // Needs to be a PWM pin to be able to control motor speed

void setup()
```

```
{
 // Set the output pins
   pinMode(IN1, OUTPUT);
   pinMode(IN2, OUTPUT);
   pinMode(IN3, OUTPUT);
   pinMode(IN4, OUTPUT);
   pinMode(speedPinA,OUTPUT);
   pinMode(speedPinB,OUTPUT);
}

void loop()
{
 analogWrite(speedPinA, 255); // Sets speed variable via PWM
 analogWrite(speedPinB, 255); // Sets speed variable via PWM
 // Rotate the Motor A clockwise
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, LOW);
 delay(2000);
 // Motor A
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, HIGH);
 delay(500);

 // Rotate the Motor B clockwise
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, LOW);
 delay(2000);
 // Motor B
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, HIGH);
 delay(500);


 // Rotates the Motor A counter-clockwise
 digitalWrite(IN1, LOW);
 digitalWrite(IN2, HIGH);
 delay(2000);
 // Motor A
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, HIGH);
 delay(500);

 // Rotates the Motor B
 digitalWrite(IN3, LOW);
 digitalWrite(IN4, HIGH);
 delay(2000);
 // Motor B
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, HIGH);
 delay(500);
}
```

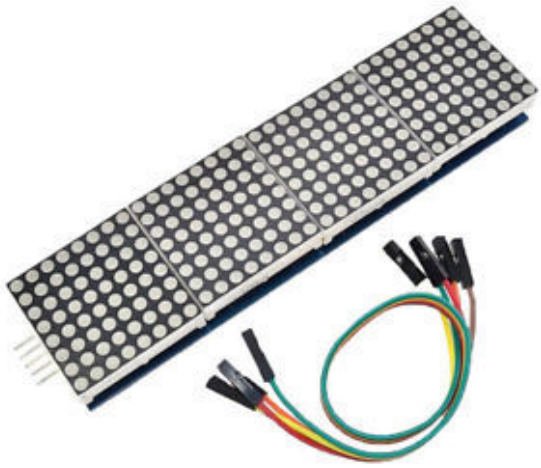
- **결과**

모터와 바퀴가 좌우로 회전하는 것을 확인할 수 있습니다.

PWM 을 변경할 경우 속도변화를 확인할 수 있습니다.

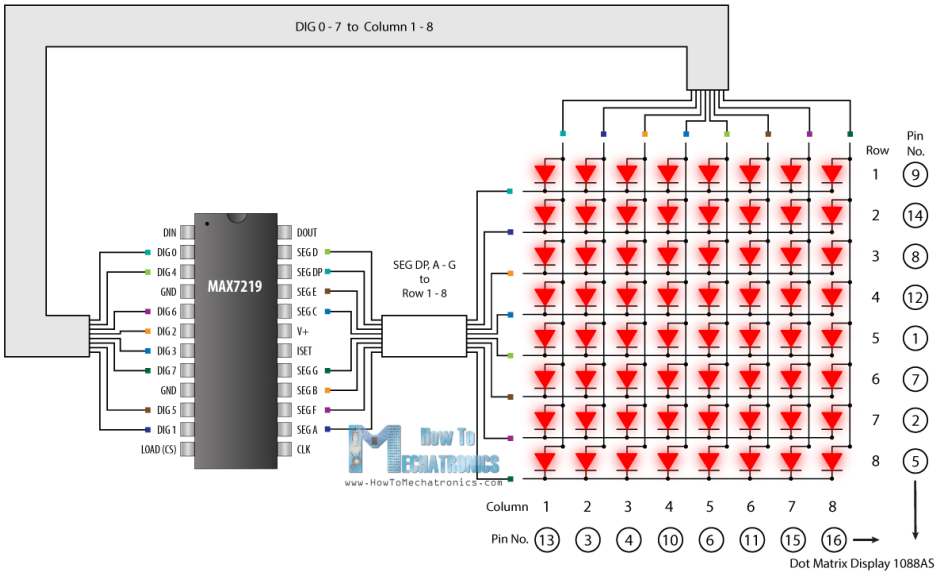
○ MAX7219 도트매트릭스

- 개요



MAX7219 IC 는 도트 매트릭스 또는 7-세그먼트와 같은 여러 개의 LED 들을 사용하기 위해서 Maxim 사에서 만든 LED 드라이버 IC 입니다. 그러므로 7-Segment 또는 LED bar 와 함께 사용하신다면 4 개(SPI)의 핀만으로도 수많은 LED 들의 On/Off 동작을 쉽게 제어하실 수 있습니다.

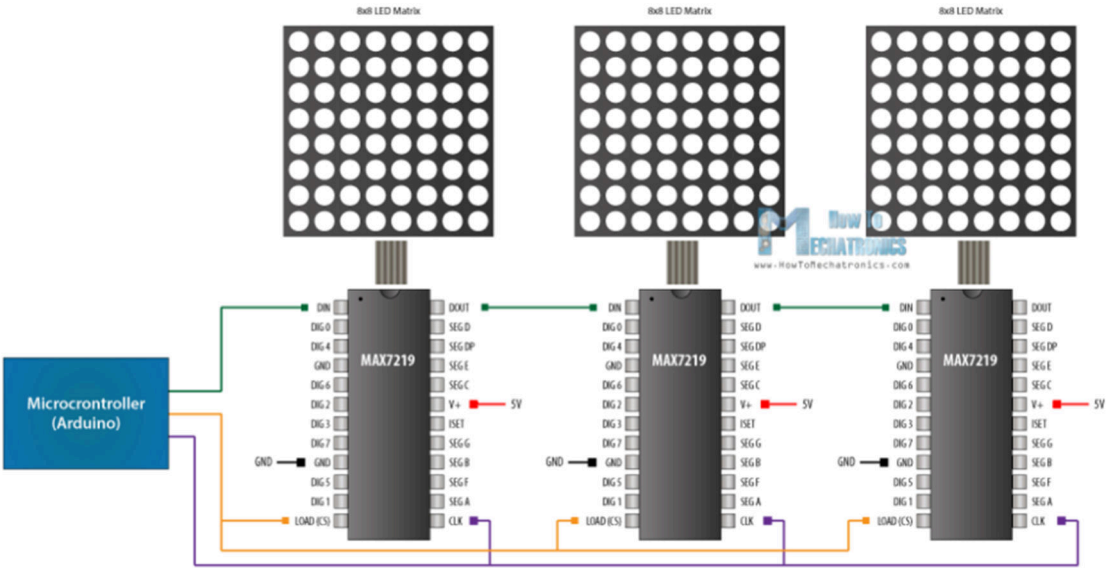
MAX7219 는 아래 사진과 같이 Dig 핀과 Seg 핀의 조합으로 총 64 개의 LED 를 제어할 수 있습니다.



(출처 : http://howtomechatronics.com)

Max7219 는 Daisy chain 을 지원하기 때문에 아래와 같이 여러 개 max7219 를 연결해서 동시에 제어가 가능합니다.

이번에 사용할 도트매트릭스는 4 개의 MAX7219 가 Daisy chain 로 연결이 되어 있는 것입니다.



MAX7219 데이터 쉬트를 잠시 살펴보면

첫 페이지에서 최대 64 개 LED 까지 지원이 가능하고, 내부에 8x8 static memory 를 가지고 있으므로 LED 상태 값이 유지할 수 있다고 되어있습니다.
또한 EMI 를 고려한 제품으로 SPI 를 지원하도록 되어있다고 적혀있습니다.

다시 설명 드리면 MAX7219 IC 는 매트릭스 방식을 상용하기 때문에 한 개 드라이버 IC 에서 총 64 개의 LED 를 까지 제어할 수 있다는 것입니다. 매트릭스 방식은 LED 를 동시에 켤 수 없습니다. LED 빠르게 On/Off 를 반복하는 것이기 때문에 사람의 눈에는 동시 켜지는 것으로 보입니다.

maxim integrated™

MAX7219/MAX7221

Serially Interfaced, 8-Digit LED Display Drivers

General Description

The MAX7219/MAX7221 are compact, serial input/output common-cathode display drivers that interface microprocessors (µPs) to 7-segment numeric LED displays of up to 8 digits, bar-graph displays, or 64 individual LEDs. Included on-chip are a BCD code-B decoder, multiplex scan circuitry, segment and digit drivers, and an 8x8 static RAM that stores each digit. Only one external resistor is required to set the segment current for all LEDs. The MAX7221 is compatible with SPI™, QSPI™, and MICROWIRE™, and has slew-rate-limited segment drivers to reduce EMI.

A convenient 4-wire serial interface connects to all common µPs. Individual digits may be addressed and updated without rewriting the entire display. The MAX7219/MAX7221 also allow the user to select code-B decoding or no-decode for each digit.

The devices include a 150µA low-power shutdown mode, analog and digital brightness control, a scan-limit register that allows the user to display from 1 to 8 digits, and a test mode that forces all LEDs on.

For applications requiring 3V operation or segment blinking, refer to the MAX6951 data sheet.

Applications

Bar-Graph Displays
Industrial Controllers
Panel Meters
LED Matrix Displays

Features

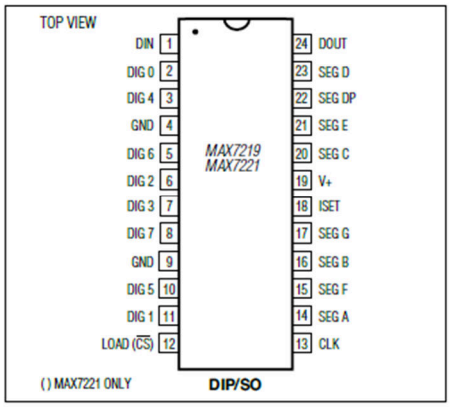
- 10MHz Serial Interface
- Individual LED Segment Control
- Decode/No-Decode Digit Selection
- 150µA Low-Power Shutdown (Data Retained)
- Digital and Analog Brightness Control
- Display Blanked on Power-Up
- Drive Common-Cathode LED Display
- Slew-Rate Limited Segment Drivers for Lower EMI (MAX7221)
- SPI, QSPI, MICROWIRE Serial Interface (MAX7221)
- 24-Pin DIP and SO Packages

Ordering Information

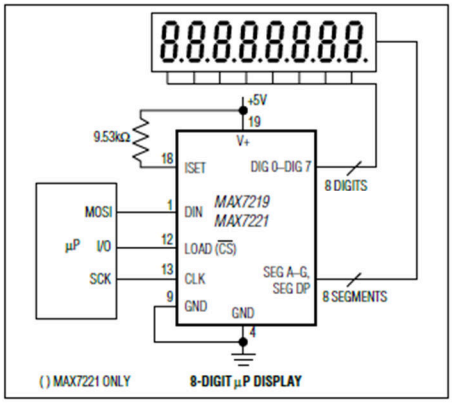
PART	TEMP RANGE	PIN-PACKAGE
MAX7219CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX7219CWG	0°C to +70°C	24 Wide SO
MAX7219C/D	0°C to +70°C	Dice*
MAX7219ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX7219EWG	-40°C to +85°C	24 Wide SO
MAX7219ERG	-40°C to +85°C	24 Narrow CERDIP

Ordering Information continued at end of data sheet.
Dice are specified at T_A = +25°C.

Pin Configuration



Typical Application Circuit



SPI and QSPI are trademarks of Motorola Inc. MICROWIRE is a trademark of National Semiconductor Corp.

For pricing, delivery, and ordering information, please contact Maxim Direct at 1-888-629-4642, or visit Maxim's website at www.maximintegrated.com.

19-4452; Rev 4; 7/03

다음 그림의 인터페이스 타이밍을 보면,
1 개 frame 은 총 16bit Data 필드로 구성이 되어있습니다.

하위 8bit 는 data 로 이루어져 있고, 상위는 4bit address 로 되어있습니다.
(D12~D15 는 사용 안 합니다.)

MAX7219/MAX7221

Serially Interfaced, 8-Digit LED Display Drivers

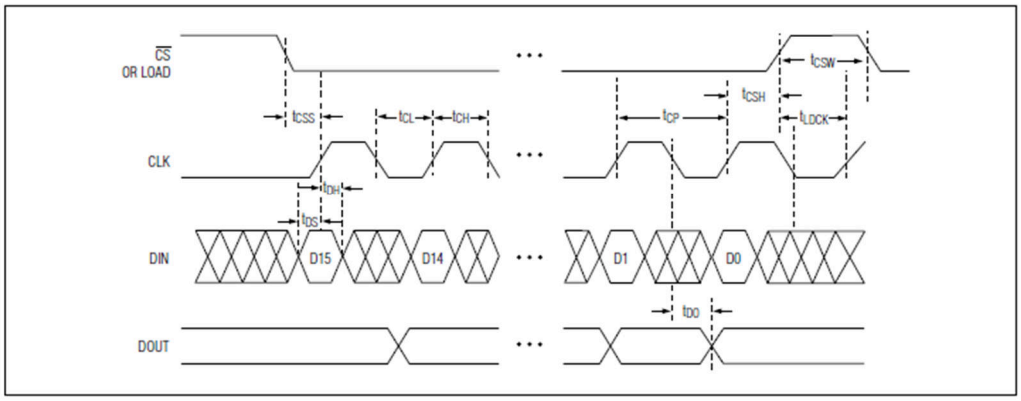


Figure 1. Timing Diagram

Table 1. Serial-Data Format (16 Bits)

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	ADDRESS				MSB	DATA						LSB

이전에 말씀 드린 것 처럼
MAX7219 는 모든 LED 를 한 번에 켜는 것이 아니고 8 개 단위로 빠르게 On/Off 를 제어하면서 LED 를 표시하는 드라이버 IC 입니다.
하지만 속도가 매우 빠르기 때문에 우리 눈에는 64 개 LED 가 동시에 켜지는 것 처럼 보입니다.

모듈 사양은 다음과 같습니다.

- Operating Voltage: 5V
- A single module can drive a 8x8 dot matrix common cathode
- Dimensions: 12.8X12.8X1.3 cm
- Fixing screws with 64 holes with a diameter 3mm
- Module with input and output interfaces, support for cascading multiple module

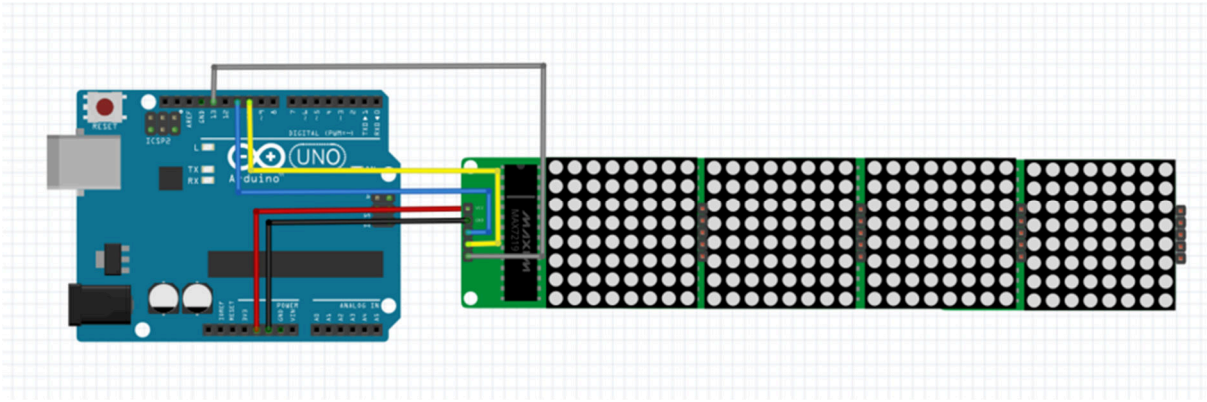
- 준비물

. 아두이노 우노(호환보드) X 1

. MAX7219 4 개 Dot matrix X 1

. 점퍼와이어

- **연결도**



- **코딩**

다음 두 개의 SW 라이브러리 설치가 필요합니다.

MAX7219_Dot_Matrix / bitBangedSPI

Max7219_Dot_matrix 는 MAX7219 제어용 라이브러리이고, bitBangedSPI 는 SPI 인터페이스 라이브러리입니다.

```
//*************************************************************************
// Project : Max7219 4 dot matrix in Advanced set
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
#include <bitBangedSPI.h>
#include <MAX7219_Dot_Matrix.h>
const byte chips = 12;
// 12 chips (display modules), hardware SPI with load on D10
MAX7219_Dot_Matrix display (chips, 10);  // Chips / LOAD
const char message [] = "Testing 1234567890";

void setup ()
{
  display.begin ();
}  // end of setup

unsigned long lastMoved = 0;
unsigned long MOVE_INTERVAL = 20;  // mS
int  messageOffset;

void updateDisplay ()
{
  display.sendSmooth (message, messageOffset);

  // next time show one pixel onwards
  if (messageOffset++ >= (int) (strlen (message) * 8))
    messageOffset = - chips * 8;
}  // end of updateDisplay
```

```
void loop ()
{

  // update display if time is up
  if (millis () - lastMoved >= MOVE_INTERVAL)
    {
    updateDisplay ();
    lastMoved = millis ();
    }
  // do other stuff here

}  // end of loop
```

MAX7219_Dot_Matrix.h 라이브러리는 자체 폰트 데이터를 가지고 있고, 여러 개의 도트 매트릭스 연결이 가능합니다. 애니메이션 효과로 우->좌 데이터 이동이 가능합니다.

아두이노 SPI 인터페이스를 같이 사용하므로 SS pin 만 아래와 같이 설정합니다.

MAX7219_Dot_Matrix display (chips, 10);

내부에 폰트가 내장되어있으므로 표시하고자 하는 글자 / 숫자 / 특수기호를 배열로 저장하시면 됩니다.

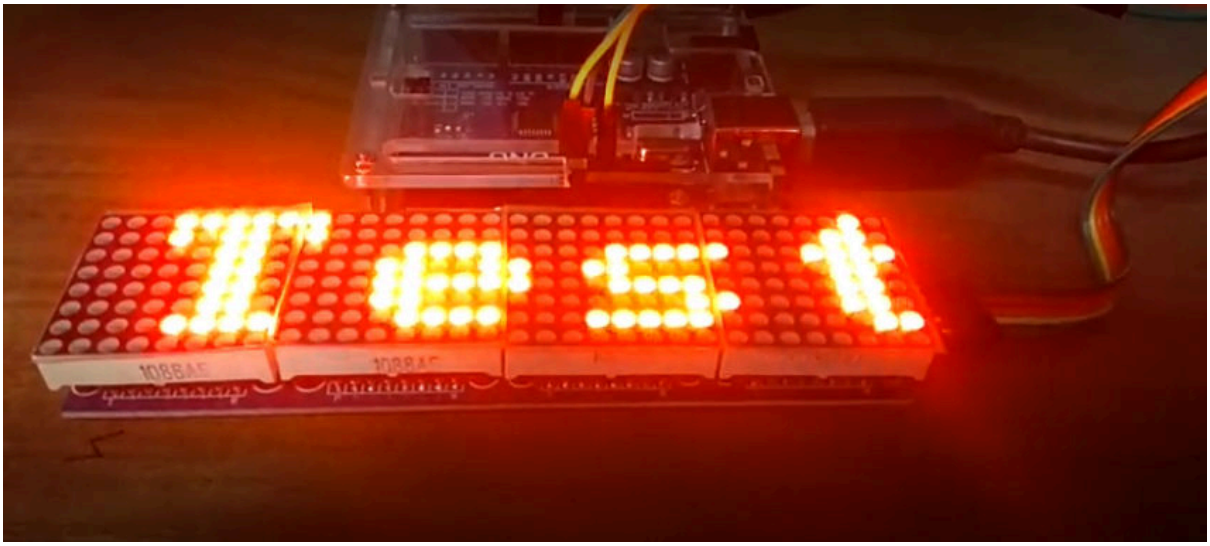
const char message [] = "testing 123456789 ";

updateDisplay ()에서는 애니메이션 효과를 위해서 offset 을 계산합니다.

display.sendSmooth (message, messageOffset);

- **결과**

설정된 글자가 도트매트릭스에서 좌에서 우로 이동되는 것을 확인할 수 있습니다.

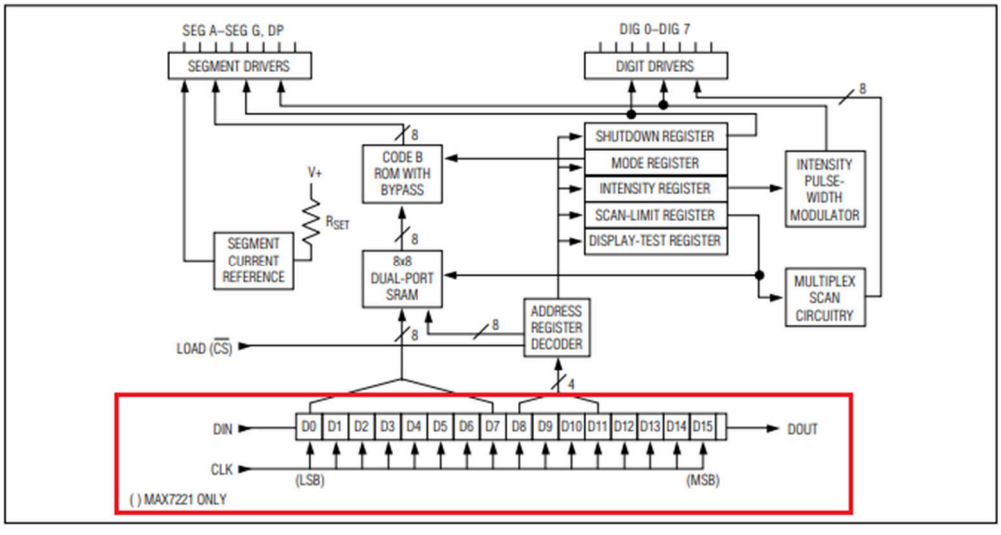


*추가 설명 (Daisy chain 에 대해서)

하나의 SPI 인터페이스에서 4 개의 도트매트릭스 max7219 를 제어하는 원리는

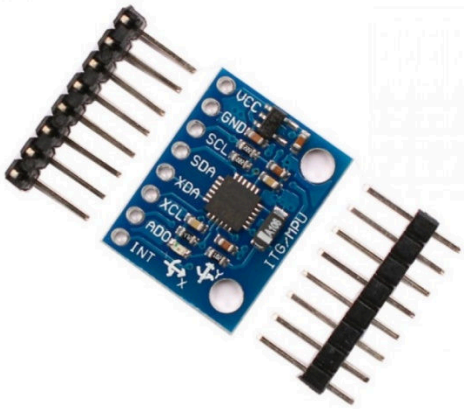
Din 과 Dout 신호에 있습니다.

모든 데이터는 Din 으로 들어오고 17 번째 bit 부터는 Shift 되어 다시 Dout 으로 나가게 되어있습니다. Dout 신호는 다른 매트릭스 Din 으로 연결이 되어있습니다.
그리고 CS 는 Load 개념입니다. CS 가 LOW 에서 HIGH 되는 순간 데이터가 Latch 가 되는 것입니다.
아래 데이터쉬트의 붉은색 그림을 확인하시면 됩니다.



○ 3축 가속/ 자이로센서

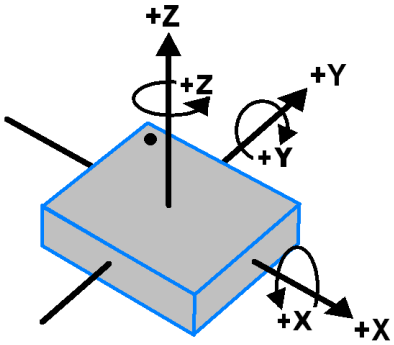
- 개요



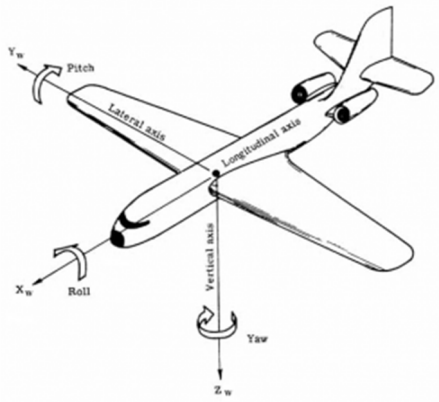
MPU 6050 은 자이로센서와 가속도센서가 결합된 형태의 모듈입니다. 자이로센서는 X,Y,Z 의 3 축의 방향 변환을 감지하고, 가속도센서는 X,Y,Z 방향으로의 움직임의 변화율을 제공합니다.

주로 사용하는 제품이나 프로젝트들은 드론, 모션인식, 무선자동차핸들 등이 있습니다. 사실 우리가 사용하는 모든 스마트폰에서 유사한 센서가 장착되어 있어 화면 회전과 같은 기능에 사용되고 있습니다.

기능만 보면, 단순히 방향과 속도를 측정하는 모듈이지만 실제 사용하는 환경에서는 많은 수학적 공식이 필요합니다. 이번 예제에서는 수학적 공식을 사용해야 하는 예제들을 제외하고 간단한 라이브러리 이용하여 드론 프로젝트에 많이 사용하는 DMP pitch roll yaw 값을 연속적으로 읽어 보도록 하겠습니다.



MPU-6050
Orientation & Polarity of Rotation

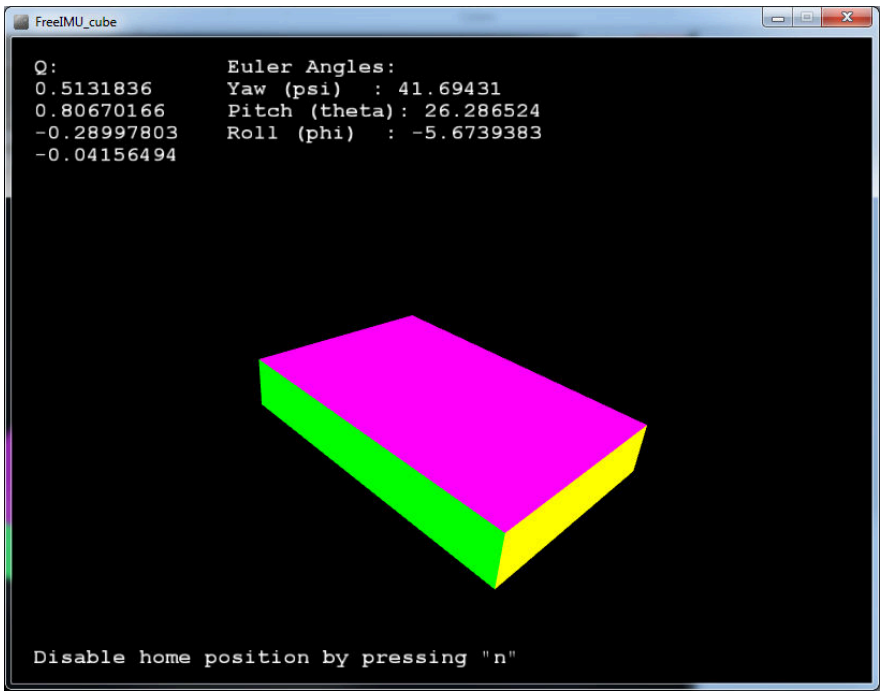


MPU 6050 주요 사양표은 아래와 같습니다.

사양	내 용	단위	부 가 설 명
동작전압	5V ~ 3.3V		
통신 방식	I²C(I2C,IIC) , SPI		
자이로 최대 측정값	±(250/500/1000/2000)	(*/sec)	초당 회전되는 자이로는 각속도
자이로 센서 감도	(131/65.5/32.8/16.4)	(LSB/*/sec)	초당 감지되는 감도, 값이높을수록 감도 높음
자이로 잡음(오차)	0.01	(dps/√Hz)	오차율
가속도 센서 최대값	±(2/4/8/16)	(g)	가속도 최대값
자이로 센서 감도	(16384/8192/4096/2048)	(LSB/g)	값이 클수록 감도가 높음

센서감도를 조정하게 되면, 미세하게 측정할 것 인지 빠른 움직임 측정할 것인 것 결정할 수 있습니다.

추가로 아두이노와 "Processing"이라는 프로그램을 같이 사용하시면 아래 그림과 같이 3 차원 그래픽으로 움직임을 시뮬레이션 할 수 있습니다.



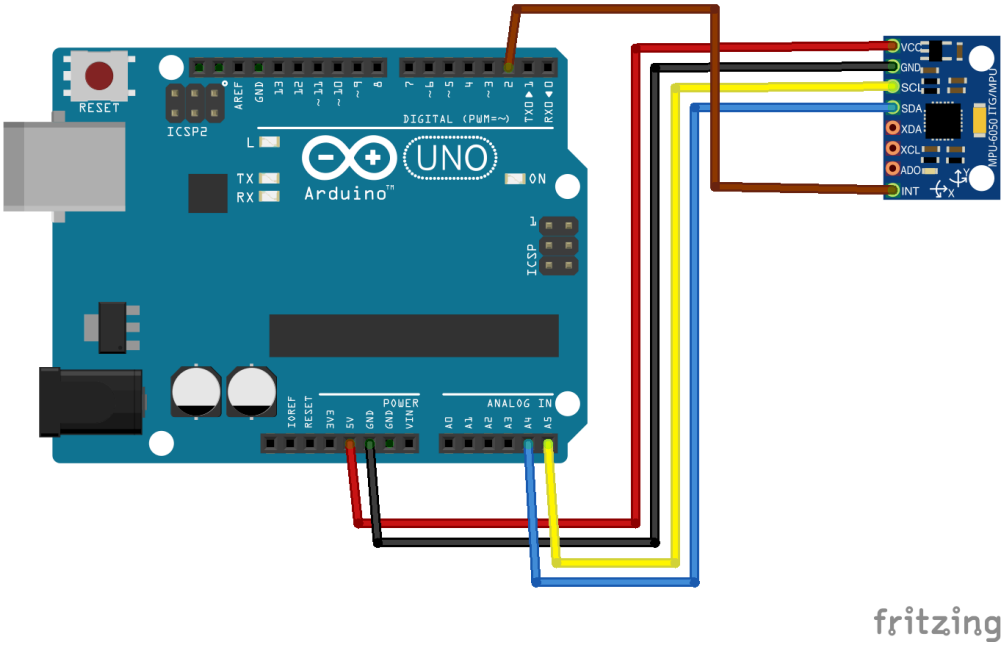
- **준비물**

. 아두이노 우노(호환보드) X 1

. MPU6050 X 1

. 점퍼와이어

- **연결도**

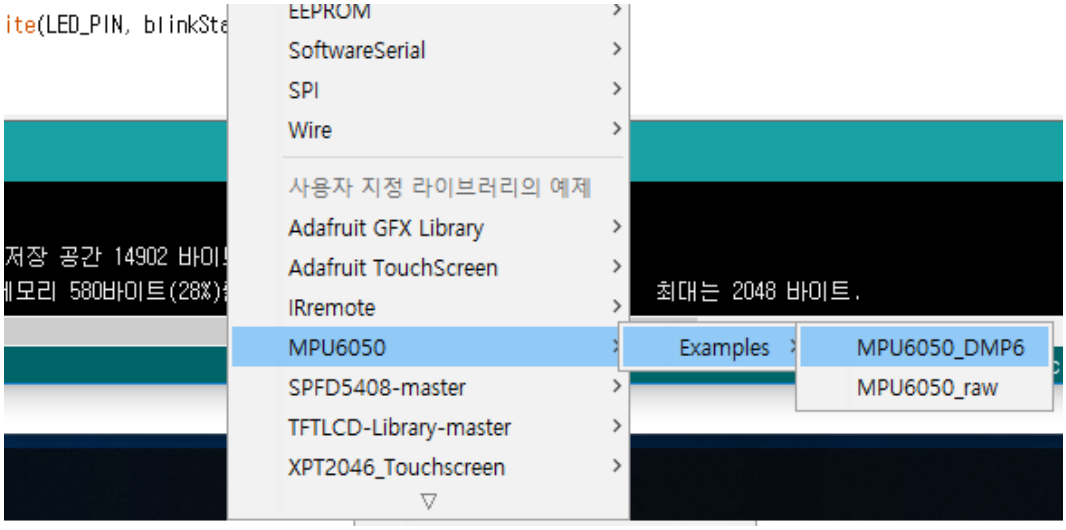


- **예제 실행 및 확인**

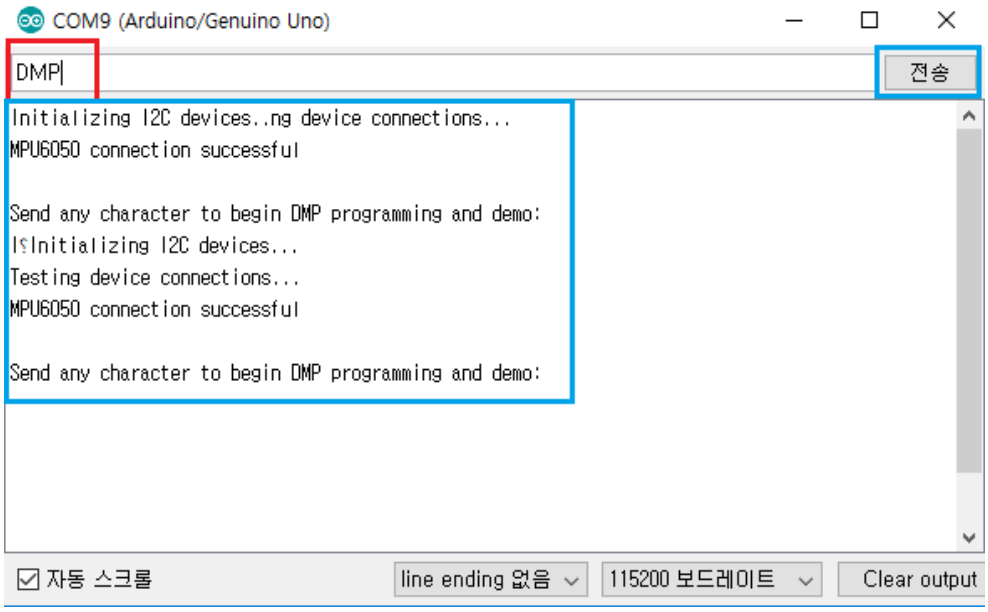
MPU6050 의 칩 내부 레지스터를 사용하는 방법은 복잡합니다. 여기서는 MPU6050 라이브러리를 사용해서 움직임을 감지할 수 있는 센서 값만 모니터링 해보도록 하겠습니다.

라이브러리는 MPU6050 과 I2Cdev 를 설치하셔야 합니다.

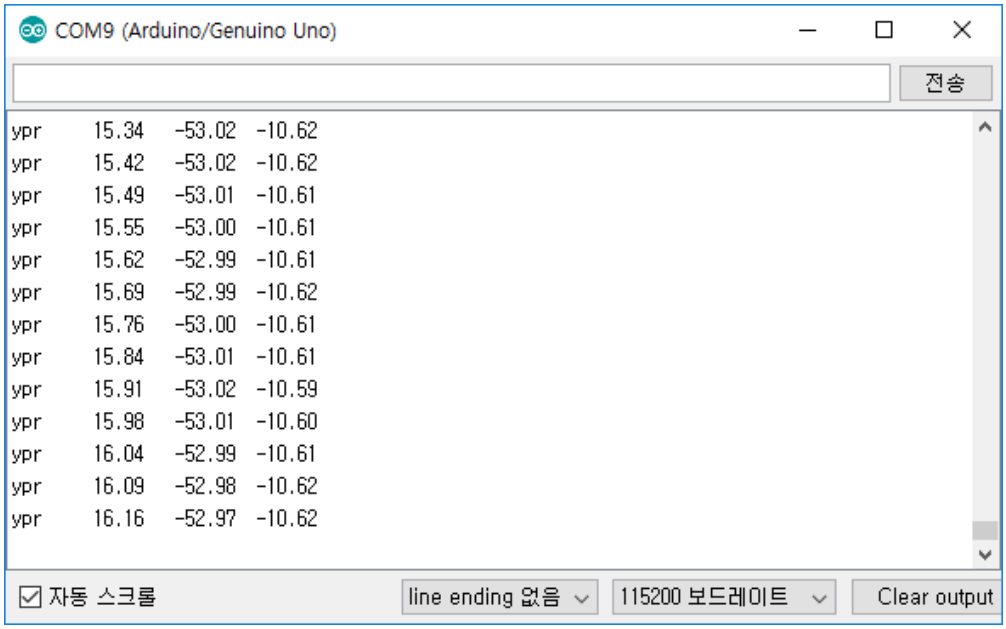
라이브러리가 설치 되셨다면, 아래 그림과 같이 MPU6050_DMP 를 예제를 불러옵니다.



예제를 빌드 / 업로드를 실행하고, 시리얼 모니터를 실행합니다. 아래와 같은 초기화 완료 문구가 나옵니다. 붉은색으로 표시된 전송 창에서 "DMP"이라고, 입력하고 전송을 누릅니다.



"DPM" 입력이 완료되면 다음 그림처럼 pitch, roll, yaw 값이 연속적으로 나오는 것을 확인 할 수 있습니다. 이제 센서를 움직여서 값의 변화를 확인하시면 됩니다.



- 예제 코드 및 설명

아래 예제 코드에 대해서 몇 가지 함수와 API 들에 대해서 설명을 드리겠습니다.

MPU6050 mpu;

MPU6050 Class 객체를 등록합니다. 주석에 나와있는 것 같이 Address 를 추가해서 MPU6050 mpu(0x69); 이런 방식으로 사용하셔도 됩니다.

mpu.initialize();

MPU6050 를 초기화 하는 과정입니다.

mpu.testConnection();

MPU6050 연결 여부를 확인합니다. fail 시 아두이노와의 연결 부분과 납땜 상태를 확인하셔야 합니다.

mpu.dmpInitialize();

DPM 초기화 과정입니다.

DMP 는 Digital Motion Processor 의 약자로 MPU6050 에서 자체적으로 제공하는 자이로 가속도 연산 유닛입니다.

계산을 이용해서 pitch roll yaw 값과 쿼터니안, 오일러값을 얻을 수 있습니다.

이후 아래 코드부터는 매크로 처리되어 일부 코드만 동작을 하도록 되어 있습니다.

저희가 사용하는 코드는 OUTPUT_READABLE_YAWPITCHROLL 매크로 입니다.

```
// I2C device class (I2Cdev) demonstration Arduino sketch for MPU6050 class using
DMP (MotionApps v2.0)
// 6/21/2012 by Jeff Rowberg <jeff@rowberg.net>
// Updates should (hopefully) always be available at
https://github.com/jrowberg/i2cdevlib
//
// Changelog:
//      2013-05-08 - added seamless Fastwire support
//                 - added note about gyro calibration
//      2012-06-21 - added note about Arduino 1.0.1 + Leonardo compatibility error
//      2012-06-20 - improved FIFO overflow handling and simplified read process
//      2012-06-19 - completely rearranged DMP initialization code and
simplification
//      2012-06-13 - pull gyro and accel data from FIFO packet instead of reading
directly
//      2012-06-09 - fix broken FIFO read sequence and change interrupt detection
to RISING
//      2012-06-05 - add gravity-compensated initial reference frame acceleration
output
//                 - add 3D math helper file to DMP6 example sketch
//                 - add Euler output and Yaw/Pitch/Roll output formats
//      2012-06-04 - remove accel offset clearing for better results (thanks
Sungon Lee)
//      2012-06-01 - fixed gyro sensitivity to be 2000 deg/sec instead of 250
//      2012-05-30 - basic DMP initialization working
/* ============================================
I2Cdev device library code is placed under the MIT license
Copyright (c) 2012 Jeff Rowberg
Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE SOFTWARE.
===============================================
*/
// I2Cdev and MPU6050 must be installed as libraries, or else the .cpp/.h files
// for both classes must be in the include path of your project
#include "I2Cdev.h"
#include "MPU6050_6Axis_MotionApps20.h"
//#include "MPU6050.h" // not necessary if using MotionApps include file
```

```
// Arduino Wire library is required if I2Cdev I2CDEV_ARDUINO_WIRE implementation
// is used in I2Cdev.h
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
    #include "Wire.h"
#endif
// class default I2C address is 0x68
// specific I2C addresses may be passed as a parameter here
// AD0 low = 0x68 (default for SparkFun breakout and InvenSense evaluation board)
// AD0 high = 0x69
MPU6050 mpu;
//MPU6050 mpu(0x69); // <-- use for AD0 high
/* =========================================================================
   NOTE: In addition to connection 3.3v, GND, SDA, and SCL, this sketch
   depends on the MPU-6050's INT pin being connected to the Arduino's
   external interrupt #0 pin. On the Arduino Uno and Mega 2560, this is
   digital I/O pin 2.
 * ========================================================================= */
/* =========================================================================
   NOTE: Arduino v1.0.1 with the Leonardo board generates a compile error
   when using Serial.write(buf, len). The Teapot output uses this method.
   The solution requires a modification to the Arduino USBAPI.h file, which
   is fortunately simple, but annoying. This will be fixed in the next IDE
   release. For more info, see these links:
   http://arduino.cc/forum/index.php/topic,109987.0.html
   http://code.google.com/p/arduino/issues/detail?id=958
 * ========================================================================= */
// uncomment "OUTPUT_READABLE_QUATERNION" if you want to see the actual
// quaternion components in a [w, x, y, z] format (not best for parsing
// on a remote host such as Processing or something though)
//#define OUTPUT_READABLE_QUATERNION
// uncomment "OUTPUT_READABLE_EULER" if you want to see Euler angles
// (in degrees) calculated from the quaternions coming from the FIFO.
// Note that Euler angles suffer from gimbal lock (for more info, see
// http://en.wikipedia.org/wiki/Gimbal_lock)
//#define OUTPUT_READABLE_EULER
// uncomment "OUTPUT_READABLE_YAWPITCHROLL" if you want to see the yaw/
// pitch/roll angles (in degrees) calculated from the quaternions coming
// from the FIFO. Note this also requires gravity vector calculations.
// Also note that yaw/pitch/roll angles suffer from gimbal lock (for
// more info, see: http://en.wikipedia.org/wiki/Gimbal_lock)
#define OUTPUT_READABLE_YAWPITCHROLL
// uncomment "OUTPUT_READABLE_REALACCEL" if you want to see acceleration
// components with gravity removed. This acceleration reference frame is
// not compensated for orientation, so +X is always +X according to the
// sensor, just without the effects of gravity. If you want acceleration
// compensated for orientation, us OUTPUT_READABLE_WORLDACCEL instead.
//#define OUTPUT_READABLE_REALACCEL
// uncomment "OUTPUT_READABLE_WORLDACCEL" if you want to see acceleration
// components with gravity removed and adjusted for the world frame of
// reference (yaw is relative to initial orientation, since no magnetometer
// is present in this case). Could be quite handy in some cases.
//#define OUTPUT_READABLE_WORLDACCEL
// uncomment "OUTPUT_TEAPOT" if you want output that matches the
// format used for the InvenSense teapot demo
//#define OUTPUT_TEAPOT
#define LED_PIN 13 // (Arduino is 13, Teensy is 11, Teensy++ is 6)
bool blinkState = false;
// MPU control/status vars
bool dmpReady = false;  // set true if DMP init was successful
uint8_t mpuIntStatus;   // holds actual interrupt status byte from MPU
uint8_t devStatus;      // return status after each device operation (0 =
success, !0 = error)
uint16_t packetSize;    // expected DMP packet size (default is 42 bytes)
```

```
uint16_t fifoCount;     // count of all bytes currently in FIFO
uint8_t fifoBuffer[64]; // FIFO storage buffer
// orientation/motion vars
Quaternion q;           // [w, x, y, z]         quaternion container
VectorInt16 aa;         // [x, y, z]            accel sensor measurements
VectorInt16 aaReal;     // [x, y, z]            gravity-free accel sensor
measurements
VectorInt16 aaWorld;    // [x, y, z]            world-frame accel sensor
measurements
VectorFloat gravity;    // [x, y, z]            gravity vector
float euler[3];         // [psi, theta, phi]    Euler angle container
float ypr[3];           // [yaw, pitch, roll]   yaw/pitch/roll container and
gravity vector
// packet structure for InvenSense teapot demo
uint8_t teapotPacket[14] = { '$', 0x02, 0,0, 0,0, 0,0, 0,0, 0x00, 0x00, '\r',
'\n' };
// ================================================================
// ===               INTERRUPT DETECTION ROUTINE                ===
// ================================================================
volatile bool mpuInterrupt = false;     // indicates whether MPU interrupt pin has
gone high
void dmpDataReady() {
    mpuInterrupt = true;
}
// ================================================================
// ===                      INITIAL SETUP                       ===
// ================================================================
void setup() {
    // join I2C bus (I2Cdev library doesn't do this automatically)
    #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
        Wire.begin();
        TWBR = 24; // 400kHz I2C clock (200kHz if CPU is 8MHz)
    #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
        Fastwire::setup(400, true);
    #endif
    // initialize serial communication
    // (115200 chosen because it is required for Teapot Demo output, but it's
    // really up to you depending on your project)
    Serial.begin(115200);
    while (!Serial); // wait for Leonardo enumeration, others continue immediately
    // NOTE: 8MHz or slower host processors, like the Teensy @ 3.3v or Ardunio
    // Pro Mini running at 3.3v, cannot handle this baud rate reliably due to
    // the baud timing being too misaligned with processor ticks. You must use
    // 38400 or slower in these cases, or use some kind of external separate
    // crystal solution for the UART timer.
    // initialize device
    Serial.println(F("Initializing I2C devices..."));
    mpu.initialize();
    // verify connection
    Serial.println(F("Testing device connections..."));
    Serial.println(mpu.testConnection() ? F("MPU6050 connection successful") :
F("MPU6050 connection failed"));
    // wait for ready
    Serial.println(F("\nSend any character to begin DMP programming and demo: "));
    while (Serial.available() && Serial.read()); // empty buffer
    while (!Serial.available());                 // wait for data
    while (Serial.available() && Serial.read()); // empty buffer again
    // load and configure the DMP
    Serial.println(F("Initializing DMP..."));
    devStatus = mpu.dmpInitialize();
    // supply your own gyro offsets here, scaled for min sensitivity
    mpu.setXGyroOffset(220);
    mpu.setYGyroOffset(76);
```

```
    mpu.setZGyroOffset(-85);
    mpu.setZAccelOffset(1788); // 1688 factory default for my test chip
    // make sure it worked (returns 0 if so)
    if (devStatus == 0) {
        // turn on the DMP, now that it's ready
        Serial.println(F("Enabling DMP..."));
        mpu.setDMPEnabled(true);
        // enable Arduino interrupt detection
        Serial.println(F("Enabling interrupt detection (Arduino external interrupt
0)..."));
        attachInterrupt(0, dmpDataReady, RISING);
        mpuIntStatus = mpu.getIntStatus();
        // set our DMP Ready flag so the main loop() function knows it's okay to
use it
        Serial.println(F("DMP ready! Waiting for first interrupt..."));
        dmpReady = true;
        // get expected DMP packet size for later comparison
        packetSize = mpu.dmpGetFIFOPacketSize();
    } else {
        // ERROR!
        // 1 = initial memory load failed
        // 2 = DMP configuration updates failed
        // (if it's going to break, usually the code will be 1)
        Serial.print(F("DMP Initialization failed (code "));
        Serial.print(devStatus);
        Serial.println(F(")"));
    }
    // configure LED for output
    pinMode(LED_PIN, OUTPUT);
}
// ================================================================
// ===                    MAIN PROGRAM LOOP                     ===
// ================================================================
void loop() {
    // if programming failed, don't try to do anything
    if (!dmpReady) return;
    // wait for MPU interrupt or extra packet(s) available
    while (!mpuInterrupt && fifoCount < packetSize) {
        // other program behavior stuff here
        // .
        // .
        // .
        // if you are really paranoid you can frequently test in between other
        // stuff to see if mpuInterrupt is true, and if so, "break;" from the
        // while() loop to immediately process the MPU data
        // .
        // .
        // .
    }
    // reset interrupt flag and get INT_STATUS byte
    mpuInterrupt = false;
    mpuIntStatus = mpu.getIntStatus();
    // get current FIFO count
    fifoCount = mpu.getFIFOCount();
    // check for overflow (this should never happen unless our code is too
inefficient)
    if ((mpuIntStatus & 0x10) || fifoCount == 1024) {
        // reset so we can continue cleanly
        mpu.resetFIFO();
        Serial.println(F("FIFO overflow!"));
    // otherwise, check for DMP data ready interrupt (this should happen
frequently)
    } else if (mpuIntStatus & 0x02) {
```

```
        // wait for correct available data length, should be a VERY short wait
        while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();
        // read a packet from FIFO
        mpu.getFIFOBytes(fifoBuffer, packetSize);

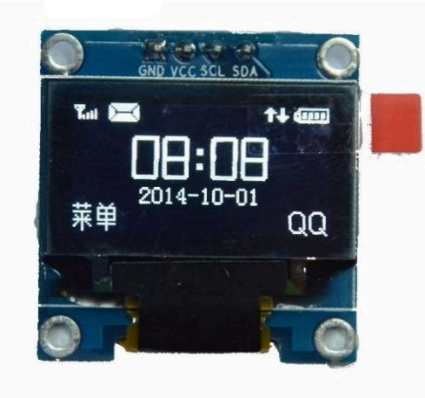
        // track FIFO count here in case there is > 1 packet available
        // (this lets us immediately read more without waiting for an interrupt)
        fifoCount -= packetSize;
        #ifdef OUTPUT_READABLE_QUATERNION
            // display quaternion values in easy matrix form: w x y z
            mpu.dmpGetQuaternion(&q, fifoBuffer);
            Serial.print("quat\t");
            Serial.print(q.w);
            Serial.print("\t");
            Serial.print(q.x);
            Serial.print("\t");
            Serial.print(q.y);
            Serial.print("\t");
            Serial.println(q.z);
        #endif
        #ifdef OUTPUT_READABLE_EULER
            // display Euler angles in degrees
            mpu.dmpGetQuaternion(&q, fifoBuffer);
            mpu.dmpGetEuler(euler, &q);
            Serial.print("euler\t");
            Serial.print(euler[0] * 180/M_PI);
            Serial.print("\t");
            Serial.print(euler[1] * 180/M_PI);
            Serial.print("\t");
            Serial.println(euler[2] * 180/M_PI);
        #endif
        #ifdef OUTPUT_READABLE_YAWPITCHROLL
            // display Euler angles in degrees
            mpu.dmpGetQuaternion(&q, fifoBuffer);
            mpu.dmpGetGravity(&gravity, &q);
            mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);
            Serial.print("ypr\t");
            Serial.print(ypr[0] * 180/M_PI);
            Serial.print("\t");
            Serial.print(ypr[1] * 180/M_PI);
            Serial.print("\t");
            Serial.println(ypr[2] * 180/M_PI);
        #endif
        #ifdef OUTPUT_READABLE_REALACCEL
            // display real acceleration, adjusted to remove gravity
            mpu.dmpGetQuaternion(&q, fifoBuffer);
            mpu.dmpGetAccel(&aa, fifoBuffer);
            mpu.dmpGetGravity(&gravity, &q);
            mpu.dmpGetLinearAccel(&aaReal, &aa, &gravity);
            Serial.print("areal\t");
            Serial.print(aaReal.x);
            Serial.print("\t");
            Serial.print(aaReal.y);
            Serial.print("\t");
            Serial.println(aaReal.z);
        #endif
        #ifdef OUTPUT_READABLE_WORLDACCEL
            // display initial world-frame acceleration, adjusted to remove gravity
            // and rotated based on known orientation from quaternion
            mpu.dmpGetQuaternion(&q, fifoBuffer);
            mpu.dmpGetAccel(&aa, fifoBuffer);
            mpu.dmpGetGravity(&gravity, &q);
            mpu.dmpGetLinearAccel(&aaReal, &aa, &gravity);
```

```
            mpu.dmpGetLinearAccelInWorld(&aaWorld, &aaReal, &q);
            Serial.print("aworld\t");
            Serial.print(aaWorld.x);
            Serial.print("\t");
            Serial.print(aaWorld.y);
            Serial.print("\t");
            Serial.println(aaWorld.z);
        #endif

        #ifdef OUTPUT_TEAPOT
            // display quaternion values in InvenSense Teapot demo format:
            teapotPacket[2] = fifoBuffer[0];
            teapotPacket[3] = fifoBuffer[1];
            teapotPacket[4] = fifoBuffer[4];
            teapotPacket[5] = fifoBuffer[5];
            teapotPacket[6] = fifoBuffer[8];
            teapotPacket[7] = fifoBuffer[9];
            teapotPacket[8] = fifoBuffer[12];
            teapotPacket[9] = fifoBuffer[13];
            Serial.write(teapotPacket, 14);
            teapotPacket[11]++; // packetCount, loops at 0xFF on purpose
        #endif
        // blink LED to indicate activity
        blinkState = !blinkState;
        digitalWrite(LED_PIN, blinkState);
    }
}
```

○ OLED

- 개요



OLED 는 TFT 와 다르게 각 pixel 이 LED 구현된 디스플레이 장치입니다.

아모레드와 같은 스마트폰에서 주로 사용하고, 최근에는 TV, 모니터 등에서도 많이 사용하는 디스플레이 장치입니다.

하지만 이번에 사용할 OLED 디스플레이는 0.96 인치(2.43Cm)로 소형이고, 해상도 역시 128 X 64 이기 때문에 단순 표시 제어용 LCD 라고 생각하시면 됩니다.

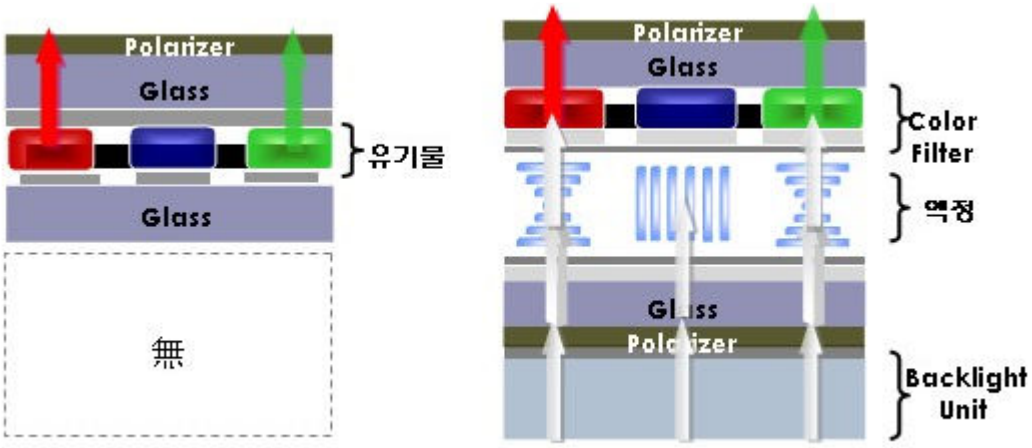
0.96 OLED 도 여러 3rd Part 제조사 들이 있고, 각 제조사별로 인터페이스 및 controller 가 다를 수 있습니다.

고급자 키트에 적용되어있는 OLED 는 SSD1306 controller 가 포함되어있고, Interface 는 I2C 입니다.

소프트웨어 라이브러리는 Adafruit 를 사용할 예정입니다.

추가로 다음 그림은 TFT LCD 와 OLED 의 개념을 비교한 그림입니다.

가장 큰 차이는 TFT LCD 는 Backlight 가 있고, OLED 는 자체 발광을 한다는 설명입니다.



OLED TFT

모듈 주요사양은 다음과 같습니다.

- Display: OLED Display
- Dot matrix size: 128x64
- Display size: 0.96"
- Controller IC: SSD1306
- Working Voltage: DC 3.3V - 5V
- Power consumption: 0.04W
- Commuication way: I2C
- Working Temperature: -30°C ~ +80°C
- Dimension: 29.3 x 27.6mm

- 준비물

. 아두이노 우노(호환보드) X 1

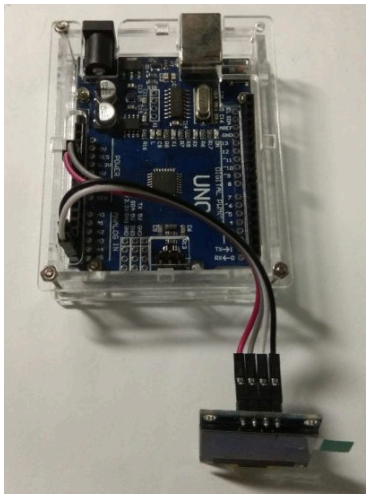
. 0.96 I2C OLED X 1

. 점퍼와이어

- 연결도

구분	*OLED 보드*	*UNO R3 보드*
GND	1 번	GND
VCC	2 번	5V

SCK	3 번	A5
SDA	4 번	A4



- **코딩**

두 개의 SW 라이브러리 설치가 필요합니다.

Adafruit_SSD1306 / Adafruit-GFX-Library

Adafruit-GFX-Library 는 아두이노에서 그래픽 처리가 가능 하도록 하는 라이브러리고, Adafruit_SSD1306 는 OLED 내부 컨트롤러를 제어하기 위한 라이브러리입니다.

라이브러리가 설치되었다면 아래 경로의 예제를 실행합니다.

(₩Documents₩Arduino₩libraries₩Adafruit_SSD1306₩examples₩ssd1306_128x64_i2c)

빌드 및 업로드를 진행한다면 다음과 같은 오류 메시지가 발생할 수 있습니다.

```
ssd1306_128x64_i2c:54: error: #error ("Height incorrect, please fix Adafruit_SSD1306.h!");
 #error("Height incorrect, please fix Adafruit_SSD1306.h!");
exit status 1
#error ("Height incorrect, please fix Adafruit_SSD1306.h!");
```

해상도가 맞지 않다는 의미입니다.

Adafruit_SSD1306.h 파일에서 아래와 같이 SSD1306_128_32 를 주석처리 하시고 SSD1306_128_64 를 주석을 풀어줍니다.

```
   #define SSD1306_128_64
//   #define SSD1306_128_32
```

이후에도 메모리 경고가 발생할 수 있지만, 무시하고 진행합니다.

(아두이노 우노 내부 메모리가 2Kbyte 이기 때문에 발생하는 오류입니다.)

업로드를 완료 하였으나 화면에 아무것도 표시가 안된다면, I2C ADDRESS 문제일 수 있습니다. Setup 에 display.begin 에서 address 를 0x3D 에서 0x3C 로 변경을 하시면 됩니다.

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3D);

display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
```

예제 소스코드 내용은 Setup 에서 display class 를 등록하고 2 초간격으로 다양한 TEST 함수를 call 해서 LCD 에 그림을 그리는 것입니다.

Display class 에는 그래픽을 위한 다양한 멤버들이 포함되어있습니다.

각 멤버의 기능은 Adafruit_GFX 라이브러리에서 확인할 수 있습니다.

```
 display.setTextSize();

 display.setTextColor(WHITE);

 display.setCursor(0,0);

 display.write(i);

display.drawCircle(display.width()/2, display.height()/2, i, WHITE);
```

```
/*********************************************************************
This is an example for our Monochrome OLEDs based on SSD1306 drivers
  Pick one up today in the adafruit shop!
  ------> http://www.adafruit.com/category/63_98
This example is for a 128x64 size display using I2C to communicate
3 pins are required to interface (2 I2C and one reset)
Adafruit invests time and resources providing this open source code,
please support Adafruit and open-source hardware by purchasing
products from Adafruit!
Written by Limor Fried/Ladyada  for Adafruit Industries.
BSD license, check license.txt for more information
All text above, and the splash screen must be included in any redistribution
*********************************************************************/
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);
#define NUMFLAKES 10
#define XPOS 0
#define YPOS 1
#define DELTAY 2
#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH  16
static const unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };
#if (SSD1306_LCDHEIGHT != 64)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
#endif
void setup()   {
  Serial.begin(9600);
  // by default, we'll generate the high voltage from the 3.3v line internally!
(neat!)
  display.begin(SSD1306_SWITCHCAPVCC, 0x3D);  // initialize with the I2C addr 0x3D
(for the 128x64)
  // init done

  // Show image buffer on the display hardware.
  // Since the buffer is intialized with an Adafruit splashscreen
  // internally, this will display the splashscreen.
  display.display();
  delay(2000);
  // Clear the buffer.
  display.clearDisplay();
  // draw a single pixel
  display.drawPixel(10, 10, WHITE);
  // Show the display buffer on the hardware.
```

```
// NOTE: You _must_ call display after making any drawing commands
// to make them visible on the display hardware!
display.display();
delay(2000);
display.clearDisplay();
// draw many lines
testdrawline();
display.display();
delay(2000);
display.clearDisplay();
// draw rectangles
testdrawrect();
display.display();
delay(2000);
display.clearDisplay();
// draw multiple rectangles
testfillrect();
display.display();
delay(2000);
display.clearDisplay();
// draw mulitple circles
testdrawcircle();
display.display();
delay(2000);
display.clearDisplay();
// draw a white circle, 10 pixel radius
display.fillCircle(display.width()/2, display.height()/2, 10, WHITE);
display.display();
delay(2000);
display.clearDisplay();
testdrawroundrect();
delay(2000);
display.clearDisplay();
testfillroundrect();
delay(2000);
display.clearDisplay();
testdrawtriangle();
delay(2000);
display.clearDisplay();

testfilltriangle();
delay(2000);
display.clearDisplay();
// draw the first ~12 characters in the font
testdrawchar();
display.display();
delay(2000);
display.clearDisplay();
// draw scrolling text
testscrolltext();
delay(2000);
display.clearDisplay();
// text display tests
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0,0);
display.println("Hello, world!");
display.setTextColor(BLACK, WHITE); // 'inverted' text
display.println(3.141592);
display.setTextSize(2);
display.setTextColor(WHITE);
display.print("0x"); display.println(0xDEADBEEF, HEX);
display.display();
```

```
  delay(2000);
  display.clearDisplay();
  // miniature bitmap display
  display.drawBitmap(30, 16,  logo16_glcd_bmp, 16, 16, 1);
  display.display();
  delay(1);
  // invert the display
  display.invertDisplay(true);
  delay(1000);
  display.invertDisplay(false);
  delay(1000);
  display.clearDisplay();
  // draw a bitmap icon and 'animate' movement
  testdrawbitmap(logo16_glcd_bmp, LOGO16_GLCD_HEIGHT, LOGO16_GLCD_WIDTH);
}
void loop() {

}
void testdrawbitmap(const uint8_t *bitmap, uint8_t w, uint8_t h) {
  uint8_t icons[NUMFLAKES][3];

  // initialize
  for (uint8_t f=0; f< NUMFLAKES; f++) {
    icons[f][XPOS] = random(display.width());
    icons[f][YPOS] = 0;
    icons[f][DELTAY] = random(5) + 1;

    Serial.print("x: ");
    Serial.print(icons[f][XPOS], DEC);
    Serial.print(" y: ");
    Serial.print(icons[f][YPOS], DEC);
    Serial.print(" dy: ");
    Serial.println(icons[f][DELTAY], DEC);
  }
  while (1) {
    // draw each icon
    for (uint8_t f=0; f< NUMFLAKES; f++) {
      display.drawBitmap(icons[f][XPOS], icons[f][YPOS], bitmap, w, h, WHITE);
    }
    display.display();
    delay(200);

    // then erase it + move it
    for (uint8_t f=0; f< NUMFLAKES; f++) {
      display.drawBitmap(icons[f][XPOS], icons[f][YPOS], bitmap, w, h, BLACK);
      // move it
      icons[f][YPOS] += icons[f][DELTAY];
      // if its gone, reinit
      if (icons[f][YPOS] > display.height()) {
        icons[f][XPOS] = random(display.width());
        icons[f][YPOS] = 0;
        icons[f][DELTAY] = random(5) + 1;
      }
    }
   }
}
void testdrawchar(void) {
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,0);
  for (uint8_t i=0; i < 168; i++) {
    if (i == '\n') continue;
    display.write(i);
```

```
    if ((i > 0) && (i % 21 == 0))
      display.println();
  }
  display.display();
  delay(1);
}
void testdrawcircle(void) {
  for (int16_t i=0; i<display.height(); i+=2) {
    display.drawCircle(display.width()/2, display.height()/2, i, WHITE);
    display.display();
    delay(1);
  }
}
void testfillrect(void) {
  uint8_t color = 1;
  for (int16_t i=0; i<display.height()/2; i+=3) {
    // alternate colors
    display.fillRect(i, i, display.width()-i*2, display.height()-i*2, color%2);
    display.display();
    delay(1);
    color++;
  }
}
void testdrawtriangle(void) {
  for (int16_t i=0; i<min(display.width(),display.height())/2; i+=5) {
    display.drawTriangle(display.width()/2, display.height()/2-i,
                  display.width()/2-i, display.height()/2+i,
                  display.width()/2+i, display.height()/2+i, WHITE);
    display.display();
    delay(1);
  }
}
void testfilltriangle(void) {
  uint8_t color = WHITE;
  for (int16_t i=min(display.width(),display.height())/2; i>0; i-=5) {
    display.fillTriangle(display.width()/2, display.height()/2-i,
                  display.width()/2-i, display.height()/2+i,
                  display.width()/2+i, display.height()/2+i, WHITE);
    if (color == WHITE) color = BLACK;
    else color = WHITE;
    display.display();
    delay(1);
  }
}
void testdrawroundrect(void) {
  for (int16_t i=0; i<display.height()/2-2; i+=2) {
    display.drawRoundRect(i, i, display.width()-2*i, display.height()-2*i,
display.height()/4, WHITE);
    display.display();
    delay(1);
  }
}
void testfillroundrect(void) {
  uint8_t color = WHITE;
  for (int16_t i=0; i<display.height()/2-2; i+=2) {
    display.fillRoundRect(i, i, display.width()-2*i, display.height()-2*i,
display.height()/4, color);
    if (color == WHITE) color = BLACK;
    else color = WHITE;
    display.display();
    delay(1);
  }
}
```

```
void testdrawrect(void) {
  for (int16_t i=0; i<display.height()/2; i+=2) {
    display.drawRect(i, i, display.width()-2*i, display.height()-2*i, WHITE);
    display.display();
    delay(1);
  }
}
void testdrawline() {
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(0, 0, i, display.height()-1, WHITE);
    display.display();
    delay(1);
  }
  for (int16_t i=0; i<display.height(); i+=4) {
    display.drawLine(0, 0, display.width()-1, i, WHITE);
    display.display();
    delay(1);
  }
  delay(250);

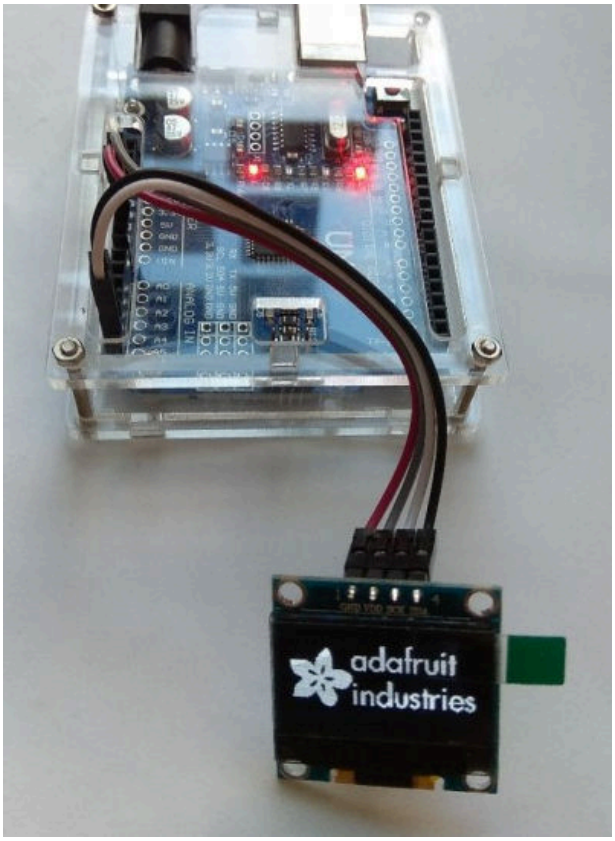
  display.clearDisplay();
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(0, display.height()-1, i, 0, WHITE);
    display.display();
    delay(1);
  }
  for (int16_t i=display.height()-1; i>=0; i-=4) {
    display.drawLine(0, display.height()-1, display.width()-1, i, WHITE);
    display.display();
    delay(1);
  }
  delay(250);

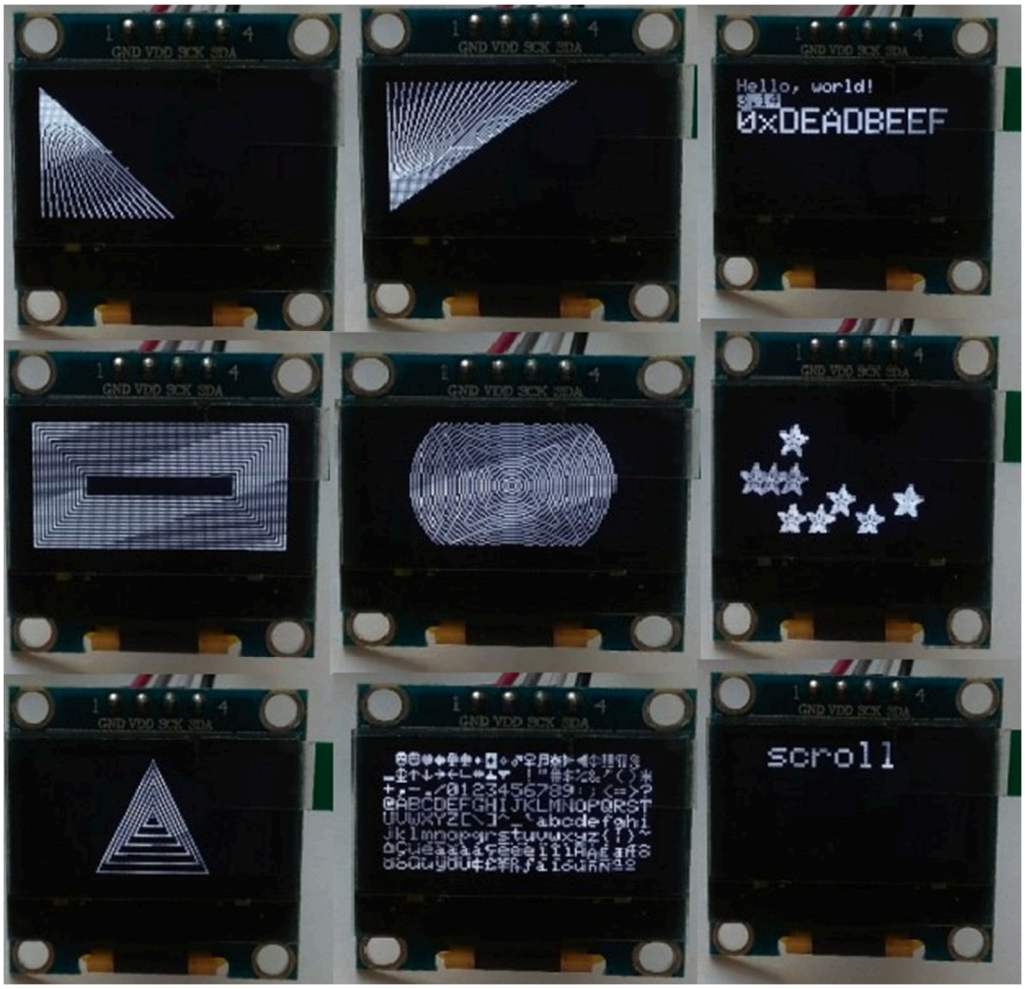
  display.clearDisplay();
  for (int16_t i=display.width()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, i, 0, WHITE);
    display.display();
    delay(1);
  }
  for (int16_t i=display.height()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, 0, i, WHITE);
    display.display();
    delay(1);
  }
  delay(250);
  display.clearDisplay();
  for (int16_t i=0; i<display.height(); i+=4) {
    display.drawLine(display.width()-1, 0, 0, i, WHITE);
    display.display();
    delay(1);
  }
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(display.width()-1, 0, i, display.height()-1, WHITE);
    display.display();
    delay(1);
  }
  delay(250);
}
void testscrolltext(void) {
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(10,0);
```

```
  display.clearDisplay();
  display.println("scroll");
  display.display();
  delay(1);

  display.startscrollright(0x00, 0x0F);
  delay(2000);
  display.stopscroll();
  delay(1000);
  display.startscrollleft(0x00, 0x0F);
  delay(2000);
  display.stopscroll();
  delay(1000);
  display.startscrolldiagright(0x00, 0x07);
  delay(2000);
  display.startscrolldiagleft(0x00, 0x07);
  delay(2000);
  display.stopscroll();
}
```

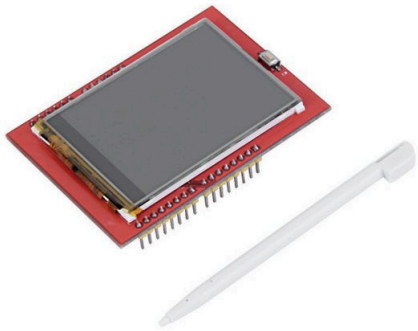
- **결과**




GND VDD SCK SDA
Hello, world!
0xDEADBEEF
scroll

○ 2.4인치 TFT

- **개요**



2.4 인치 TFT LCD 는 아두이노 우노 또는 아두이노 메가와 결합해서 사용이 가능한 TFT LCD 입니다. 터치가 내장되어있으며 18bit 컬러 과 320x240 해상도 표현이 가능합니다. 내부 LCD 컨트롤러는 SPFD5408 이 적용되어있으며 8 비트 데이터 인터페이스를 사용합니다.

주요 사양은 아래와 같습니다.

- 2.4" diagonal LCD TFT display
- Bright, 4 white-LED backlight
- Colorful, 18-bit 262,000 different shades
- 4-wire resistive touchscreen
- 240×320 resolution
- spfd5408 controller with built in video RAM buffer or S6D0154 controller or S6D04H0X
- 8 bit digital interface, plus 4 control lines
- 5V compatible, use with 3.3V or 5V logic
- Onboard 3.3V @ 300mA LDO regulator
- Size: 71*52*7mm

- Weight: about 31g

- 준비물

. 아두이노 우노(호환보드) X 1

. 2.4 인치 TFT LCD X 1

- 연결도

핀 배치가 아두이노 우노 R3와 호환되기 때문에 LCD를 아두이노 우노에 수직으로 연결만 하면 됩니다.



- 코딩

Adafruit-GFX-Library-master 와 TFTLCD-Library-master 라이브러리가 설치가 필요합니다.

다음 예제코드는 Setup()에서 LCD 로 팔렛트를 그래픽으로 표시하고, Loop() 에서 터치를 인식해서 Paint 작업을 하는 예제입니다.

```
//*************************************************************************
// Project : 2_4_TFT_LCD control in Advance set
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************/
#include <Adafruit_GFX.h>    // Core graphics library
#include <Adafruit_TFTLCD.h> // Hardware-specific library
#include <TouchScreen.h>

#if defined(__SAM3X8E__)
    #undef __FlashStringHelper::F(string_literal)
    #define F(string_literal) string_literal
#endif

//I need to change the pins below for it to work. NOt sure why
```

```
#define YP A3  // must be an analog pin, use "An" notation!
#define XM A2  // must be an analog pin, use "An" notation!
#define YM 9   // can be a digital pin
#define XP 8   // can be a digital pin

#define TS_MINX 150
#define TS_MINY 120
#define TS_MAXX 920
#define TS_MAXY 940

// For better pressure precision, we need to know the resistance
// between X+ and X- Use any multimeter to read it
// For the one we're using, its 300 ohms across the X plate
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300);

#define LCD_CS A3
#define LCD_CD A2
#define LCD_WR A1
#define LCD_RD A0
// optional
#define LCD_RESET A4

// Assign human-readable names to some common 16-bit color values:
#define  BLACK   0x0000
#define BLUE    0x001F
#define RED     0xF800
#define GREEN   0x07E0
#define CYAN    0x07FF
#define MAGENTA 0xF81F
#define YELLOW  0xFFE0
#define WHITE   0xFFFF


Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);

#define BOXSIZE 40
#define PENRADIUS 3
int oldcolor, currentcolor;

void setup(void) {
  Serial.begin(9600);
  Serial.println(F("Paint!"));

  tft.reset();

  //Once again, need to hard code the code below
  uint16_t identifier = 0x9341;

  if(identifier == 0x9325) {
    Serial.println(F("Found ILI9325 LCD driver"));
  } else if(identifier == 0x9328) {
    Serial.println(F("Found ILI9328 LCD driver"));
  } else if(identifier == 0x7575) {
    Serial.println(F("Found HX8347G LCD driver"));
  } else if(identifier == 0x9341) {
    Serial.println(F("Found ILI9341 LCD driver"));
  } else if(identifier == 0x8357) {
    Serial.println(F("Found HX8357D LCD driver"));
  } else {
    Serial.print(F("Unknown LCD driver chip: "));
    Serial.println(identifier, HEX);
    Serial.println(F("If using the Adafruit 2.8\" TFT Arduino shield, the
line:"));
```

```
    Serial.println(F("  #define USE_ADAFRUIT_SHIELD_PINOUT"));
    Serial.println(F("should appear in the library header (Adafruit_TFT.h)."));
    Serial.println(F("If using the breakout board, it should NOT be #defined!"));
    Serial.println(F("Also if using the breakout, double-check that all wiring"));
    Serial.println(F("matches the tutorial."));
    return;
  }

  tft.begin(identifier);

  tft.fillScreen(BLACK);

  tft.fillRect(0, 0, BOXSIZE, BOXSIZE, RED);
  tft.fillRect(BOXSIZE, 0, BOXSIZE, BOXSIZE, YELLOW);
  tft.fillRect(BOXSIZE*2, 0, BOXSIZE, BOXSIZE, GREEN);
  tft.fillRect(BOXSIZE*3, 0, BOXSIZE, BOXSIZE, CYAN);
  tft.fillRect(BOXSIZE*4, 0, BOXSIZE, BOXSIZE, BLUE);
  tft.fillRect(BOXSIZE*5, 0, BOXSIZE, BOXSIZE, MAGENTA);
  // tft.fillRect(BOXSIZE*6, 0, BOXSIZE, BOXSIZE, WHITE);

  tft.drawRect(0, 0, BOXSIZE, BOXSIZE, WHITE);
  currentcolor = RED;

  pinMode(13, OUTPUT);
}

#define MINPRESSURE 10
#define MAXPRESSURE 1000

void loop()
{
  digitalWrite(13, HIGH);
  TSPoint p = ts.getPoint();
  digitalWrite(13, LOW);

  // if sharing pins, you'll need to fix the directions of the touchscreen pins
  //pinMode(XP, OUTPUT);
  pinMode(XM, OUTPUT);
  pinMode(YP, OUTPUT);
  //pinMode(YM, OUTPUT);

  // we have some minimum pressure we consider 'valid'
  // pressure of 0 means no pressing!

  if (p.z > MINPRESSURE && p.z < MAXPRESSURE) {
    /*
    Serial.print("X = "); Serial.print(p.x);
    Serial.print("\tY = "); Serial.print(p.y);
    Serial.print("\tPressure = "); Serial.println(p.z);
    */

    if (p.y < (TS_MINY-5)) {
      Serial.println("erase");
      // press the bottom of the screen to erase
      tft.fillRect(0, BOXSIZE, tft.width(), tft.height()-BOXSIZE, BLACK);
    }
    // scale from 0->1023 to tft.width

    p.x=1023-p.x;
    p.y=1023-p.y;
    //Code below to fix the bug of inverted coordinates
     p.x = map(p.x, TS_MINX, TS_MAXX, 0, tft.width() );
    p.y = map(p.y, TS_MINY, TS_MAXY, 0, tft.height() );
```

```
    //Correct offset of touch. Manual calibration
    p.x+=5;
    p.y+=18;
    if (p.y < BOXSIZE) {
       oldcolor = currentcolor;

       if (p.x < BOXSIZE) {
         currentcolor = RED;
         tft.drawRect(0, 0, BOXSIZE, BOXSIZE, WHITE);
       } else if (p.x < BOXSIZE*2) {
         currentcolor = YELLOW;
         tft.drawRect(BOXSIZE, 0, BOXSIZE, BOXSIZE, WHITE);
       } else if (p.x < BOXSIZE*3) {
         currentcolor = GREEN;
         tft.drawRect(BOXSIZE*2, 0, BOXSIZE, BOXSIZE, WHITE);
       } else if (p.x < BOXSIZE*4) {
         currentcolor = CYAN;
         tft.drawRect(BOXSIZE*3, 0, BOXSIZE, BOXSIZE, WHITE);
       } else if (p.x < BOXSIZE*5) {
         currentcolor = BLUE;
         tft.drawRect(BOXSIZE*4, 0, BOXSIZE, BOXSIZE, WHITE);
       } else if (p.x < BOXSIZE*6) {
         currentcolor = MAGENTA;
         tft.drawRect(BOXSIZE*5, 0, BOXSIZE, BOXSIZE, WHITE);
       }

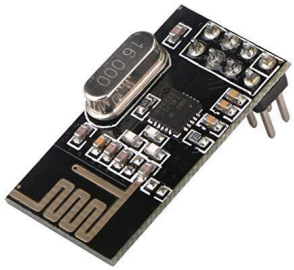
       if (oldcolor != currentcolor) {
          if (oldcolor == RED) tft.fillRect(0, 0, BOXSIZE, BOXSIZE, RED);
          if (oldcolor == YELLOW) tft.fillRect(BOXSIZE, 0, BOXSIZE, BOXSIZE,
YELLOW);
          if (oldcolor == GREEN) tft.fillRect(BOXSIZE*2, 0, BOXSIZE, BOXSIZE,
GREEN);
          if (oldcolor == CYAN) tft.fillRect(BOXSIZE*3, 0, BOXSIZE, BOXSIZE, CYAN);
          if (oldcolor == BLUE) tft.fillRect(BOXSIZE*4, 0, BOXSIZE, BOXSIZE, BLUE);
          if (oldcolor == MAGENTA) tft.fillRect(BOXSIZE*5, 0, BOXSIZE, BOXSIZE,
MAGENTA);
       }
    }
    if (((p.y-PENRADIUS) > BOXSIZE) && ((p.y+PENRADIUS) < tft.height())) {
      tft.fillCircle(p.x, p.y, PENRADIUS, currentcolor);
    }
  }
}
```

- 결과



○ 2.4Ghz 무선송수신

- 개요



2.4Ghz 주파수는 우리 주변에서 흔히 사용하는 주파수입니다. (WI-FI, BT, 무선마우스 등) 2.4Gh 주파수의 장점은 높은 주파수로 인해서 수신감도와 전송속도가 높고, 원거리 양방향 통신이 가능합니다.

NRF24L01 모듈은 2.4Ghz 를 사용하는 양방향 통신모듈입니다. 가격도 저렴하고 아두이노와 쉽게 연결하여 사용이 가능합니다. 그래서 가정용 IOT 를 직접 만들 때 단순 무선 통신용으로 많이 사용합니다.

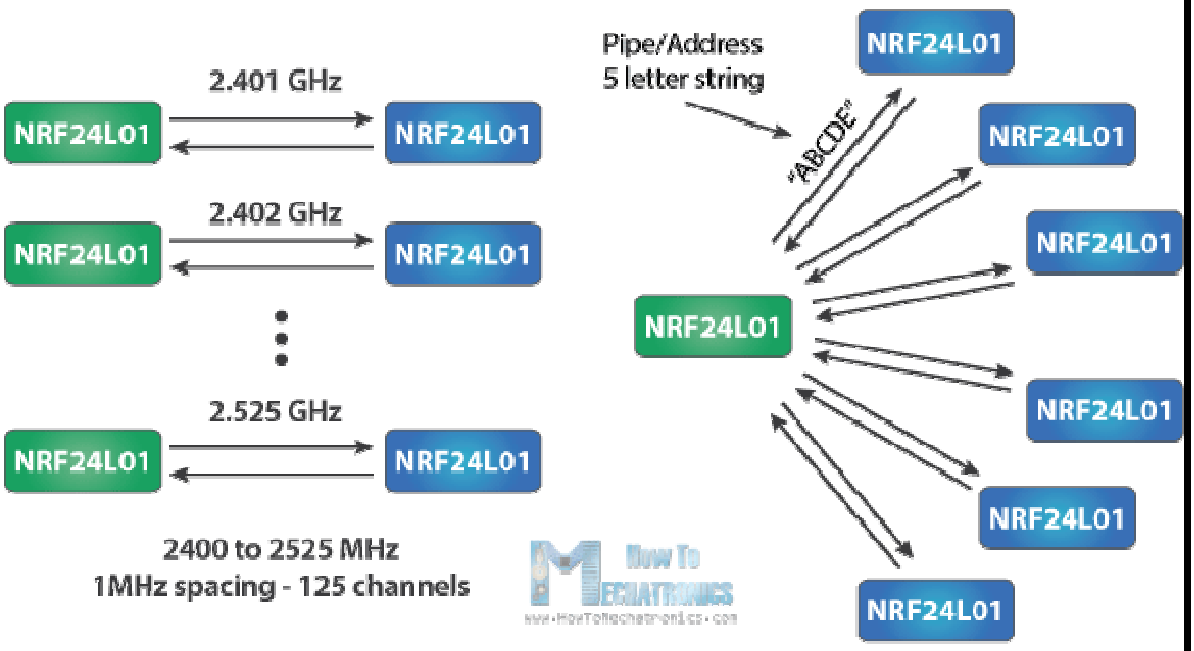
주요 사양은 아래와 같습니다.

- 2.4GHz 무선통신 Nordic 모듈
- SPI 통신을 사용하며 전압은 1.9V~3.6V
- 최대 2Mbps 의 속도
- 자동 재 전송 부가 기능
- 최대 송신 거리 200m(안테나 달린 모듈은 1000m)

주의 사항은

- 노이즈 감소를 하기 위해서 VCC, GND 바이패스 캐패시티(10uF)가 필요
- 가품이 많음
- 전원단에 5V 인가시 모듈이 고장

NRF24L01 은 다음 그림과 같이 1:1 및 1:N 통신을 지원합니다.



- **준비물**

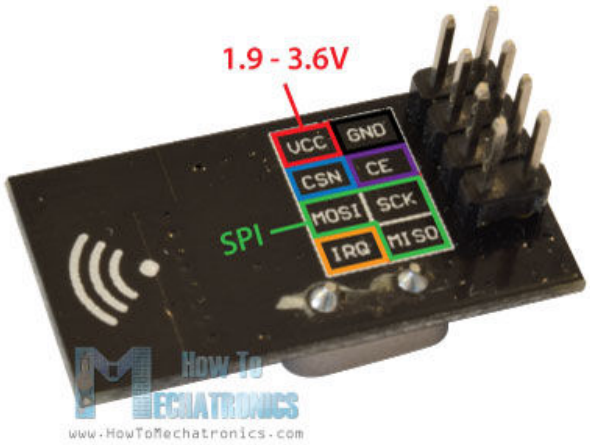
. 아두이노 우노(호환보드) X 2

. NRF24L01 X 2

. 점퍼와이어

- **연결도**

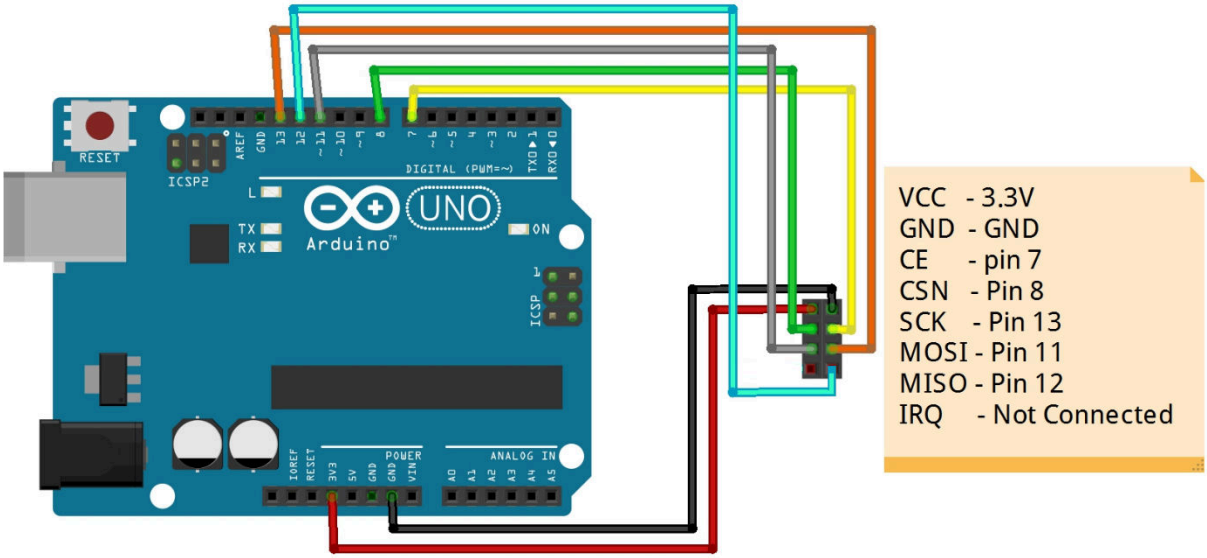
핀맵은 아래와 같습니다.



(출처: https://howtomechatronics.com)

송수신 통신을 확인하기 위해서는 아두이노 우노 R3 보드 2 대가 필요합니다.

아래 연결도와 같이 아두이노 보드 2 대를 구성합니다.



- **코딩**

nRF24L01 라이브러리 설치가 필요합니다.

아래코드는 전송하는 아두이노 보드 코드입니다.

```
//*********************************************************************************
// Project : nRF24L01 TX in Advanced kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(7, 8); // CE, CSN

const byte address[6] = "00001";

void setup() {
  radio.begin();
  radio.openWritingPipe(address);
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();
}
void loop() {
  const char text[] = "Hello Kit plus";
  radio.write(&text, sizeof(text));
  delay(1000);
}
```

아래코드는 수신 받는 아두이노의 코드입니다.

```
//*********************************************************************************
// Project : nRF24L01 RX in Advanced kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(7, 8); // CE, CSN
const byte address[6] = "00001";
void setup() {
  Serial.begin(9600);
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();
}
void loop() {
  if (radio.available()) {
    char text[32] = "";
    radio.read(&text, sizeof(text));
    Serial.println(text);
  }
}
```

코드를 간단히 설명 드리면

RF24 radio(7,8); SPI 기본인 CE와 RF24 라이브러리에서 필요한 CSN을 설정

radio.openReadingPipe(0, address);

radio.openWritingPipe(address); 전송/수신하고자 하는 nRF24L01 ADDR 설정

radio.setPALevel(RF24_PA_MIN); 파워증폭 레벨 설정

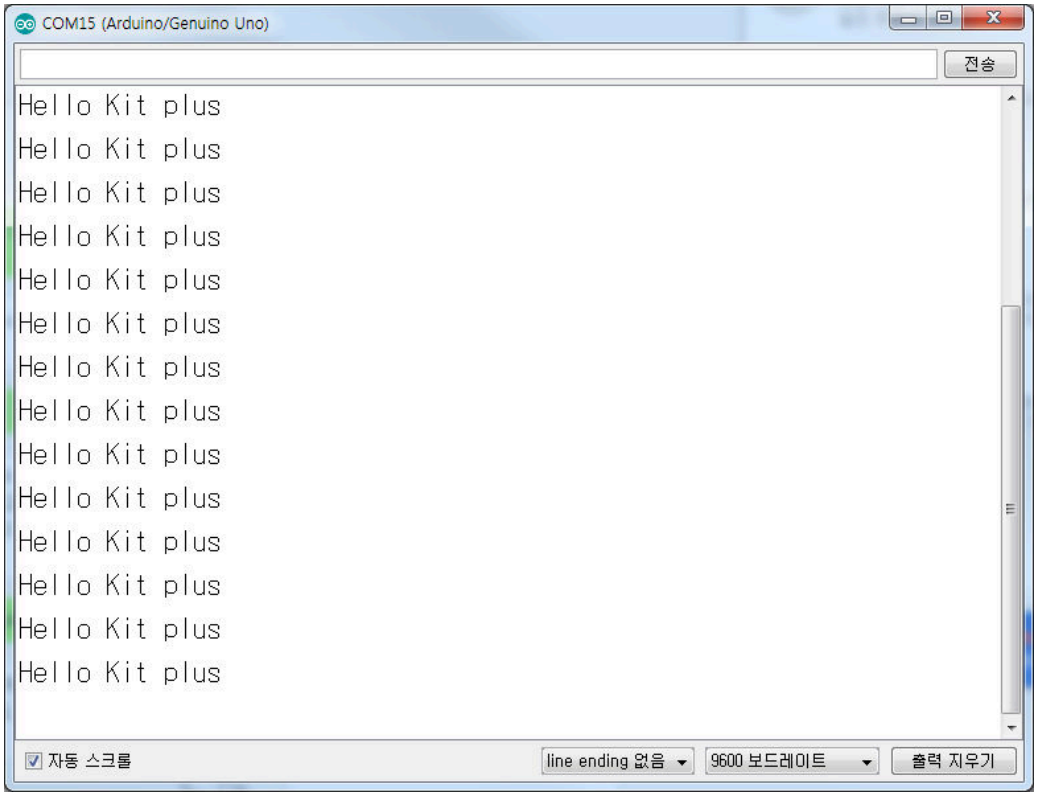
radio.startListening(); 수신 상태를 설정

radio.available()) 수신 데이터를 체크

radio.read(&text, sizeof(text)); 데이터를 버퍼로 수신

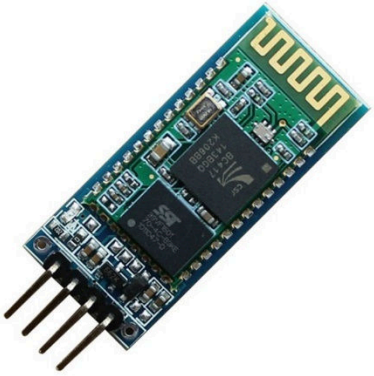
- **결과**

1 초단위로 데이터를 수신 받습니다.


COM15 (Arduino/Genuino Uno)
전송
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
Hello Kit plus
자동 스크롤
line ending 없음
9600 보드레이트
출력 지우기

○ 블루투스 슬레이브 모듈

- 개요



블루투스는 2.4GHz 무선 링크를 통해 데이터 통신을 하는 통신 표준 규격입니다. 특징으로는 단거리, 저전력, 저비용으로 장치들을 무선 연결할 수 있습니다.

IOT 키트에서 제공되는 블루투스 모듈은 슬레이브 모듈이기 때문에 마스터 블루투스 장치가 추가로 필요합니다. 본 매뉴얼에서는 마스터 블루투스 장치를 우리가 흔히 사용하는 스마트폰으로 활용하도록 하겠습니다.

블루투스 모듈은 기본적으로 2.4G 대역의 주파수를 사용합니다. 각각의 개별 채널을 통해 주파수 간섭이 일어 나는 것을 방지할 수 있지만, WIFI 와 함께 사용한다면 같은 대역의 주파수가 충돌하는 문제가 발생되어 재밍이 발생 할 수도 있습니다.

모듈에 포함되어있는 LED 는 페이링 연결 이전에는 준비단계로서 빨간색 LED 가 빠르게 깜빡 거리고, 페어링이 완료된다면 LED 가 5 초마다 깜빡입니다.

기본적인 사양은 아래와 같습니다.

- EDR 블루투스 2.0, 2Mbps – 3Mbps 변조
- 2.4GHz 안테나 내장
- 외부 8Mbit FLASH
- 3.3V 저전압 동작
- 옵션 PIO 제어
- 표준 HCI 포트 (UART)
- SMD 배치 프로세스로 모듈
- RoHS 규제 절차
- 디지털 2.4GHz 무선 송신
- CSR BC04 블루투스 칩 기술
- 크기 (27mm × 13mm × 2mm)
- 블루투스 클래스 2 전력 레벨
- 보관 온도 : -40 +85 도, 작동 온도 : -25 으로 75 도

- 준비물

. 아두이노 우노(호환보드) X 1

. 블루투스 슬레이브 모듈 X 1

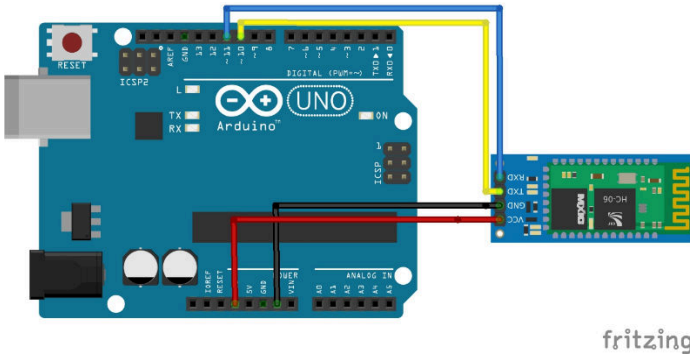
. 안드로이드 스마트폰 X 1

. 점퍼와이어

- 연결도

핀맵에 대한 설명입니다.

Pin	Description
VCC	Supply input 3.3V to 5.5V Input
GND	Ground
TXD	Transmit Data (Connect RX line from other peripheral here)
RXD	Receive Data (Connect TX line from other peripheral here)



(모듈내부에 저항이 포함되어있습니다. 추가 저항은 필요 없습니다.)

- 코딩

별도의 추가 라이브러리설치 없이 소프트웨어 시리얼을 사용합니다.

다음 코드는 1 초에 한번씩 스마트폰으로 데이터를 전송하고, 스마트폰으로 부터 수신 받은 데이터를 시리얼 모니터로 문자를 출력합니다. 만약 데이터가 '0' 이 입력되면 LED 를 켜고, '1'이 입력되면 LED 를 끄는 동작을 보여줍니다.

```
//*********************************************************************************
// Project : bluetooth slave in Advanced set
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
// Define Pins
#include <SoftwareSerial.h> // import the serial library
SoftwareSerial mySerial(10, 11); // RX, TX
int ledpin=13; // led on D13 will show blink on / off
char BluetoothData; // the data given from Computer
void waitForResponse() {
    delay(1000);
    while (mySerial.available()) {
      Serial.write(mySerial.read());
    }
    Serial.write("\n");
}
void setup()
{
  Serial.begin(9600);
  // put your setup code here, to run once:
  mySerial.begin(9600);
  Serial.println("Check Bluetooth LED blinking");
  pinMode(ledpin,OUTPUT);
}
void loop()
{ // put your main code here, to run repeatedly:
  if (mySerial.available()){
    BluetoothData=mySerial.read();
    Serial.print("Recevied data : ");
    Serial.println(BluetoothData);
```

```
    if(BluetoothData=='1') { // if number 1 pressed ....
      digitalWrite(ledpin,1);
    }else if (BluetoothData=='0'){ // if number 0 pressed ....
        digitalWrite(ledpin,0);
    }
  }
  delay(1000);// prepare for next data ...
    mySerial.println("Send to Phone");
}
```

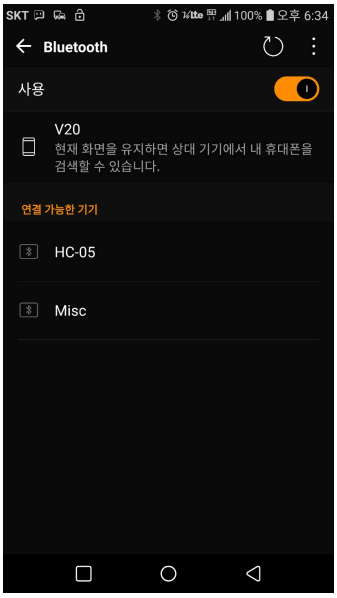
- **결과**

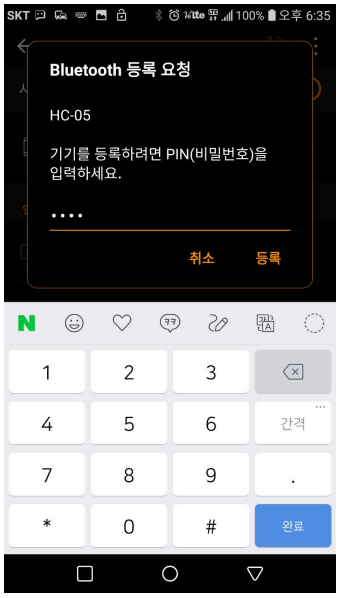
다시 말씀드리지만, IOT 키트에서 제공되는 블루투스는 slave 모듈입니다. 그러므로 마스터에서(스마트폰) 페어링을 과정을 거쳐야 사용을 할 수 있습니다.

안드로이드폰을 기준으로 설명을 드리겠습니다.

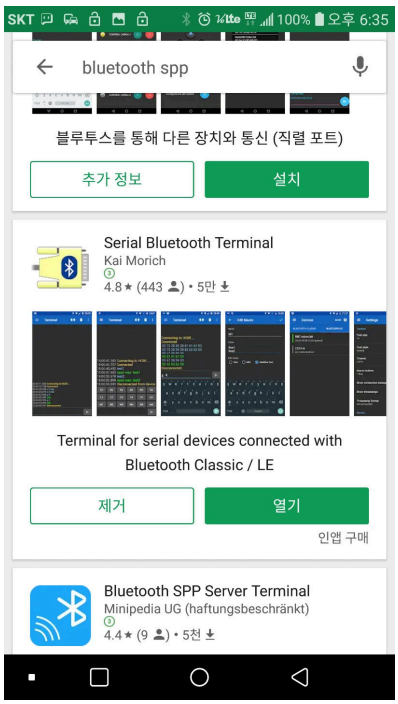
안드로이드폰 설정으로 이동하셔서 블루투스 페어링을 시도합니다.

아래그림에서 HC-05 또는 HC-06 를 누르고, PIN code 을 "1234"로 입력합니다.



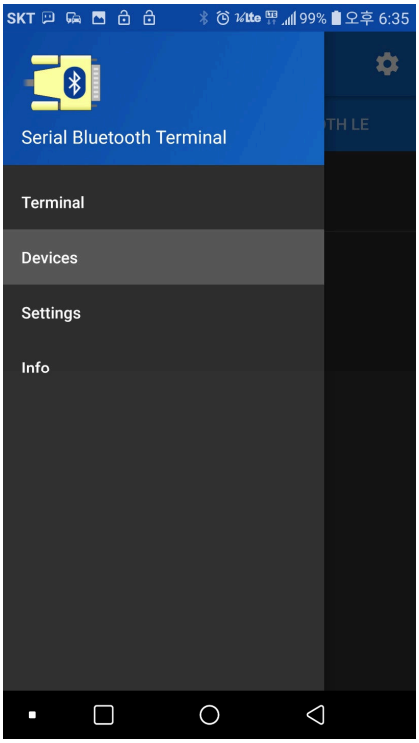


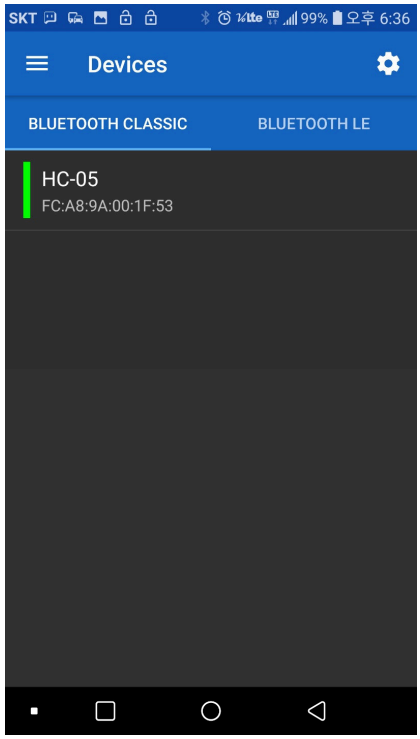
안드로이드 Play 스토어에서 Bluetooth spp 를 검색하여, Serial Bluetooth Terminal 프로그램을 다운받고 실행합니다.



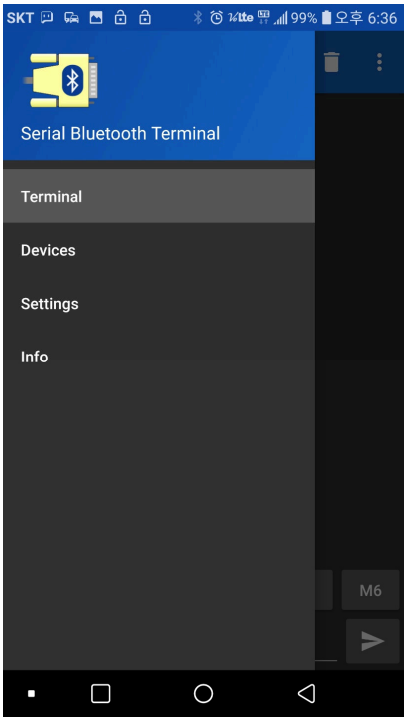
프로그램 왼쪽상단에 ☰메뉴를 눌러서 "Devices"를 선택합니다.

그리고 다음 그림과 같이 HC-05(HC-06)를 선택하면 초록색 상태로 변경됩니다.

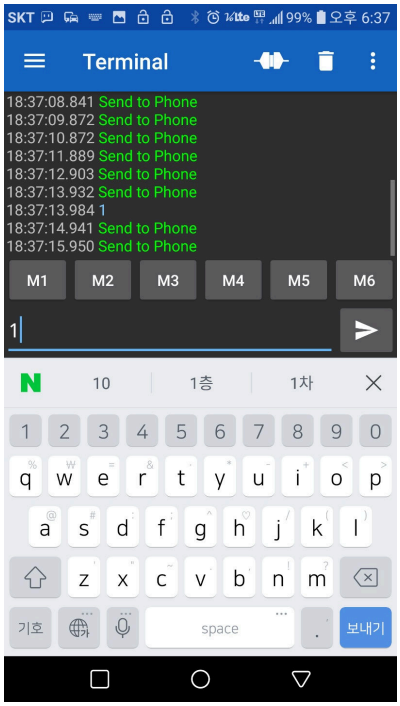




다시 왼쪽 상단에 메뉴를 눌러 Terminal 로 이동합니다.



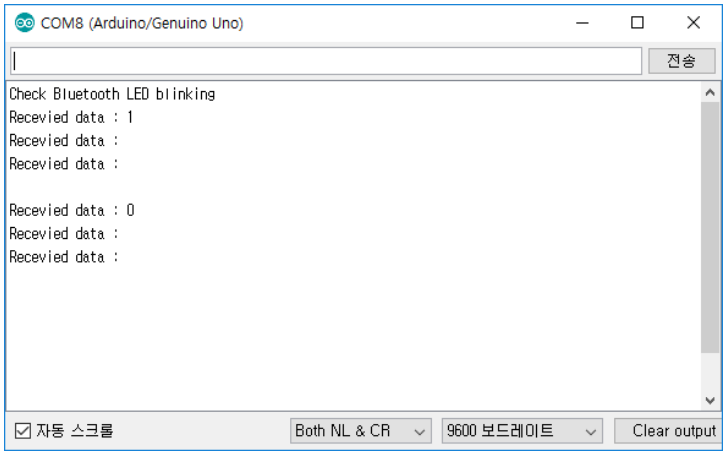
터미널 화면에서 접속 버튼을 누르게 되면, 블루투스가 접속되고, 아두이노에서 보내는 Send to Phone 메시지가 1 초 마다 표시가 됩니다.



아래 입력란에 숫자 '0'을 입력하고 전송버튼을 누릅니다.

다음에 숫자 '1'을 입력하고 전송버튼을 누릅니다.

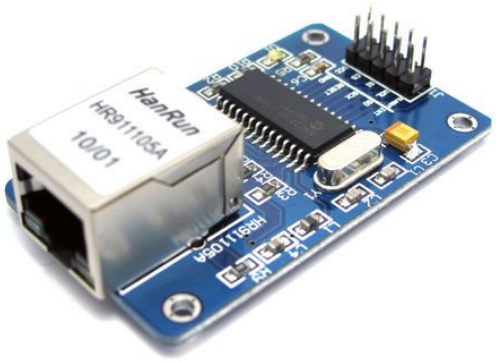
아두이노 시리얼 모니터에 블루투스로 부터 전송 받은 데이터와 함께 아두이노 보드의 LED 상태가 변화 하는 것을 확인할 수 있습니다.



* 1 과 0 이 표시된 이후 두 줄이 더 표시되는 이유는 안드로이드 블루투스 프로그램이 캐리지 리턴을 같이 보내기 때문입니다.

○ 이더넷 모듈

- 개요



ENC28J60 은 10BASE-T 기반의 이더넷 컨트롤러 보드입니다. 아두이노를 PC 컴퓨터로 생각한다면 ENC28J60 은 네트워크 카드입니다.

ENC28J60 IC 는 이더넷의 MAC 및 PHY, 버퍼 RAM 를 포함하고 있으며, RJ45 규격 포트를 사용합니다. 외부와 통신 방식을 SPI 직렬을 사용합니다. 활용범위는 VoIP, IOT 통신, 웹서버 등 다양한 어플리케이션이 포함될 수 있습니다.

일반적으로 웹에서 찾아 찾아보실 수 있는 ENC28J60 예제는 이더넷을 통한 아두이노 LED 제어하기, IOT 전원제어, 트위터 글 올리기, 소형 웹서버 구축하기, Google Ping 등이 있습니다.

본 매뉴얼에서는 DHCP 로 연결하여 NTP 서버에서 현재 시간 가져오기를 코딩해보도록 하겠습니다.

기본적인 사양은 다음과 같습니다.

- ENC28J60 이더넷 칩, SOP28 패키지, 최대 10MB / s 의 속도
- SPI 인터페이스
- 2×6 커넥터 와 쉽게 MCU 와 연결해서 사용
- RJ45 커넥터 HR911102A 내장
- 전원 상태 LED
- 전원 공급 : 3.3V 레귤레이터가 내장 되어있어 3.3V 와 5V(VCC) 핀으로 선택해서 공급 가능

– 25Mhz 크리스탈
– 보드크기 : 4.76 x 1.88 (Cm)

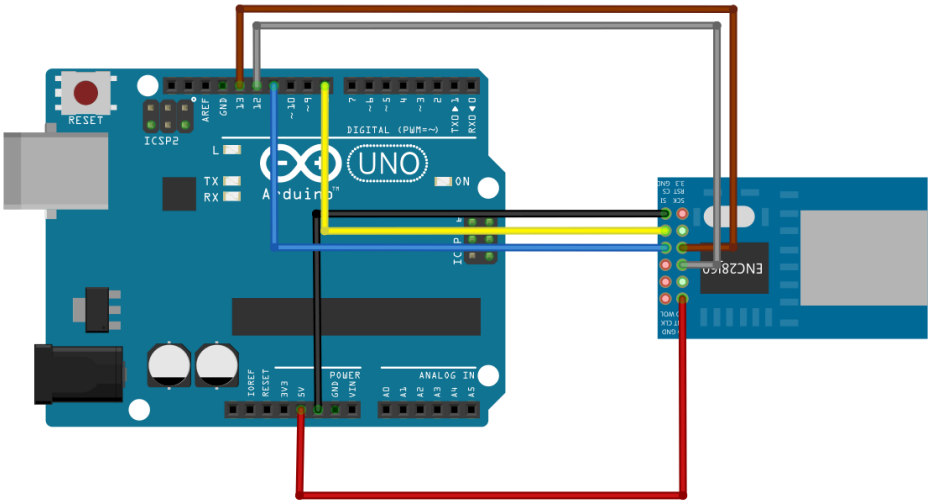
- **준비물**

. 아두이노 우노(호환보드) X 1

. ENC28J60 X 1

. 점퍼와이어

- **연결도**



- **코딩**

ENC28J60 EtherCard 라이브러리 설치가 필요합니다.

Ethernet Class 멤버들을 설명 드리면,

DHCP = ether.begin(sizeof Ethernet::buffer, mymac,SS);

➔ 이더넷 장치를 시작, MAC address 와 SPI 의 ChipSelect 를 설정

ether.dhcpSetup()

➔ DHCP 설정을 시작

ether.ntpRequest(ntpServer, ntpMyPort);

➔ NTP 시간 서버로 Request 를 전송

ether.ntpProcessAnswer(&timeFromNTP,ntpMyPort))

➔ 시간정보를 수신(시간정보는 Long 형으로 1900 년 1 월 1 일 0 시 0 분 0 초 부터의 경과된 초 단위입니다.)

localtime(&now);

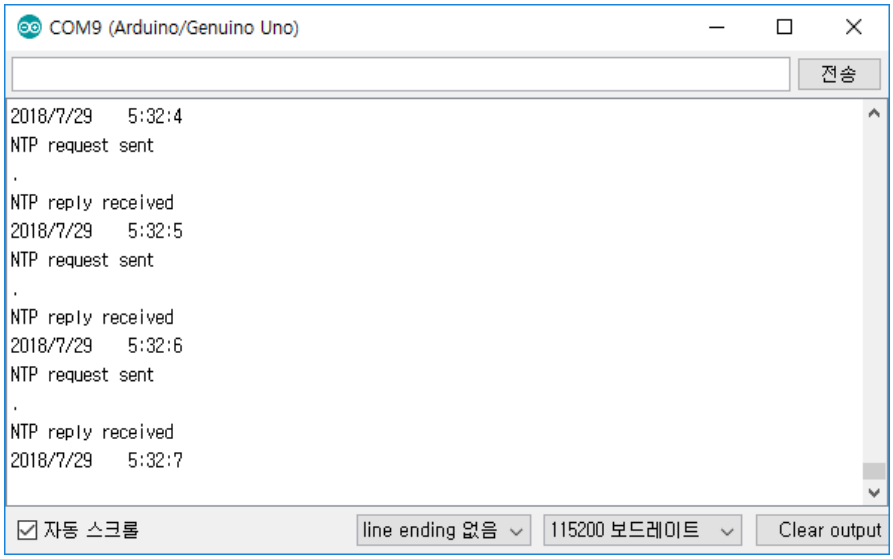
➔ 초단위 시간을 년월일시분초로 변환

```
//**************************************************************************
// Project : ENC28J60 in Advenced set(Get time)
// Board : Arduino Uno
// By : Kit Plus
//**************************************************************************
#include <EtherCard.h>
#include "time.h"
static byte ntpServer[] = {194,109,6,2}; //ntp.xs4all.nl but you can cange in any
public ntp server.
uint8_t ntpMyPort = 123;
boolean networkConnection = true;
#define SS 8
static byte mymac[] = { 0x74,0x69,0x69,0x2D,0x30,0x31 };
byte Ethernet::buffer[700];
void setup () {
  Serial.begin(115200);
  Serial.println(F("\n[testDHCP]"));
  Serial.print("MAC: ");
  for (byte i = 0; i < 6; ++i) {
    Serial.print(mymac[i], HEX);
    if (i < 5)
      Serial.print(':');
  }
  Serial.println();
  int i = 1;
  int DHCP = 0;
   //Try to get dhcp settings 3 times before giving up
  Serial.println("Resolve dhcp ...");
  while( DHCP == 0 && i < 3){
    Serial.print("Trying to resolve DHCP attempt nr. ");
    Serial.println(i);
    delay(1000);
    DHCP = ether.begin(sizeof Ethernet::buffer, mymac,SS);
    i++;
    }
  Serial.println(F("Setting up DHCP"));
  if (!ether.dhcpSetup()) {
    Serial.println(F("DHCP failed"));
```

```
    networkConnection = false;
    }

  ether.printIp("My IP: ", ether.myip);
  ether.printIp("Netmask: ", ether.netmask);
  ether.printIp("GW IP: ", ether.gwip);
  ether.printIp("DNS IP: ", ether.dnsip);
  Serial.println("End setup");
} //End of setup
unsigned long getNtpTime() {
  unsigned long timeFromNTP;
  int i = 60; //Number of attempts
  Serial.println("NTP request sent");
  while(i > 0) {
  ether.ntpRequest(ntpServer, ntpMyPort);
      Serial.print("."); //Each dot is a NTP request
      word length = ether.packetReceive();
      ether.packetLoop(length);
      if(length > 0 && ether.ntpProcessAnswer(&timeFromNTP,ntpMyPort)) {
        Serial.println();
        Serial.println("NTP reply received");
        return timeFromNTP;
      }
      delay(500);
      i--;
    }
  Serial.println();
  Serial.println("NTP reply failed");
  return 0;
}
void loop () {
  // Get the current time
  time_t now = getNtpTime();
  struct tm * timeinfo;
  timeinfo = localtime(&now);
 //Output Day and Time
  Serial.print(timeinfo->tm_year+1800);
  Serial.print("/");
  Serial.print(timeinfo->tm_mon+1);
  Serial.print("/");
  Serial.print(timeinfo->tm_mday);
  Serial.print("    ");
  Serial.print(timeinfo->tm_hour+1);
  Serial.print(":");
  Serial.print(timeinfo->tm_min);
  Serial.print(":");
  Serial.println(timeinfo->tm_sec);
  delay(1000);
}
```

- **결과**



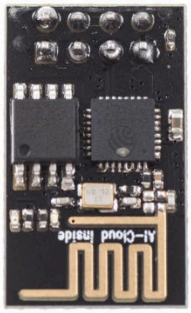
* 시간이 맞지 않는 이유는 TimeZone 때문이고, 코드에서 년과 월에 상수를 더한 이유는 시간 구조체 때문입니다. 아래의 구조체 설명을 참고하십시오.

```
struct tm
{
  int tm_sec;                 /* Seconds.     [0-60] (1 leap second) */
  int tm_min;                 /* Minutes.     [0-59] */
  int tm_hour;                /* Hours.       [0-23] */
  int tm_mday;                /* Day.         [1-31] */
  int tm_mon;                 /* Month.       [0-11] */
  int tm_year;                /* Year - 1900.  */
  int tm_wday;                /* Day of week. [0-6] */
  int tm_yday;                /* Days in year.[0-365] */
  int tm_isdst;               /* DST.         [-1/0/1]*/

};
```

○ 무선 WIFI 모듈

- 개요



WI-FI 는 우리 생활 전반에 사용되는 무선통신 규격입니다. 가정에서 사용하는 PC 또는 스마트폰과 같은 개인 전자 기기를 포함해 학교나 회사, 카페 등과 같은 모든 곳에 에서 사용이 되고 있는 통신 수단입니다.

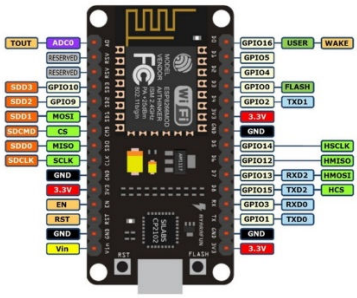
이제부터 알아볼 ESP8266 은 FULL TCP/IP 지원 및 MCU controller 가 포함된 저가의 WIFI chip 으로 상하이 Espressif System 사에서 만들었습니다.

2014 년 8 월 ESP-01 이 출시되었을 때 저렴한 가격($2~5)과 고성능 WIFI 모듈로 세상을 놀라게 했다고 합니다. IC 제조사인 Espressif 는 소형 WIFI IC 만 제공하고 많은 Third part 회사들에 의해서 다양한 모듈이 탄생을 하기 시작했습니다.

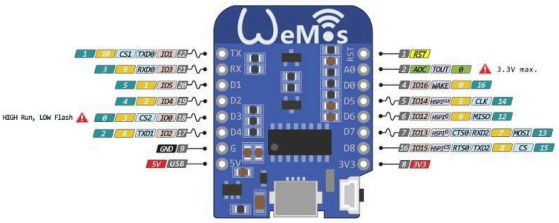
특히, ESP8266 모듈에 들어가는 소프트웨어를 직접 개발할 수 있는 SDK 도 Espressif 사에서 오픈 소스로 제공하고 있습니다. 기본적인 탑재 소프트웨어는 Lua script 가 탑재되어 있고 Putty 환경에서 shell prompt 제공합니다.

가격이 저렴한 이유 때문에 수 많은 메이커분 들이 esp8266 을 사용한 IOT 사물 인터넷, 스마트홈등을의 프로젝트를 만들고 자료를 공유하고 있습니다.

ESP8266 은 다양한 보드가 존재합니다. ESP8266 NodeMCU V3 또는 ESP8266 WEMOS 는 ESP-12E 모듈을 탑재하여 아두이노와 같은 외부 컨트롤러가 없이도 아두이노 코딩 환경과 SPI I2C, UART 등와 같은 외부 인터페이스를 제공합니다.

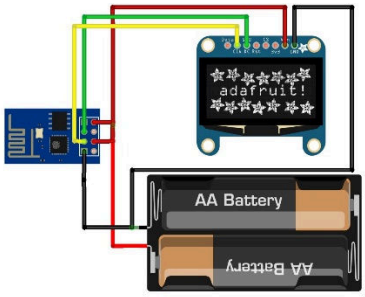


ESP-12E NodeMCU V3



ESP-12E Wemos

하지만 지금부터 사용할 ESP8266 ESP-01 버전은 외부 인터페이스가 부족하기 때문에 아두이노와 같은 외부 컨트롤러가 필요합니다. 만약 외부 인터페이스가 크게 필요 없는 소형 IOT 를 구현하신다면, 아두이노 없이 내부 펌웨어만 직접 구동할 수도 있습니다.



(외부 전원을 이용한 IOT 모니터)

주요 사항은 아래와 같습니다.

Module	Espressif ESP8266
ESP8266 Version	ESP-01
동작 명령	UART AT command
Antenna configuration	On-boad PCB antenna
무선 규격	802.11 b/g/n 지원
Baud rate	기본 9600 bps (변경가능함)

Flash memory	4M Byte
Processor speed	80-160Mhz
크기(Size)	14.5 x 24.8mm
동작 전압	3.3V,(5V 입력 할 경우 모듈이 손상됨)

이전에 설명 드린 것 같이 ESP8266 는 3rd part 보드들의 다양한 기능을 지원하기 위해서 여러 가지 버전의 펌웨어가 내장되어 있습니다. 적용되어있는 내부 펌웨어에 따라 기능과 사용법이 다를 수 있습니다. 그러므로 초기에 AT command 용 펌웨어로 업데이트를 진행 해야 다음의 예제를 사용할 수 있습니다.

본 문서에서는 사용할 예제는 AT command 를 사용한 기상청 날씨 가져오기를 진행합니다. 그러므로 1 단계로 ESP-01 내부 펌웨어 업데이트를 진행하는 방법과, 2 단계로 아두이노에서 기상청 날씨 가져오기를 설명 드리겠습니다.

- 준비물

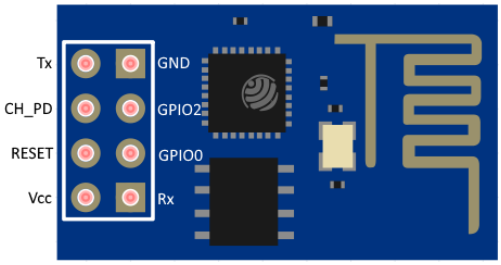
. 아두이노 우노(호환보드) X 1

. 브래드보드 X 1

. ESP-01 X 1

. 점퍼와이어

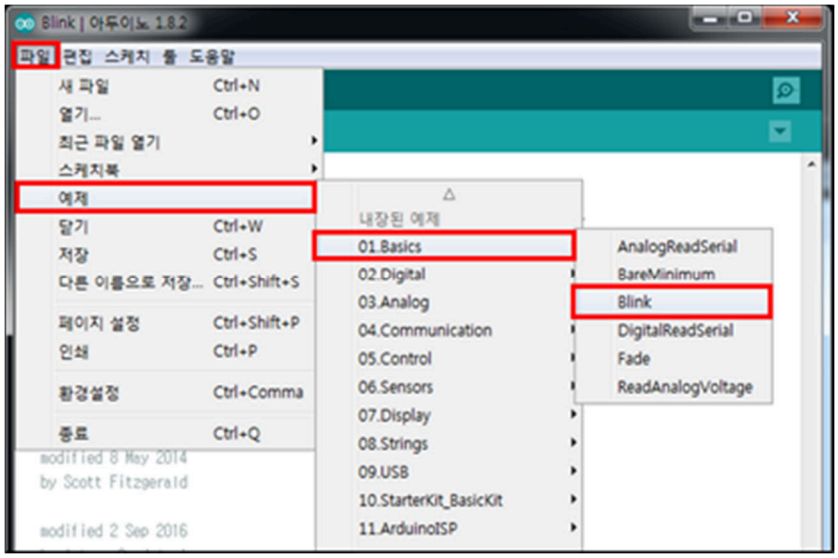
1) 펌웨어 업데이트 절차



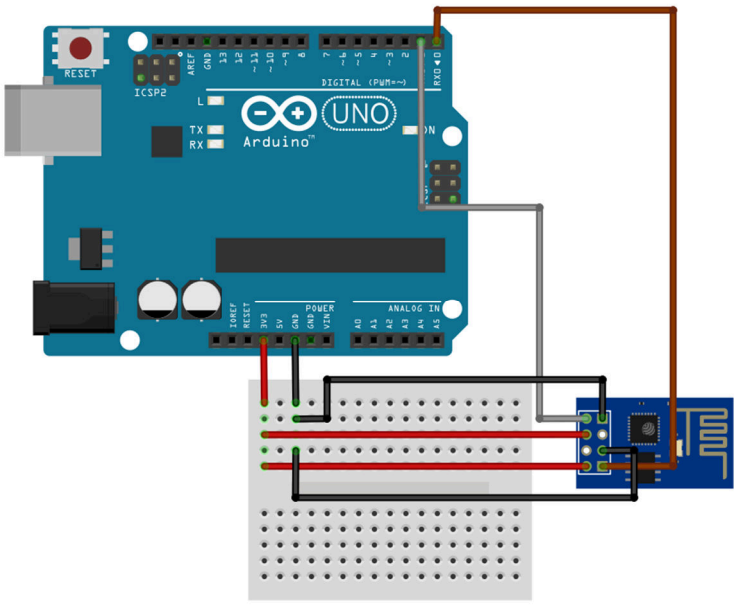
** 중요 : GPIO0 는 Flash mode 를 결정하는 포트로 LOW 라면 Flash mode 상태 입니다.*

IOT 용 펌웨어를 업데이트 해보도록 하겠습니다.

ESP-01 펌웨어 업데이트 통신은 아두이노 보드의 UART 포트를 사용합니다. 그러므로 통신 충돌을 막기 위해서 아두이노 내부 코드를 초기화 해야 합니다. Blink 예제를 선택해서 보드에 업로드 시켜 UART 를 초기화 합니다.



1-2) 펌웨어 업데이트 연결도

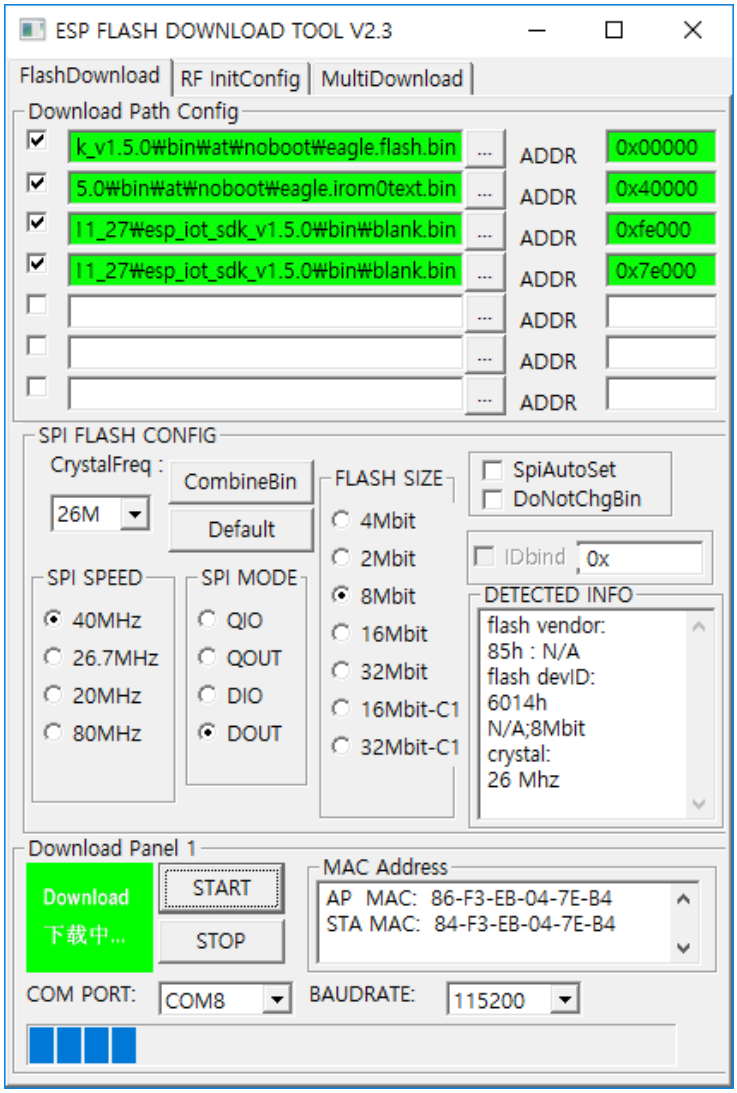


Flash tool 과 펌웨어 파일의 압축을 풉니다.

esp_iot_sdk_v1.5.0_15_11_27.zip

FLASH_DOWNLOAD_TOOLS_v2.4_150924.7z

Flash download 프로그램을 실행시켜 다음 그림과 같이 설정을 하고 "시작"을 버튼을 누릅니다.



* 참고사항

- esp_iot_sdk_v1.5.0_15_11_27 안에 포함되어있는 4 개의 Bin 파일의 위치는 아래를 참고하시고, 디렉토리에 한글로 되어있는 폴더가 포함되어있으면 안됩니다.

  esp_iot_sdk_v1.5.0₩bin₩at₩noboot₩eagle.flash.bin

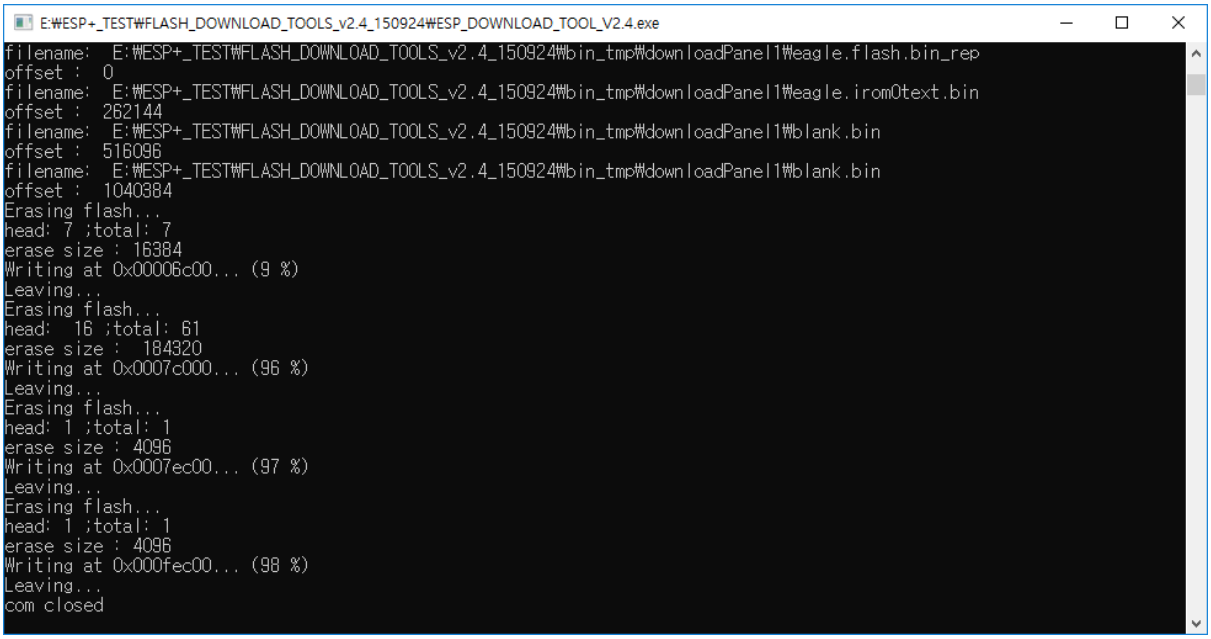
  esp_iot_sdk_v1.5.0₩bin₩at₩noboot₩eagle.irom0text.bin

  esp_iot_sdk_v1.5.0₩bin₩blank.bin

  esp_iot_sdk_v1.5.0₩bin₩blank.bin

- 위 그림과 같이 파일명, 설정을 정확히 하시고, PC 와 아두이노와 연결된 COMx 를 확인하시고 COM PORT 를 설정합니다. 시작을 누르면 Command 창에서 업데이트 진행사항 나옵니다. 최종 Com close 가 나오면 완료된 것입니다.

- 진행사항 완료시 command 창



1-3) 업데이트 정상 확인 방법

아두이노 USB 를 제거하여 전원을 완전 분리합니다.

GND 와 연결된 ESP8266 ESP-01 의 GPIO0 의 연결을 제거합니다. (중요)

다시 USB 연결하고, 아두이노의 시리얼 모니터를 실행합니다.

펌웨어를 설치한 뒤에는 ESP-01 의 보레이트는 115200bps 설정되어있습니다.

앞으로 ESP-01 는 아두이노와 연결해서 사용할 때 SoftwareSerial 사용 합니다. SoftwareSerial 의 최대 속도는 57600bps 이므로 안전성 있게 9600 으로 재설정을 해보도록 하겠습니다

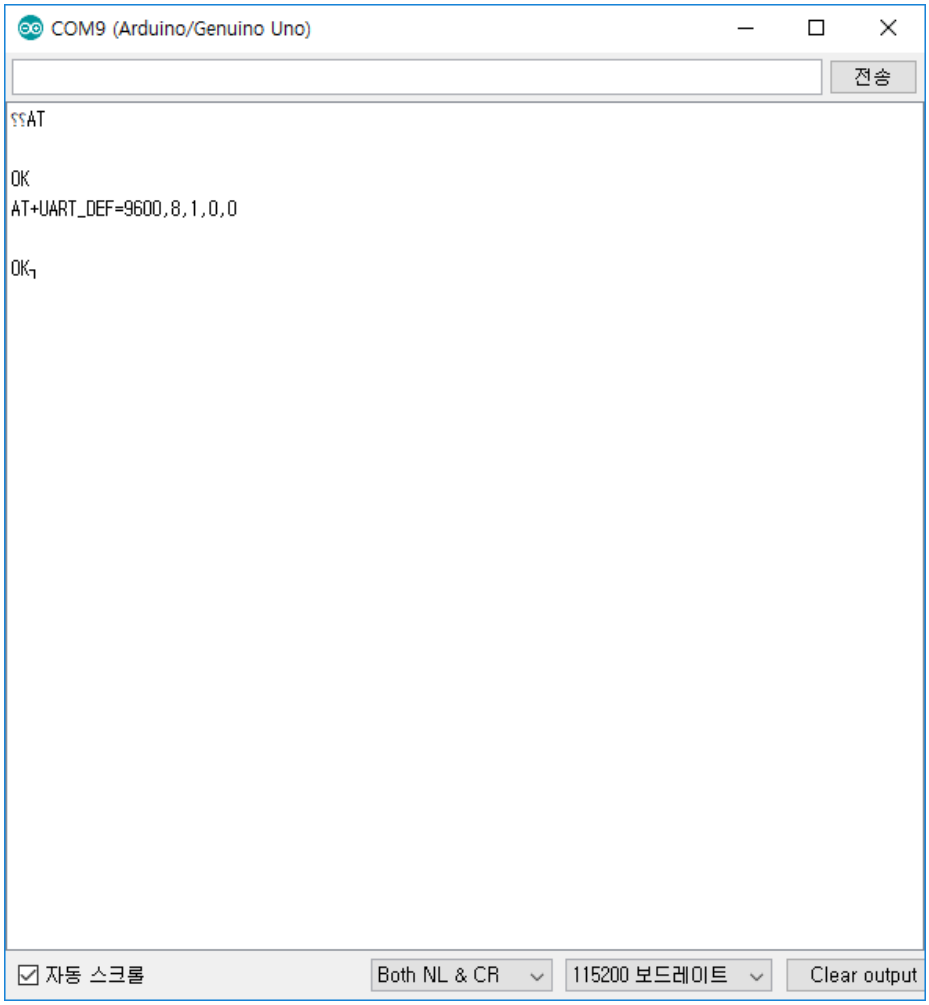
우선 시리얼 모니터의 Both NL & CR 과 115200 보드레이트로 변경하고,

입력창에 "AT" 입력하시고 전송을 누르면, 아래 모니터창에 AT OK 이라는 문구가 표시되면 IOT 펌웨어가 정상적으로 업데이트 된 것입니다.

보드레이트를 속도를 변경하기 위해서 아두이노 시리얼 모니터창에서 아래와 같이 입력하고 전송을 누르면 설정이 완료됩니다.

- BAUD rate 변경 AT command

AT+UART_DEF=9600,8,1,0,0

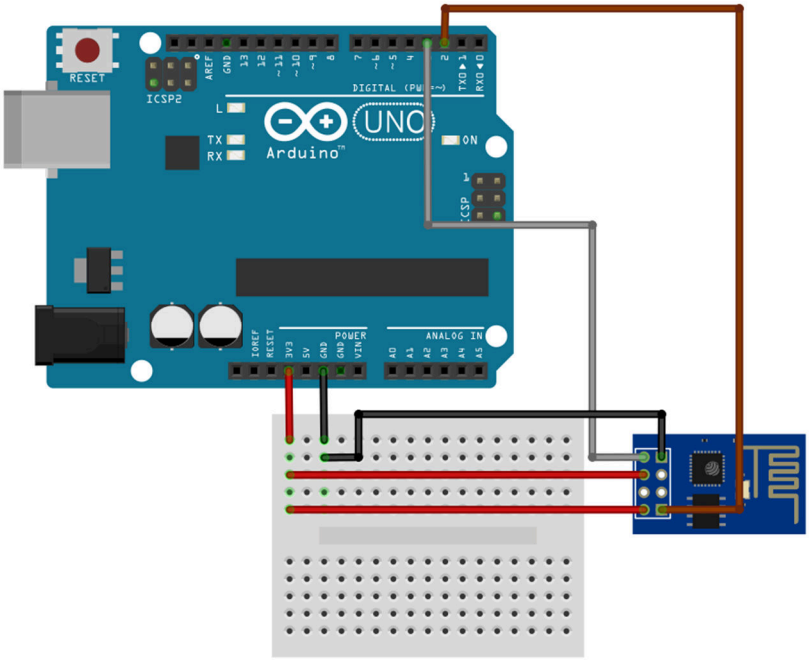


여기까지 IOT 펌웨어 업데이트 과정이 였습니다.

그럼 이제부터 ESP8266 EPS-01 을 아두이와 연결해서 무선 인터넷을 사용해보겠습니다.

2) 아두이노 사용 연결도

Flash mode 를 결정하는 GPIO0 을 제거하고, 소프트웨어 시리얼을 사용하기 위해서 UART 를 2,3 으로 이동하였습니다.



- **코딩**

WIFIESP 라이브러리 설치가 필요합니다.

다음 코드는 기상청 "kma.go.kr"을 접속하여 특정지역의 날씨를 xml 형식으로 데이터를 가져와 String 을 분리하는 방식입니다.

코드에서 지역을 선택하는 부분은 아래 부분입니다.

```
const String KMA_url = "/wid/queryDFSRSS.jsp?zone=1117053000";
```

원하시는 지역을 변경하기 위해서는 위치 정보를 알아야 합니다.

아래 주소로 들어가셔 원하시는 지역을 선택하시고 "RSS"를 누르면 변경된 위치 주소가 나옵니다. 복사하셔서 코드에 적용하시면 됩니다.

http://www.weather.go.kr/weather/lifenindustry/sevice_rss.jsp

Xml 의 상세정보를 알고 싶으시다면 아래 PDF 를 다운로드 받으시면 됩니다.

http://www.kma.go.kr/images/weather/lifenindustry/timeseries_XML.pdf

각 데이터 분리는 XML 이므로 String Search 방식으로 사용합니다.

```
//*****************************************************************************
// Project : ESP8266 ESP-01 in Advenced set(Get Weather)
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
#include "WiFiEsp.h"
#include "SoftwareSerial.h"
SoftwareSerial esp(3, 2); // RX, TX
char ssid[] = "SO070VOIPA7E5";            // Your AP SSID
char pass[] = "8D3861A7E4";               // Your AR Password
int status = WL_IDLE_STATUS;        // Status
unsigned long ulPreTime = 0;
bool flag_stop;
void setup()
{
  Serial.begin(9600);
  esp.begin(9600);
  Serial.println("Start get weather forecast");
  WiFi.init(&esp);
  Serial.println("Completed esp init");
  if (WiFi.status() == WL_NO_SHIELD)
  {
    Serial.println("WiFi shield not present");
    while (true);
  }
  // Check WIFI connection
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print("Attempting to connect to WPA SSID: ");
    Serial.println(ssid);
    status = WiFi.begin(ssid, pass);
  }
  // Display WIFI information
  Serial.println("You're connected to the network");
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());
  // Display WIFI information
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);
  // Display WIFI information
  long rssi = WiFi.RSSI();
  Serial.print("Signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");
  Serial.println();
}
const String KMA_url = "/wid/queryDFSRSS.jsp?zone=1117053000";
const char* wSERVER = "www.kma.go.kr";
#define httpPort 80
int GetWeatherInformWIFI()
{
  WiFiEspClient client;
  int i=0;
  String temp;
  String tmp_str;
  int cnt=0;
  int res=1;
  float t=0;
  Serial.println("start connect webserver");
  client.setTimeout(5000);
  client.flush();
  if (client.connect(wSERVER, httpPort)) {
```

```
    client.print(String("GET ") + KMA_url + " HTTP/1.1\r\n" +
    "Host: " + wSERVER + "\r\n" +
    "Connection: close\r\n\r\n");
  }
   delay(1000);
    while(client.available()){
      yield();
      String line = client.readStringUntil('\n');
    // search forecast time
      i= line.indexOf("</hour>");
      if(i>0){
        tmp_str="<hour>";
        temp = line.substring(line.indexOf(tmp_str)+tmp_str.length(),i);
        Serial.print(temp+":00  " );
      }
    // search forecast temperature
      i= line.indexOf("</temp>");
      if(i>0){
        tmp_str="<temp>";
        temp = line.substring(line.indexOf(tmp_str)+tmp_str.length(),i);
        Serial.print(temp+" ");
      }
    // search forecast weather string
      i= line.indexOf("</wfEn>");
      if(i>0){
        tmp_str="<wfEn>";
        temp = line.substring(line.indexOf(tmp_str)+tmp_str.length(),i);

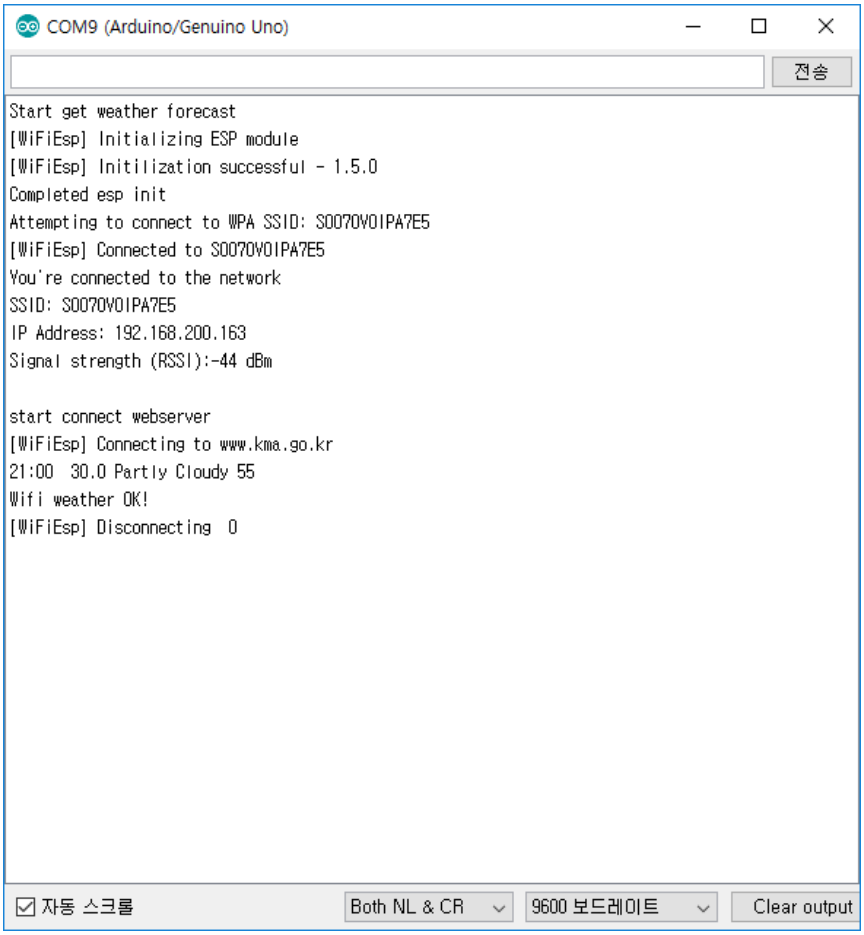
          Serial.print(temp+" ");
      }
    // search Humidity
      i= line.indexOf("</reh>");
      if(i>0){
        tmp_str="<reh>";
        temp = line.substring(line.indexOf(tmp_str)+tmp_str.length(),i);

          Serial.println(temp);
          res=0;
          break;
        }
    }

  if(res == 1){
    Serial.println("Wifi weather Fail!");
  }else{
    Serial.println("Wifi weather OK!");
  }
  client.stop();

  return res;
}
void loop()
{
  unsigned long ulCurTime = millis();
  // period 1 sec
  if (ulCurTime - ulPreTime >= 1000)
  {
    ulPreTime = ulCurTime;
    GetWeatherInformWIFI();
  }
}
```

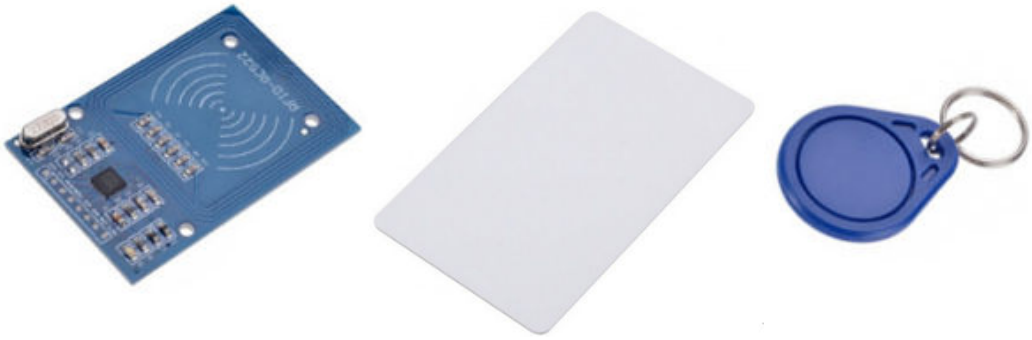
- 결과


COM9 (Arduino/Genuino Uno)
전송
Start get weather forecast
[WiFiEsp] Initializing ESP module
[WiFiEsp] Initilization successful - 1.5.0
Completed esp init
Attempting to connect to WPA SSID: SOO7OVOIPA7E5
[WiFiEsp] Connected to SOO7OVOIPA7E5
You're connected to the network
SSID: SOO7OVOIPA7E5
IP Address: 192.168.200.163
Signal strength (RSSI):-44 dBm
start connect webserver
[WiFiEsp] Connecting to www.kma.go.kr
21:00 30.0 Partly Cloudy 55
Wifi weather OK!
[WiFiEsp] Disconnecting 0
자동 스크롤
Both NL & CR
9600 보드레이트
Clear output

○ RFID 키트

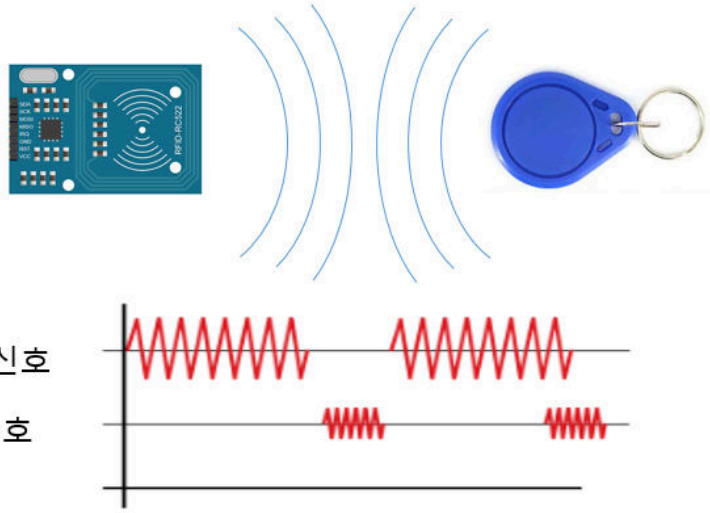
- 개요

RFID(Radio-Frequency Identification) 에 대해서 알아 보겠습니다.



[RFID RC555 판독기 모듈] [Card 전자태그] [Key 고리형 전자태그]

버스카드와 출입카드에 많이 사용되는 RFID 는 주파수를 이용해서 ID 를 식별하는 장치로 전자태그 라고도 불립니다.
판독기(RC555)는 무선주파수를 사용하여 전자태그의 정보를 읽을 수 있습니다.
전자태그는 안테나와 직접회로(비휘발성 메모리)로 구성이 되어있습니다.
판독기에서 무선주파수를 발생시키면 전자태그는 전파에너지를 사용하여 직접회로를 구동하면서 저장되어있는 데이터 신호를 전달합니다.



RC555 판독기 모듈은 안테나가 내장되어있으며 13.56Mhz 로 동작하는 RFID Reader/Writer 입니다. 기본적으로 ISO14443A/MIFARE 모드를 지원합니다.

- 준비물

. 아두이노 우노(호환보드) X 1

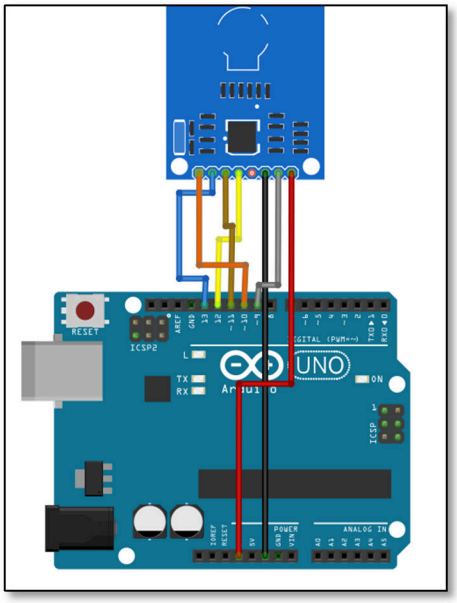
. RFID Reader X 1

. RFID CARD X 1

. RFID Key X1

. 점퍼와이어

- 연결도



- 코딩

RFID 라이브러리 설치가 필요합니다.

```
//*****************************************************************************
// Project : 14_RFID in Learning Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
/*
 * ----------------------------------------------------------------------------
-------------------------------------
 * Example to change UID of changeable MIFARE card.
 * ----------------------------------------------------------------------------
-------------------------------------
```

```
 * This is a MFRC522 library example; for further details and other examples see:
https://github.com/miguelbalboa/rfid
 *
 * This sample shows how to set the UID on a UID changeable MIFARE card.
 * NOTE: for more informations read the README.rst
 *
 * @author Tom Clement
 * @license Released into the public domain.
 *
 * Typical pin layout used:
 * -----------------------------------------------------------------------------
----------
 *             MFRC522      Arduino       Arduino   Arduino    Arduino
Arduino
 *             Reader/PCD   Uno           Mega      Nano v3    Leonardo/Micro   Pro
Micro
 * Signal      Pin          Pin           Pin       Pin        Pin              Pin
 * -----------------------------------------------------------------------------
----------
 * RST/Reset   RST          9             5         D9         RESET/ICSP-5     RST
 * SPI SS      SDA(SS)      10            53        D10        10               10
 * SPI MOSI    MOSI         11 / ICSP-4   51        D11        ICSP-4           16
 * SPI MISO    MISO         12 / ICSP-1   50        D12        ICSP-1           14
 * SPI SCK     SCK          13 / ICSP-3   52        D13        ICSP-3           15
 */
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN   9     // Configurable, see typical pin layout above
#define SS_PIN    10    // Configurable, see typical pin layout above
MFRC522 mfrc522(SS_PIN, RST_PIN);   // Create MFRC522 instance
/* Set your new UID here! */
#define NEW_UID {0xDE, 0xAD, 0xBE, 0xEF}
MFRC522::MIFARE_Key key;
void setup() {
  Serial.begin(9600);  // Initialize serial communications with the PC
  while (!Serial);     // Do nothing if no serial port is opened (added for
Arduinos based on ATMEGA32U4)
  SPI.begin();         // Init SPI bus
  mfrc522.PCD_Init();  // Init MFRC522 card
  Serial.println(F("Warning: this example overwrites the UID of your UID
changeable card, use with care!"));

  // Prepare key - all keys are set to FFFFFFFFFFFFh at chip delivery from the
factory.
  for (byte i = 0; i < 6; i++) {
    key.keyByte[i] = 0xFF;
  }
}
// But of course this is a more proper approach
void loop() {

  // Look for new cards, and select one if present
  if ( ! mfrc522.PICC_IsNewCardPresent() || ! mfrc522.PICC_ReadCardSerial() ) {
    delay(50);
    return;
  }

  // Now a card is selected. The UID and SAK is in mfrc522.uid.

  // Dump UID
  Serial.print(F("Card UID:"));
  for (byte i = 0; i < mfrc522.uid.size; i++) {
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
```

```
    Serial.print(mfrc522.uid.uidByte[i], HEX);
  }
  Serial.println();

  // Set new UID
  byte newUid[] = NEW_UID;
  if ( mfrc522.MIFARE_SetUid(newUid, (byte)4, true) ) {
    Serial.println(F("Wrote new UID to card."));
  }

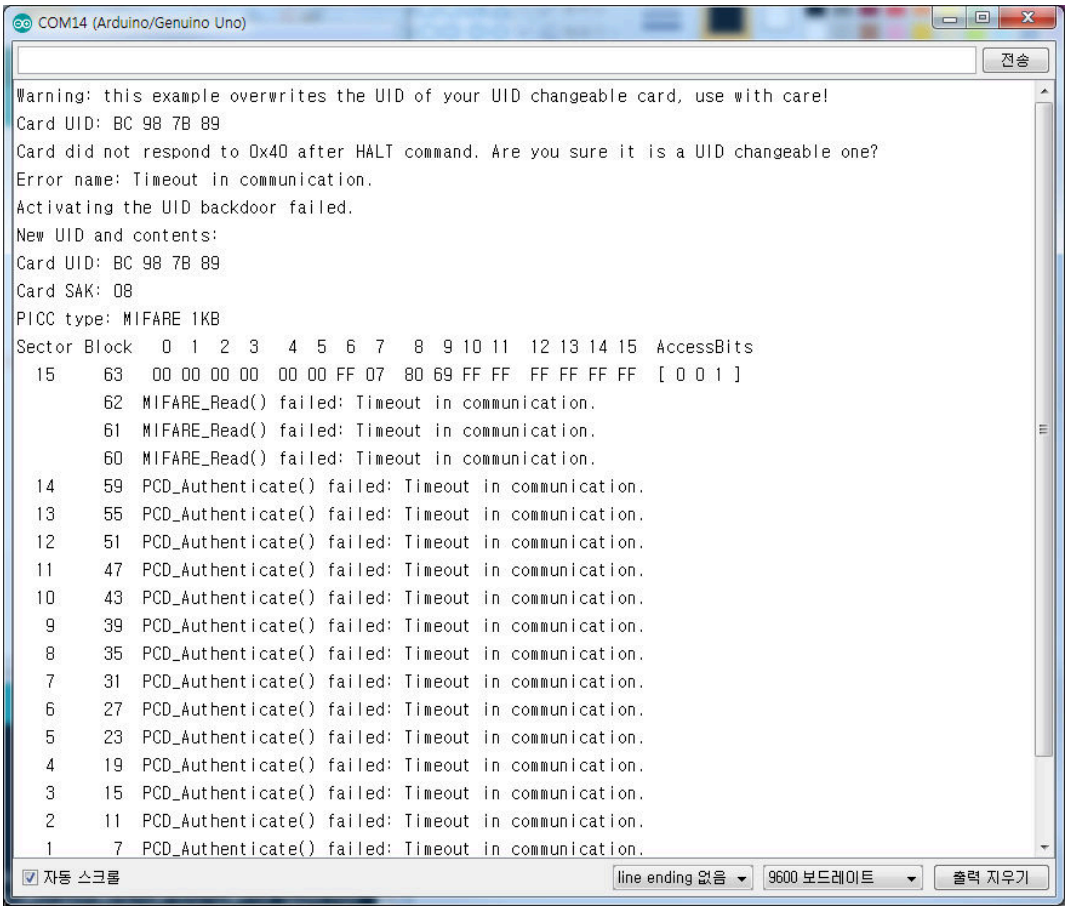
  // Halt PICC and re-select it so DumpToSerial doesn't get confused
  mfrc522.PICC_HaltA();
  if ( ! mfrc522.PICC_IsNewCardPresent() || ! mfrc522.PICC_ReadCardSerial() ) {
    return;
  }

  // Dump the new memory contents
  Serial.println(F("New UID and contents:"));
  mfrc522.PICC_DumpToSerial(&(mfrc522.uid));

  delay(2000);
}
```

- 결과

CARD 또는 Key 의 EERPROM 정보를 보실 수 있습니다.



❑ 참고자료 출처

https://www.arduino.cc/en/Main/Education

https://www.elegoo.com/

http://fritzing.org/download/

https://howtomechatronics.com/

https://en.wikipedia.org

* 본 문서의 저작권은 출처를 밝히고, 비영리 목적으로 배포/활용이 가능합니다.

아두이노 37종 센서 키트 매뉴얼



Last Update : 181205

By : 키트플러스

시작하며

본 매뉴얼은 아두이노 교육에서 자주 사용되는 37 종의 센서에 관한 사용자 매뉴얼입니다.

아두이노 초보자를 위한 별도의 납땜 작업이 필요 없이 최소한의 점퍼를 연결하여 37 종류의 다양한 센서 동작을 확인할 수 있습니다. 본 문서에서 적용되는 아두이노 보드는 UNO R3 를 사용하였습니다.

각 센서의 설명 센서 개요, 연결도, 코드, 확인절차로 구성 되어있습니다.

난이도 조절을 위해 각 센서의 이론과 원리에 대한 설명은 생략 하였으며, 문서에 적용된 소스코드는 별도로 제공을 하고 있습니다.

본 문서가 아두이노 코딩교육을 시작하는 분들에게 많은 도움이 되었으면 합니다.

** 제조사 사정으로 인해 본 문서에서 사용되는 일부 센서모듈들은 유사 센서모듈로 변경될 수 있습니다.*

"나는 나의 의지대로 된다."

-쇼펜하우어

목차 및 제품 구성표

순번	영문이름	한글이름	제공 수량	Page
1	JOYSTICK	조이스틱	1	
2	RELAY	릴레이	1	
3	BIG SOUND	고감도사운드	1	
4	SMALL SOUND	소형사운드	1	
5	TRACKING	IR 트래킹	1	
6	AVOIDANCE	IR 장애물감지	1	
7	FRAME	불꽃감지센서	1	
8	LINEAR HALL	리니어홀센서	1	
9	METAL TOUCH	터치센서	1	
10	Digtal TEMPERTURE	디지털 온도센서	1	
11	NATIVE BUZZER	회로내장부저	1	
12	PASSITIVE BUZZER	무회로부저	1	
13	RGB LED	3 컬러 LED	1	
14	SMD RGB	3 컬러 SMT LED	1	
15	TWO-COLOR	2 컬러 LED	1	
16	MINI TWO LED	2 컬러 소형 LED	1	
17	MAGNETIC SPRING	리드스위치	1	
18	MINI REED SWTICH	미니리드스위치	1	
19	HEARTBEAT SENSOR	손가락맥박센서	1	
20	7 COLOR FLASH	7 컬러 플래쉬	1	
21	LASER EMIT	레이저송신	1	
22	BUTTON	버튼스위치	1	
23	SHOCK	충격센서	1	
24	ROTARY ENCODERS	로터리엔코더	1	
25	2PCS LIGHT CUP	매직컵라이트	2	
26	TILT-SWITCH	각도(틸트)스위치	1	
27	BALL SWITCH	볼스위치	1	
28	PHOTO RESISTOR	광센서(CDS)센서	1	
29	TEMP AND HUMIDITY	온습도센서	1	
30	ANALOG HALL	아날로그 홀센서	1	
31	HALL MAGNETIC	마그네틱 홀센서	1	
32	18B20 TEMP	온도센서	1	
33	ANALOG TEMP	아날로그온도센서	1	
34	IR TRANSMIT	IR 적외선송신	1	
35	IR RECEIVE	IR 적외선수신	1	
36	TAP MODULE	노크센서	1	
37	LIGHT BLOCKING	포트인트럽터	1	

□ 조이스틱 모듈

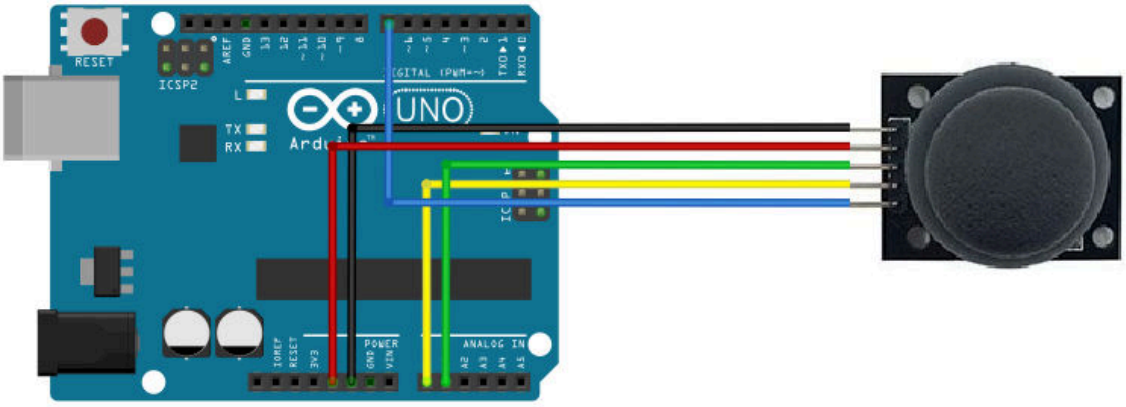
○ 모듈개요

조이스틱모듈은 게임기 조작에서 쉽게 볼 수 있는 스위치모듈입니다. 아날로그 X, Y축 이동은 아날로그 신호로 출력되고, 버튼은 디지털 신호를 출력합니다.



PIN	이름	기 능
1	SW	버튼(디지털출력)
2	VRy	Y축(아날로그 출력)
3	VRx	X축(아날로그 출력)
4	5V	VCC
5	GND	GND

○ 연결도



○ 소스코드

```
//*********************************************************************************
// Project : joystick in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************

int JoyStick_X = A0; // x-axis
int JoyStick_Y = A1; // y-axis
int JoyStick_Z = 7;  // key-axis
void setup ()
{
  Serial.begin (9600); // speed 9600
  pinMode (JoyStick_X, INPUT);
  pinMode (JoyStick_Y, INPUT);
```

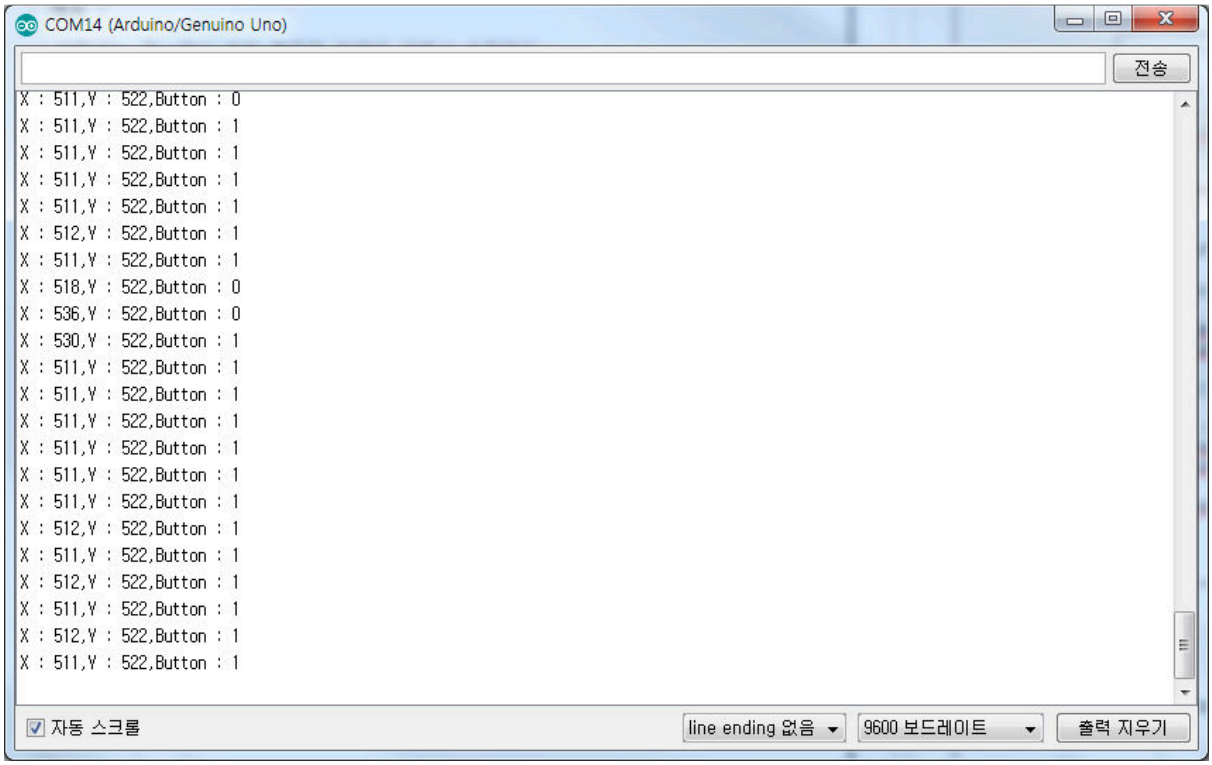
```
  pinMode (JoyStick_Z, INPUT_PULLUP);
}
void loop ()
{
  int x, y, z;
  x = analogRead (JoyStick_X);  // Read X 축
  y = analogRead (JoyStick_Y);  // Read Y 축
  z = digitalRead (JoyStick_Z); // Read 버튼

  Serial.print ("X : ");
  Serial.print (x, DEC);
  Serial.print (",");
  Serial.print ("Y : ");
  Serial.print (y, DEC);
  Serial.print (",");
  Serial.print ("Button : ");
  Serial.println (z, DEC);
  delay (100);
}
```

○ 결과

조이스틱 상하좌우를 이동하면 X,Y 아날로그 좌표가 변합니다. (0~1024 범위)

버튼을 누르면 0과 1이 출력 됩니다.



□ 릴레이 모듈

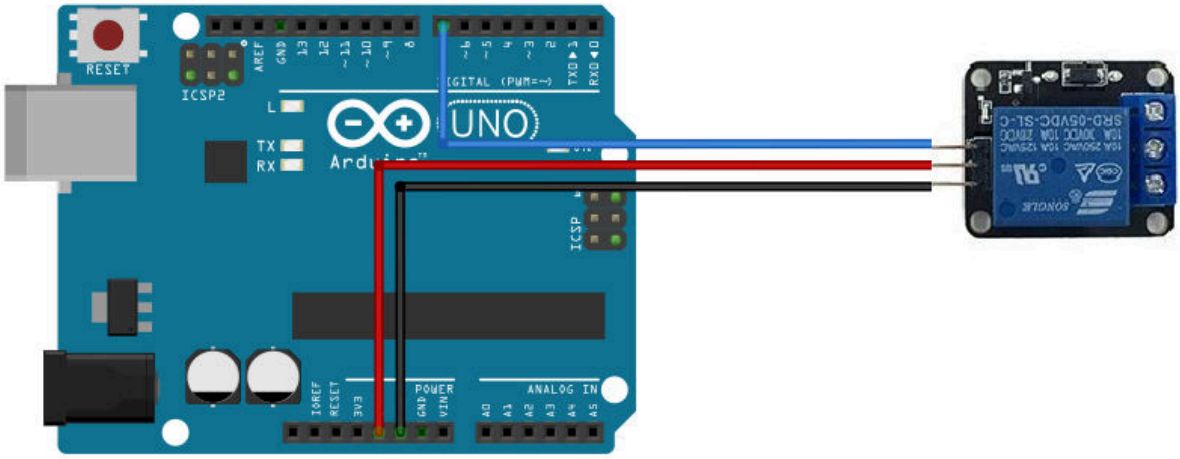
○ 모듈개요

릴레이모듈은 전기적 스위치 종류로 외부 신호에 의해서 열리고 닫히는 스위치모듈입니다. 5V 입력전압으로 고전압을 제어할 수 있습니다. 내부 코일이 동작할 때 "딸깍" 소리가 들립니다.



PIN	이름	기 능
1	-	GND
2	+	5V
3	S	Signal(디지털 입력)

○ 연결도



○ 소스코드

```
//*****************************************************************************
// Project : Relay in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************

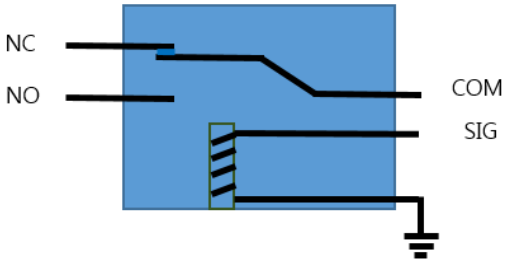
int relay = 7; // relay trigger signal(active high)
void setup ()
{
  pinMode (relay, OUTPUT); // Define port attribute is output;
}
void loop ()
{
  digitalWrite (relay, HIGH); // relay conduction;
```

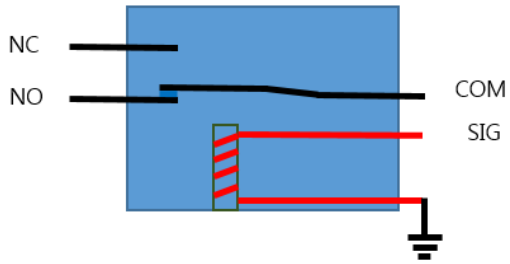
```
  delay (1000);
  digitalWrite (relay, LOW); // relay switch is turned off;
  delay (1000);
}
```

○ 결과확인

"딸깍" 소리로 동작 확인이 가능하며, 5V 신호 인가시 COM단자와 NO단자가 연결됩니다.





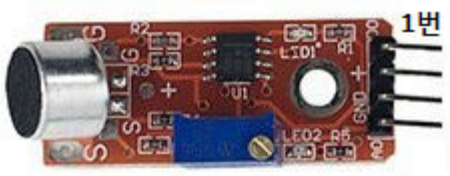


동작 전압: 5V
최대 전류: 10A (30VDC, 250VAC)

□ 고감도사운드모듈

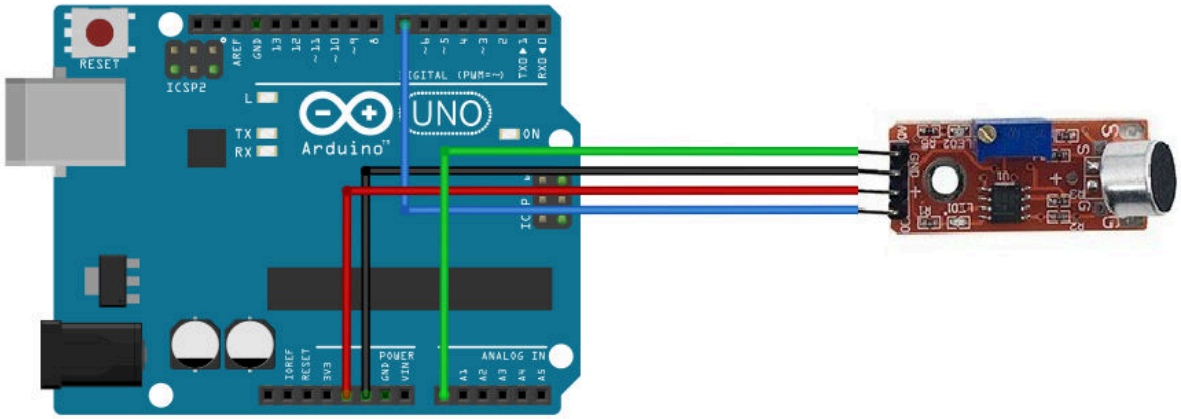
○ 모듈개요

고감도사운드모듈은 외부의 소리파동을 전기적 신호로 바꿔주는 센서 모듈입니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

○ 연결도



○ 소스코드

```
//************************************************************************************
// Project : big sound in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************************
int sensorPin = A0; // Sensor ouput
int ledPin = 12;     // for the LED
int sensorValue = 0; // Sensor Data

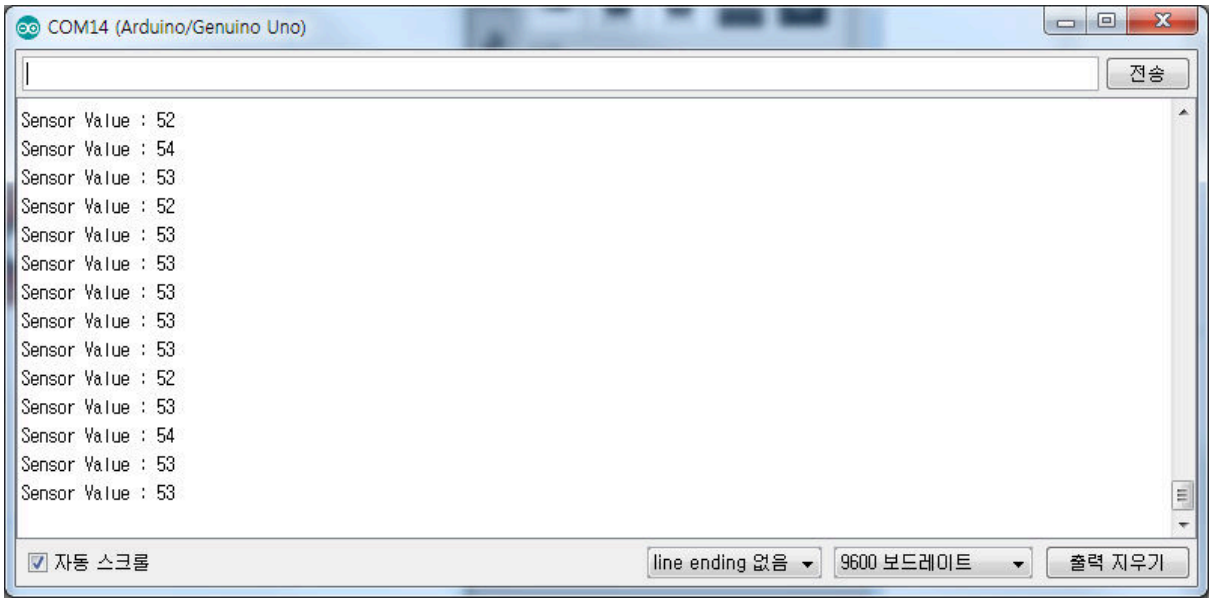
void setup ()
{
  pinMode (ledPin, OUTPUT);
  Serial.begin (9600);
}

void loop ()
{
```

```
  sensorValue = analogRead (sensorPin);
  digitalWrite (ledPin, HIGH);
  delay (sensorValue);
  digitalWrite (ledPin, LOW);
  delay (sensorValue);
  Serial.print ("Sensor Value : ");
  Serial.println (sensorValue, DEC);
}
```

○ 결과확인

소리크기에 따라 시리얼 모니터의 값과 LED가 상태가 바뀌는 것을 확인할 수 있습니다.



□ 소형사운드 모듈

○ 모듈개요

소형사운드모듈은 고감도사운드 모듈과 유사합니다. 외부의 소리파동을 인식해서 전기적 신호로 바꿔주는 센서모듈입니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

○ 연결도

고감도 사운드 모듈과 동일합니다.

○ 소스코드

고감도 사운드 모듈과 동일합니다.

○ 결과확인

고감도 사운드 모듈과 동일합니다.

□ IR 트래킹 모듈

○ 모듈개요

IR 트래킹 모듈은 적외선 발광부에서는 적외선을 보내고 수신부에서는 물체에 반사되어 돌아오는 적외선을 인식하여 거리와 컬러로 인식할 수 있는 모듈입니다. 라인트레이서와 같은 로봇에 응용 될 수 있습니다.

1)



PIN	이름	기 능
1	S	Signal(디지털 출력)
2	V+	5V
3	G	GND

2)

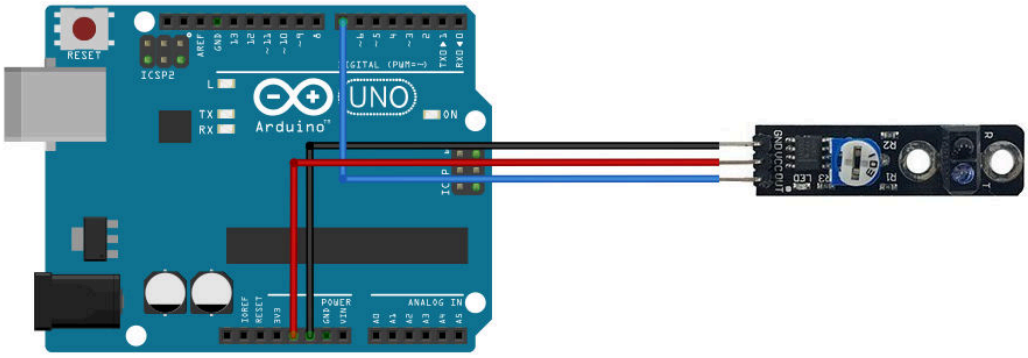


PIN	이름	기 능
1	VCC	5V
2	OUT	Signal(디지털 출력)
3	G	GND

○ 연결도

아래 연결도는 1)번 IR 트래킹 모듈 모양에 맞춰 되어있습니다.

2)번의 모듈일 경우 5V와 OUT 신호를 변경하셔야 합니다.



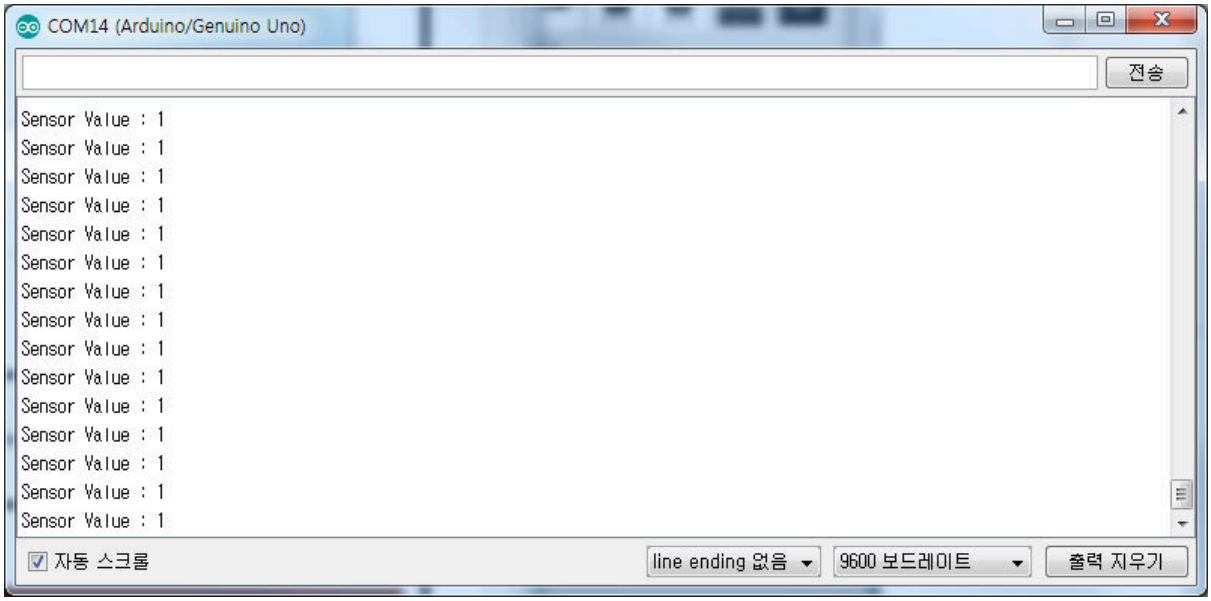
○ 소스코드

```
//*****************************************************************************
// Project : IR tracking in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int sensorPin = 7;  // Select IR tracking Input
int ledPin = 13;     // select the pin for the LED
int sensorValue = 0; // variable for Sensor Data
void setup () {
  pinMode (ledPin, OUTPUT);
  Serial.begin (9600);
}
void loop () {
  sensorValue = digitalRead (sensorPin);
  if(sensorValue)
    digitalWrite (ledPin, HIGH);
  else
    digitalWrite (ledPin, LOW);
  Serial.print ("Sensor Value : ");
  Serial.println (sensorValue, DEC);

  delay (500);
}
```

○ 결과확인

거리 또는 색상에 따라 수신되는 디지털 출력의 변화를 확인 할 수 있습니다.



□ IR 장애물감지 모듈

○ 모듈개요

IR 장애물감지 모듈은 적외선 발광부에서는 적외선을 보내고 수신부에서는 물체에 반사되어 돌아오는 적외선을 인식하여 전방에 장애물을 인식할 수 있는 모듈입니다.

1)



PIN	이름	기 능
1	EN	ENABLE
2	OUT	Signal(디지털 출력)
3	+	5V
4	GND	GND

2)

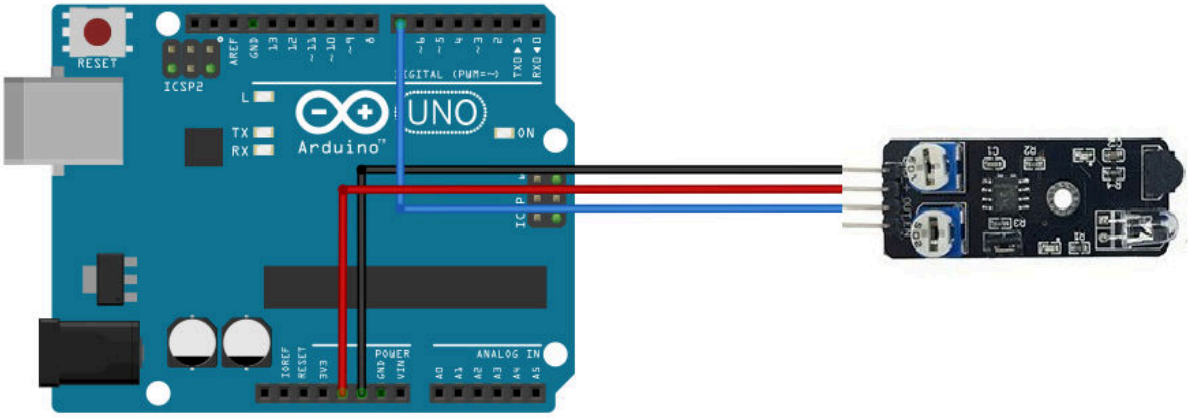


PIN	이름	기 능
1	GND	GND
2	OUT	Signal(디지털 출력)
3	+	5V
4	EN	ENABLE

○ 연결도

아래 연결도는 1)번 IR 장애물감지 모듈 모양에 맞춰 되어있습니다.

2)번의 모듈일 경우 핀맵을 변경하셔야 합니다.

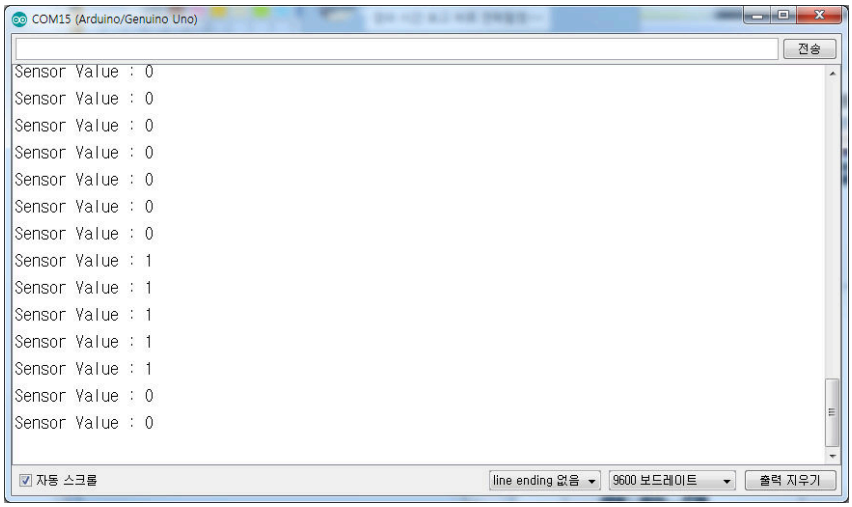


○ 소스코드

```
//*********************************************************************************
// Project : 6_AVOIDANCE in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int DigitalPin =7;  // Digital input
int Led = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup()
{
  Serial.begin (9600);
  pinMode(Led, OUTPUT); // LED pin as output
  pinMode(DigitalPin, INPUT); //Avoidance  pin as input
}
void loop()
{
  gValue=! digitalRead(DigitalPin); //read the value of the sensor
  if(gValue == HIGH) // turn on LED when sensor is blocked
  {
    digitalWrite(Led,HIGH);
  }
  else
  {
    digitalWrite(Led,LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인

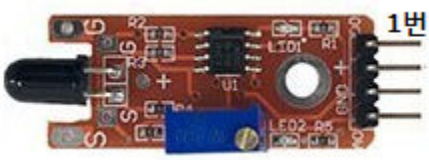
장애물 인식할 수 있는 거리 및 감도는 가변저항으로 조절이 가능합니다.



□ 불꽃감지센서 모듈

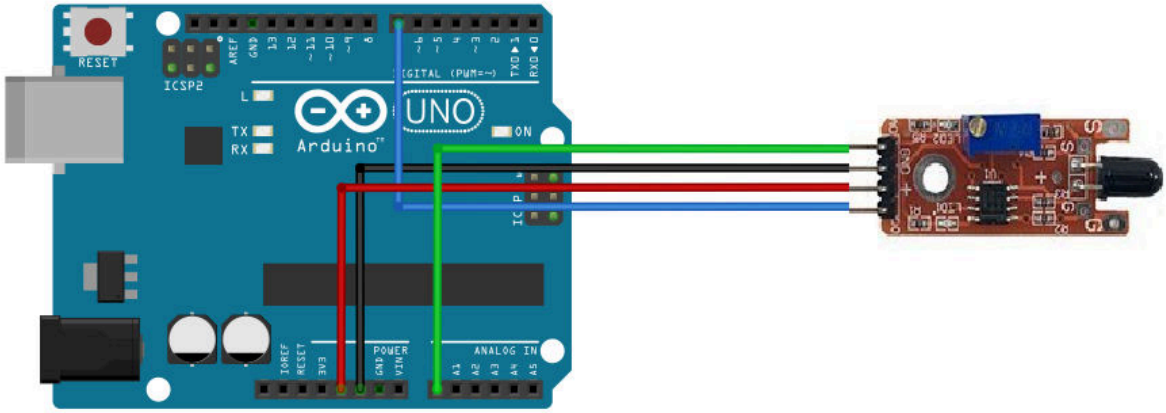
○ 모듈개요

불꽃감지센서는 불꽃 또는 화염을 감지할 수 있는 센서입니다. 불꽃에서 발생되는 760nm ~1100nm의 적외선파장을 감지하는 센서입니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

○ 연결도



○ 소스코드

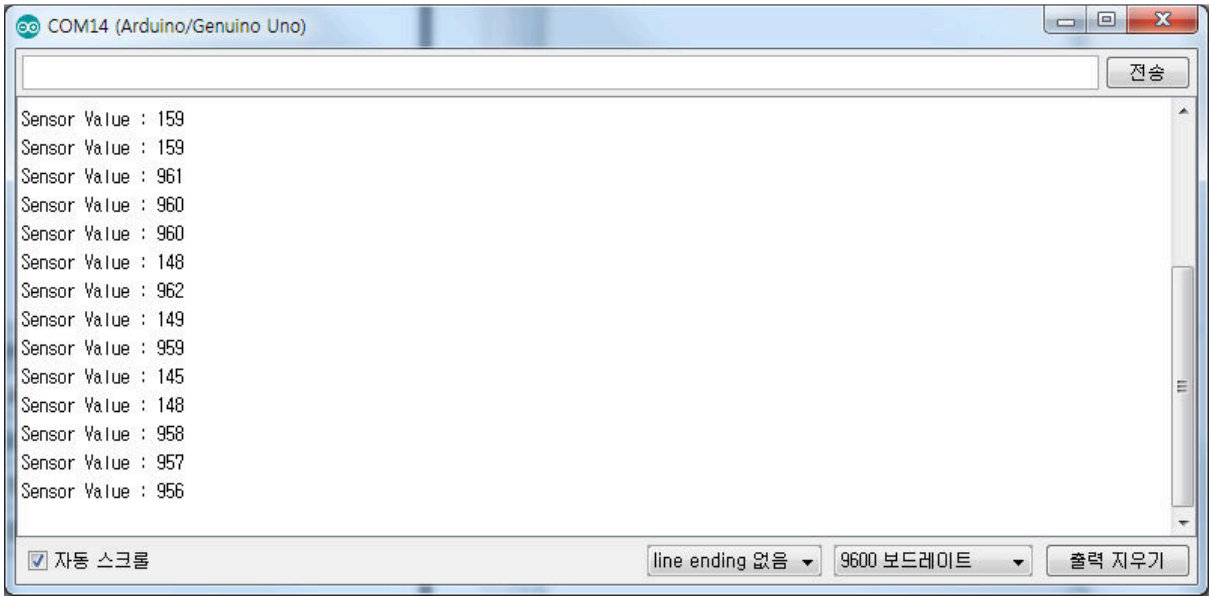
```
//************************************************************************************
// Project : 07 Flame in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
void loop ()
```

```
{
  gValue = analogRead(AnalogPin);

  if (digitalRead (DigitalPin)  ){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인

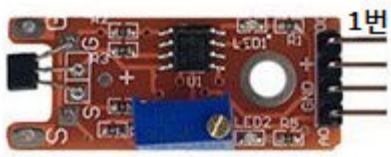
불꽃이 감지되면 값이 상승합니다.



□ 리니어홀센서 모듈

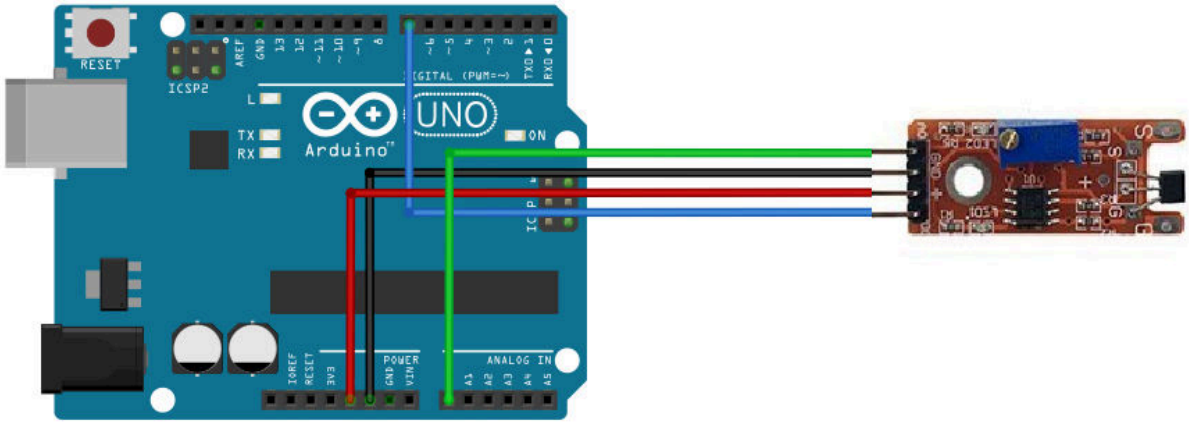
○ 모듈개요

리니어 홀센서모듈은 자석(자기장)에 반응하는 센서모듈입니다. N극과 S극 구분이 가능합니다. 도어 출입감지시스템으로 응용될 수 있습니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

○ 연결도



○ 소스코드

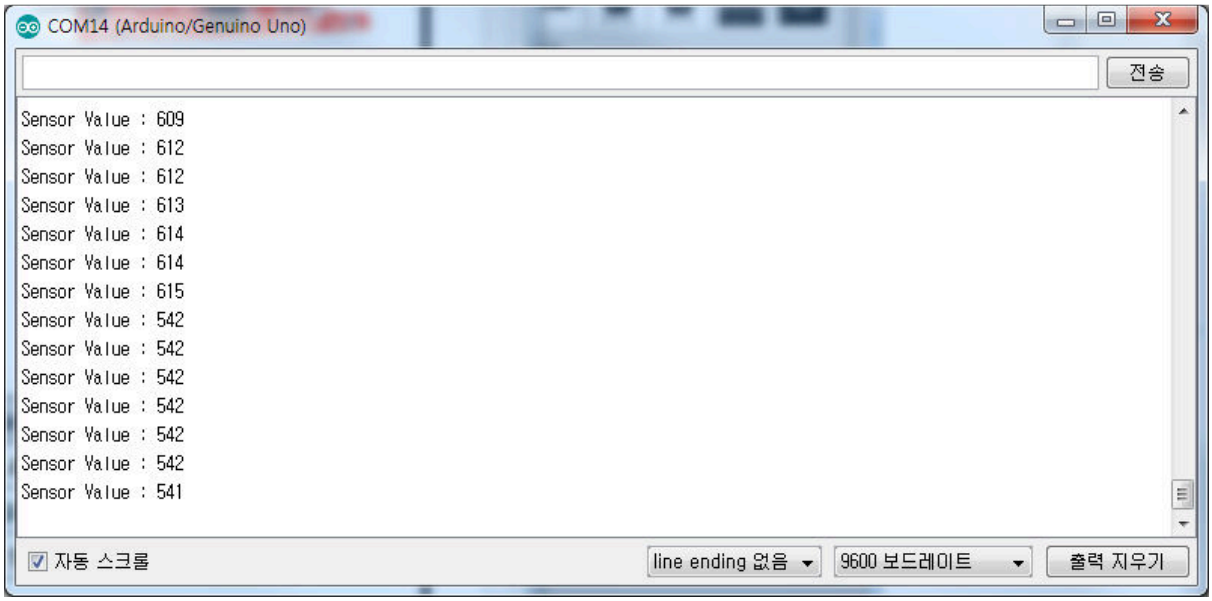
```
//*************************************************************************
// Project : 08 linear hall in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
```

```
  Serial.begin (9600);
}
void loop ()
{
  gValue = analogRead(AnalogPin);

  if (digitalRead (DigitalPin)  ){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인

자석(자기장)을 인식하면 입력되는 아날로그 값이 변화되는 것을 확인할 수 있습니다.



□ 터치센서 모듈

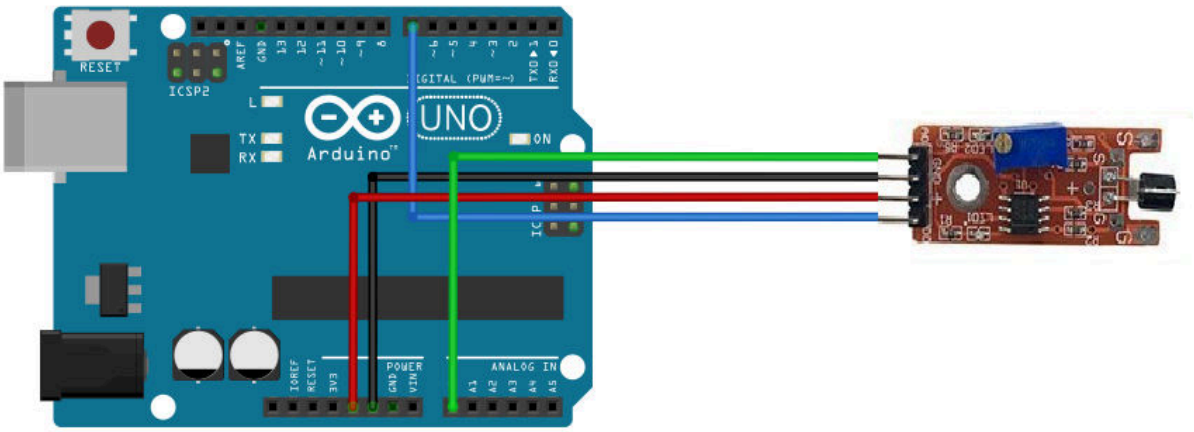
○ 모듈개요

터치센서모듈는 금속성 물체 또는 사람의 피부를 감지하는 센서모듈입니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

○ 연결도



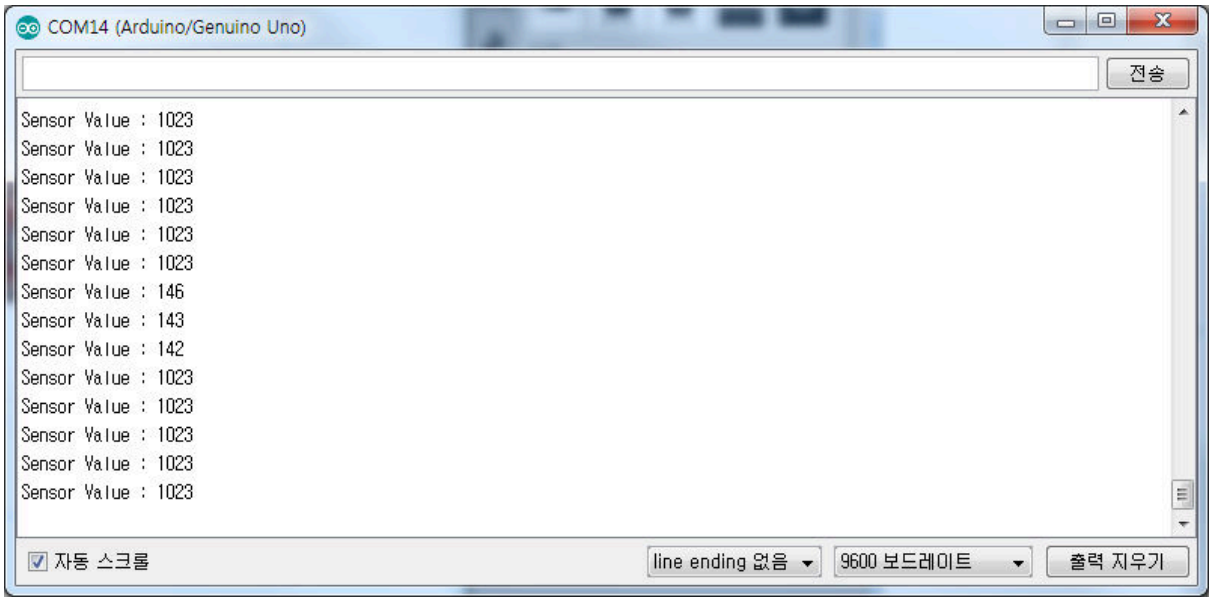
○ 소스코드

```
//*********************************************************************************
// Project : 09 touch in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
```

```
}
void loop ()
{
  gValue = analogRead(AnalogPin);

  if (digitalRead (DigitalPin)  ){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인



□ 디지털 온도센서 모듈

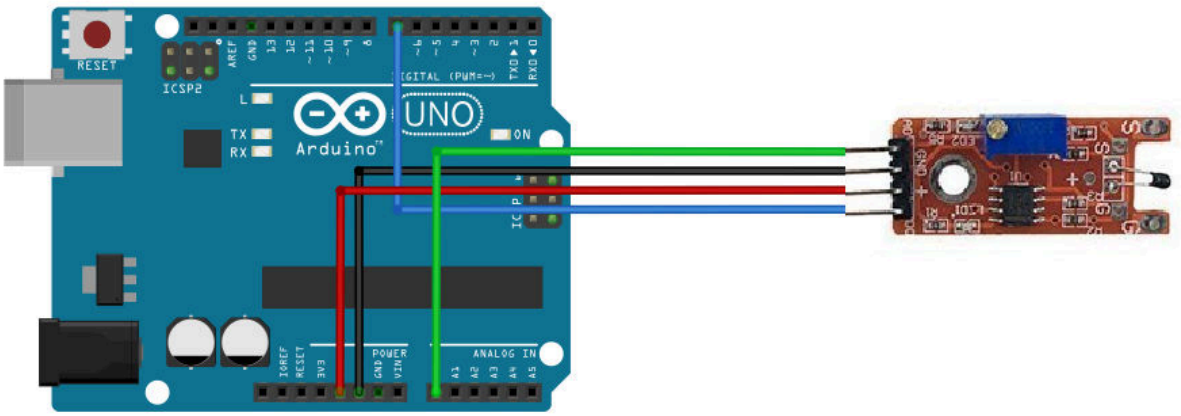
○ 모듈개요

디지털 온도센서 모듈은 주변 온도값에 따라 저항 값이 변하는 서미스터가 장착되어있는 센서모듈입니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

○ 연결도



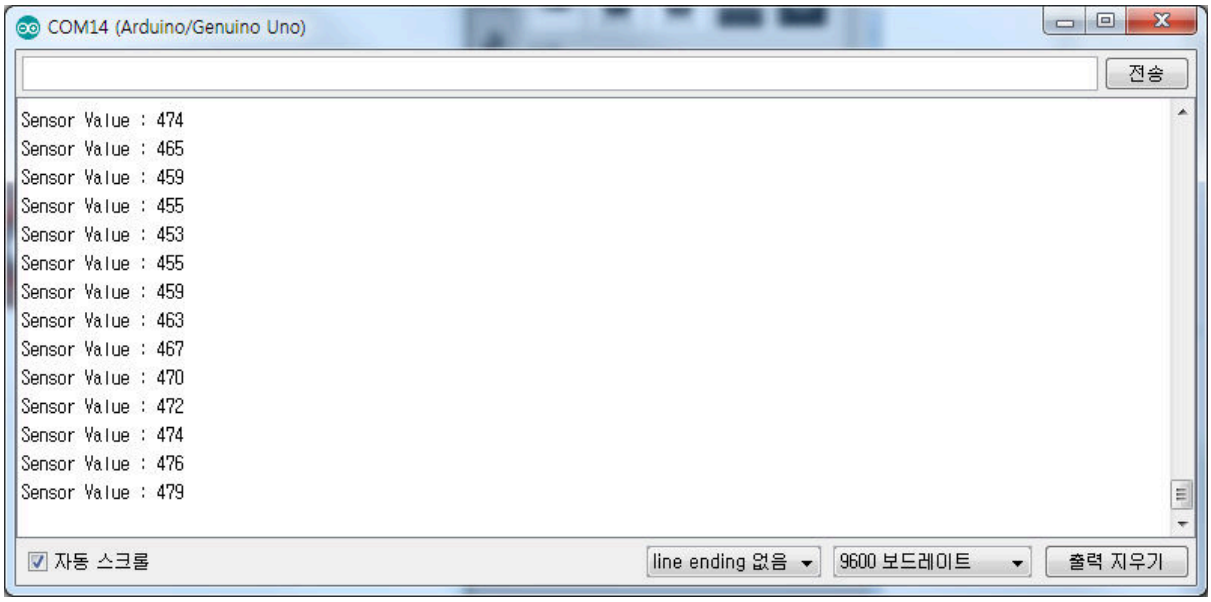
○ 소스코드

```
//*********************************************************************************
// Project : 10  digital temperature in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
```

```
void loop ()
{
  gValue = analogRead(AnalogPin);

  if (digitalRead (DigitalPin)  ){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인



□ 회로내장부저 모듈

○ 모듈개요

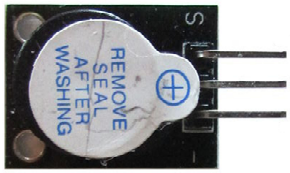
회로내장부저모듈은 발진회로가 내장되어있는 부저모듈입니다. 회로가 내장되어있기 때문에 동일한 음색과 크기로 비프음이 출력이 됩니다.

1)



PIN	이름	기 능
1	-	GND
2	+	5V
3	S	Signal(디지털 입력)

2)

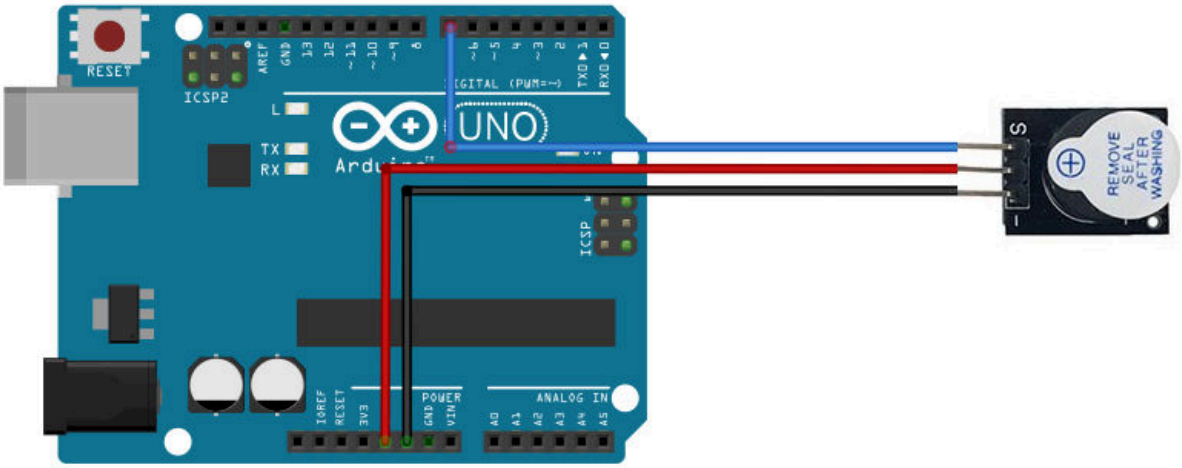


PIN	이름	기 능
1	S	Signal(디지털 입력)
2	+	5V
3	-	GND

○ 연결도

아래 연결도는 1)번 부저 모양에 맞춰 되어있습니다.

2)번 부저 경우는 "S"와 "–"를 반대로 연결 해야 합니다.



○ 소스코드

```
//*****************************************************************************
// Project : 11 Buzzer in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int DigitalPin = 7;  // Digital input
void setup ()
{
  pinMode (DigitalPin, OUTPUT);
  Serial.begin (9600);
}
void loop ()
{
  digitalWrite (DigitalPin, HIGH);
  delay(1000);
  digitalWrite (DigitalPin, LOW);
  delay(1000);
}
```

○ 결과확인

1 초 주기로 비프음을 확인할 수 있습니다.

□ 무회로부저 모듈

○ 모듈개요

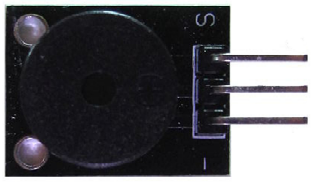
무회로장부저 모듈은 회로가 내장되어있는 않는 부저모듈입니다. 또는 수동/패시브 부저라고도 부릅니다. 소리(비프음)을 발생하기 위해서는 별도의 발진신호(PWM)를 입력해 주어야 합니다. 그러므로 다양한 음색 출력이 가능합니다.

1)번



PIN	이름	기 능
1	-	GND
2	+	5V
3	S	Signal(PWM 입력)

2)번

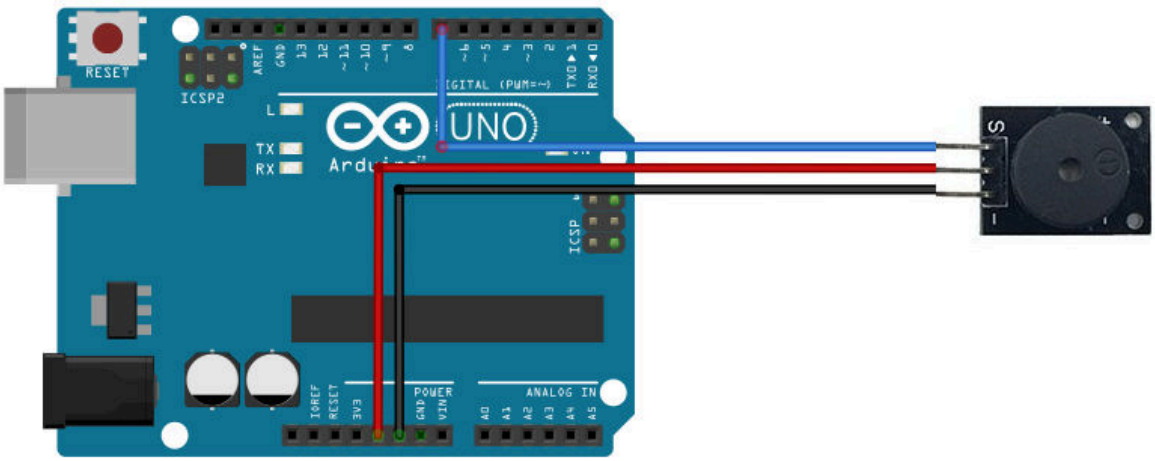


PIN	이름	기 능
1	S	Signal(PWM 입력)
2	+	5V
3	-	GND

○ 연결도

아래 연결도는 1)번 부저 모양에 맞춰 되어있습니다.

2)번 부저 경우는 "S"와 "–"를 반대로 연결 해야 합니다.



○ 소스코드

```
//*****************************************************************************
// Project : 12 Passive Buzzer in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int DigitalPin = 7;  // Digital input
void setup ()
{
  pinMode (DigitalPin, OUTPUT);
  Serial.begin (9600);
}
void loop ()
{
   int i;
    for (i = 0; i <80; i++)
    {
      digitalWrite (DigitalPin, HIGH) ;
      delay (1) ;// Delay 1ms
      digitalWrite (DigitalPin, LOW) ;
      delay (1) ;// delay ms
    }
   delay(1000);
    for (i = 0; i <100; i++)
    {
      digitalWrite (DigitalPin, HIGH) ;
      delay (2) ;// delay 2ms
      digitalWrite (DigitalPin, LOW) ;
      delay (2) ;// delay 2ms
    }
   delay(1000);
}
```

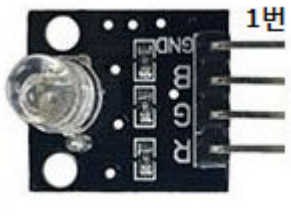
○ 결과확인

소리로 두 가지 음색을 확인할 수 있습니다.

□ 3 컬러 LED 모듈

○ 모듈개요

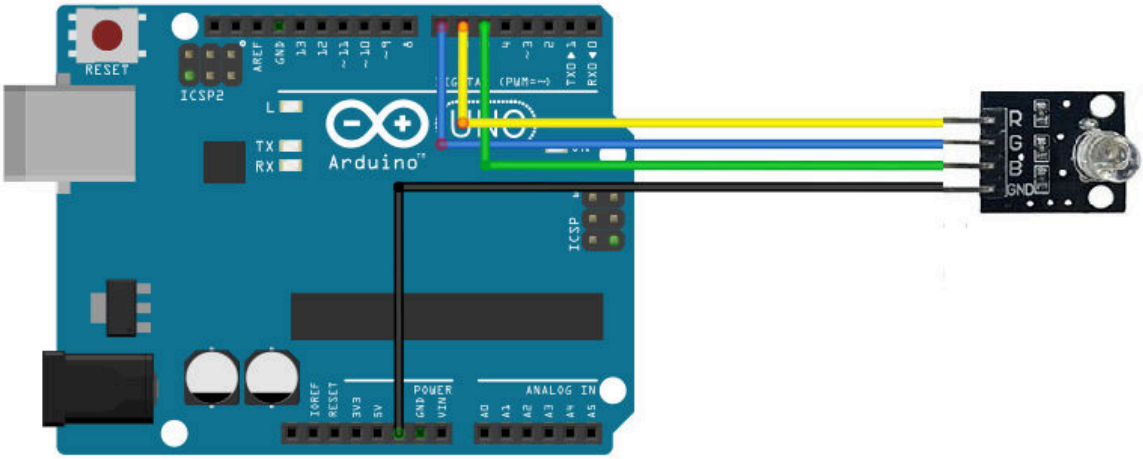
3컬러 LED모듈은 Red, Green, Blue LED가 내장되어있어 풀 컬러 표현이 가능한 LED모듈입니다. 풀 컬러를 표현하기 위해서는 Red, Green, Blue 신호 각각 밝기 조절(PWM)이 필요합니다.



PIN	이름	기 능
1	-	GND
2	B	Blue Signal(PWM 가능)
3	R	Red Signal(PWM 가능)
4	G	Green Signal(PWM 가능)

R, G, B 실크 위치가 다를 수 있습니다.

○ 연결도



○ 소스코드

```
//*********************************************************************************
// Project : 13_LED in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int RED= 6;  // Digital input
int GREEN= 7;  // Digital input
int BLUE= 5;  // Digital input
void setup ()
{
  pinMode (RED, OUTPUT);
  pinMode (GREEN, OUTPUT);
```

```
  pinMode (BLUE, OUTPUT);
}
void loop ()
{
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, LOW);
  digitalWrite (BLUE, LOW);
  delay(1000);
  digitalWrite (RED, LOW);
  digitalWrite (GREEN, HIGH);
  digitalWrite (BLUE, LOW);
  delay(1000);
  digitalWrite (RED, LOW);
  digitalWrite (GREEN, LOW);
  digitalWrite (BLUE, HIGH);
  delay(1000);
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, HIGH);
  digitalWrite (BLUE, HIGH);
  delay(1000);
}
```

○ 결과확인

LED 확인(RED -> GREEN -> BLUE -> WHITE)

□ 3 컬러 SMD LED 모듈

○ 모듈개요

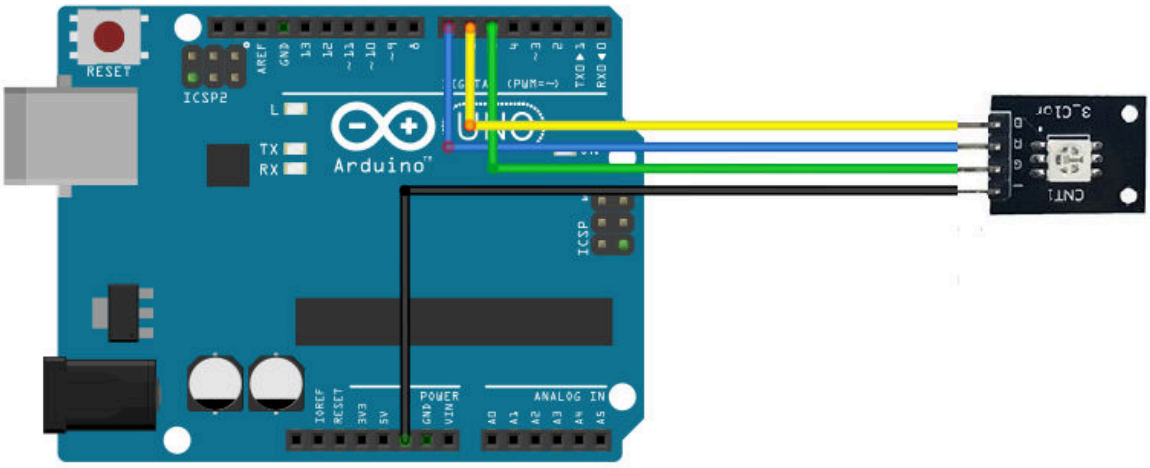
3컬러 SMD LED모듈은 Red, Green, Blue LED가 내장 되어있는 풀컬러 표현이 가능한 LED 모듈입니다. LED package는 SMD타입 입니다.



PIN	이름	기 능
1	-	GND
2	G	Green Signal(PWM 입력)
3	R	Red Signal(PWM 입력)
4	B	Blue Signal(PWM 입력)

**R, G, B 실크 위치가 다를 수 있습니다.*

+○ 연결도



○ 소스코드

```
//*****************************************************************************
// Project : 14_SMD LED in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int RED= 7;  // Digital input
int GREEN= 5;  // Digital input
int BLUE= 6;  // Digital input
void setup ()
{
  pinMode (RED, OUTPUT);
  pinMode (GREEN, OUTPUT);
  pinMode (BLUE, OUTPUT);
}
```

```
void loop ()
{
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, LOW);
  digitalWrite (BLUE, LOW);
  delay(1000);
  digitalWrite (RED, LOW);
  digitalWrite (GREEN, HIGH);
  digitalWrite (BLUE, LOW);
  delay(1000);
  digitalWrite (RED, LOW);
  digitalWrite (GREEN, LOW);
  digitalWrite (BLUE, HIGH);
  delay(1000);
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, HIGH);
  digitalWrite (BLUE, HIGH);
  delay(1000);
}
```

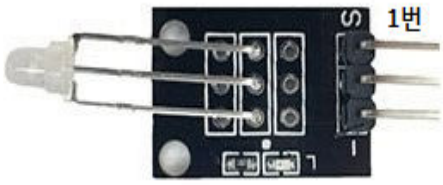
○ 결과확인

LED 확인(RED -> GREEN -> BLUE -> WHITE)

□ 2 컬러 LED 모듈

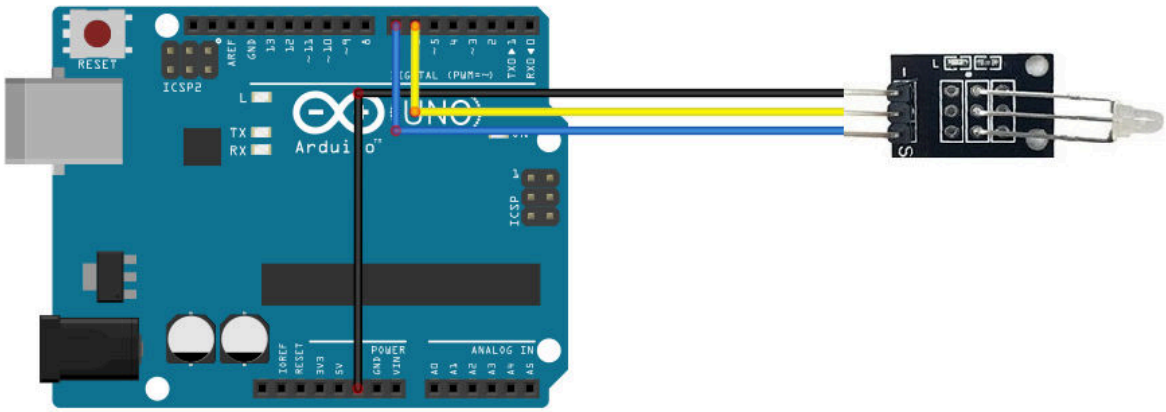
○ 모듈개요

2컬러 LED모듈은 Red, Green LED가 내장 되어있는 LED모듈입니다.



PIN	이름	기 능
1	S	Red Signal(PWM 가능)
2		Green Signal(PWM 가능)
3	-	GND

○ 연결도



○ 소스코드

```
//*****************************************************************************
// Project : 15_TWO_LED in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int RED = 6;  // Digital input
int GREEN = 7;  // Digital input
void setup ()
{
  pinMode (RED, OUTPUT);
  pinMode (GREEN, OUTPUT);

}
void loop ()
{
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, LOW);

  delay(1000);
```

```
  digitalWrite (RED, LOW);
  digitalWrite (GREEN, HIGH);

  delay(1000);
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, HIGH);

  delay(1000);
}
```

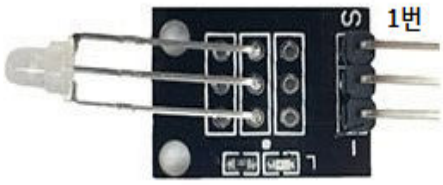
○ 결과확인

LED 확인(RED -> GREEN > YELLOW)

□ 2 컬리 소형 LED 모듈

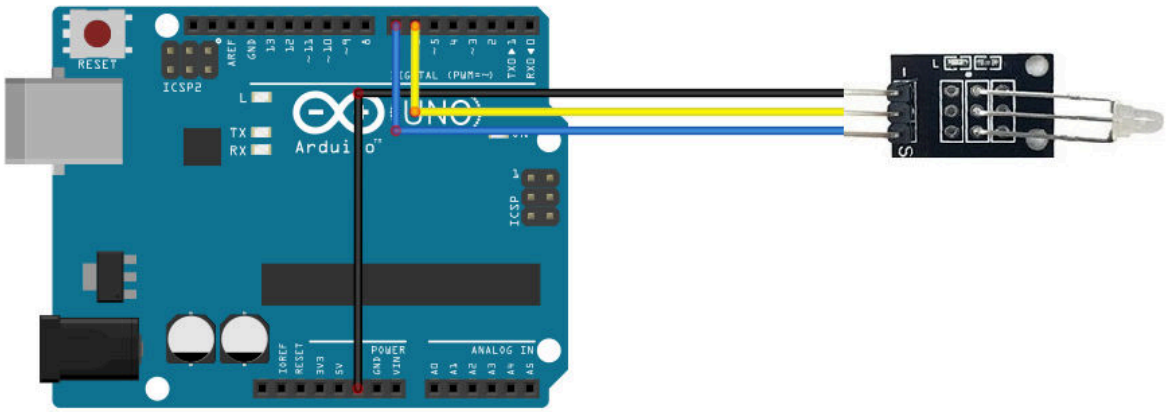
○ 모듈개요

2컬러 LED모듈은 Red, Green LED가 내장 되어있는 소형 LED모듈입니다.



PIN	이름	기 능
1	S	Red Signal(PWM 입력)
2		Green Signal(PWM 입력)
3	-	GND

○ 연결도



○ 소스코드

```
//*****************************************************************************
// Project : 16_MINI TWO_LED in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int RED = 6;  // Digital input
int GREEN = 7;  // Digital input
void setup ()
{
  pinMode (RED, OUTPUT);
  pinMode (GREEN, OUTPUT);

}
void loop ()
{
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, LOW);

  delay(1000);
```

```
  digitalWrite (RED, LOW);
  digitalWrite (GREEN, HIGH);

  delay(1000);
  digitalWrite (RED, HIGH);
  digitalWrite (GREEN, HIGH);

  delay(1000);
}
```

○ 결과확인

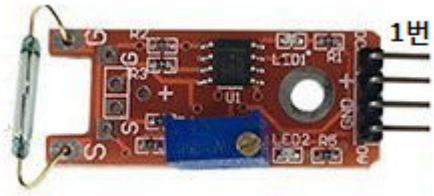
LED 확인(RED -> GREEN -> YELLOW)

□ 리드스위치 모듈

○ 모듈개요

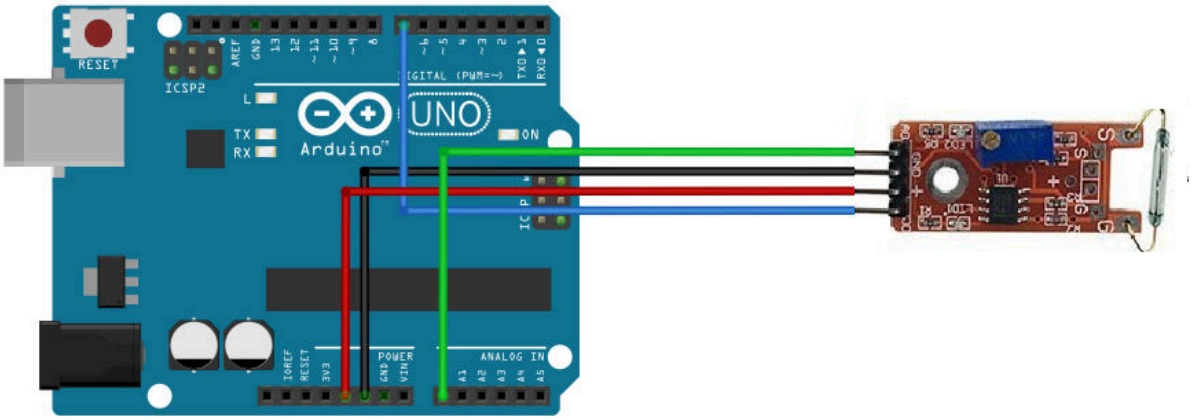
리드스위치모듈은 홀센서와 동일하게 자석(자기장)에 반응하는 센서모듈입니다.

동작원리는 내부 스프링들의 탄성에 의해서 스프링들이 열려있는 상태를 유지하다가 자석(자기장)이 인식되면 두 개의 리드 스프링이 접점 되어 닫히는 구조입니다.



PIN	이름	기 능
1	D0	Signal(디지털 출력)
2	+	5V
3	G	GND
4	A0	Signal(아날로그 출력)

○ 연결도



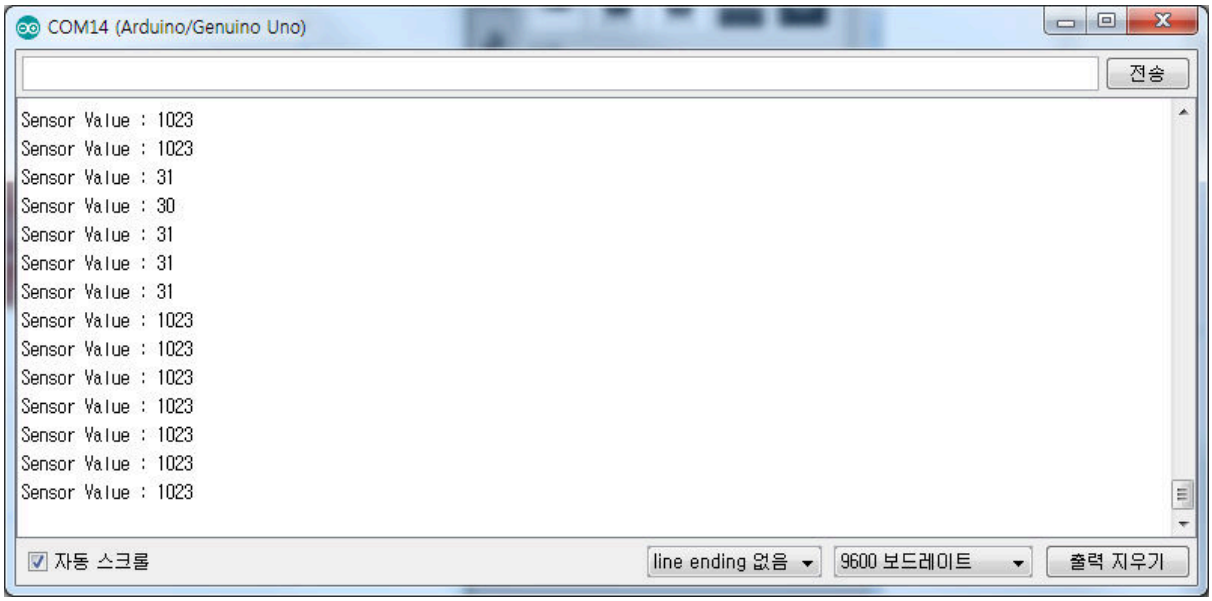
○ 소스코드

```
//*********************************************************************************
// Project : 17 reed in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
```

```
}
void loop ()
{
  gValue = analogRead(AnalogPin);

  if (digitalRead (DigitalPin)  ){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인



□ 미니리드스위치 모듈

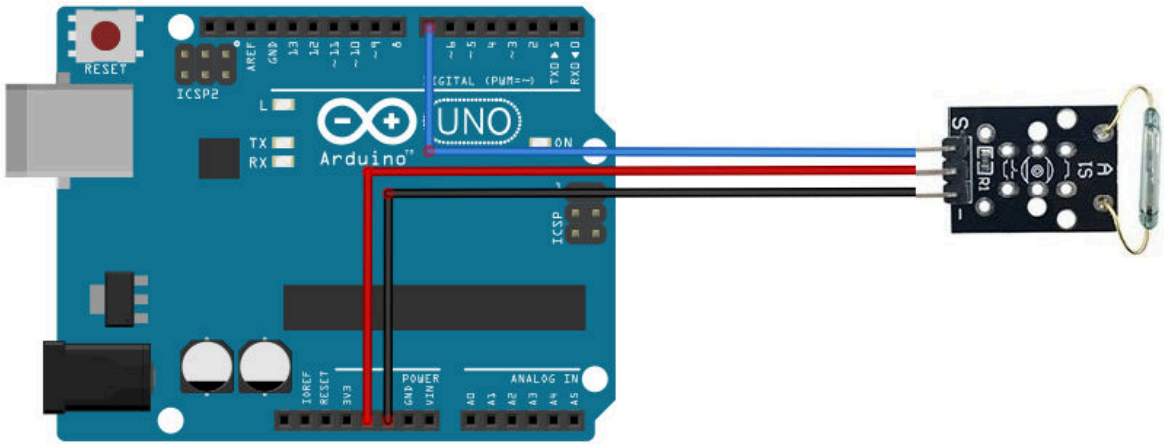
○ 모듈개요

미니 리드스위치모듈은 리드스위치센서와 동일하게 자석(자기장)에 반응하는 센서모듈입니다. 동작원리도 리드스위치 모듈과 동일합니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(디지털 출력)

○ 연결도



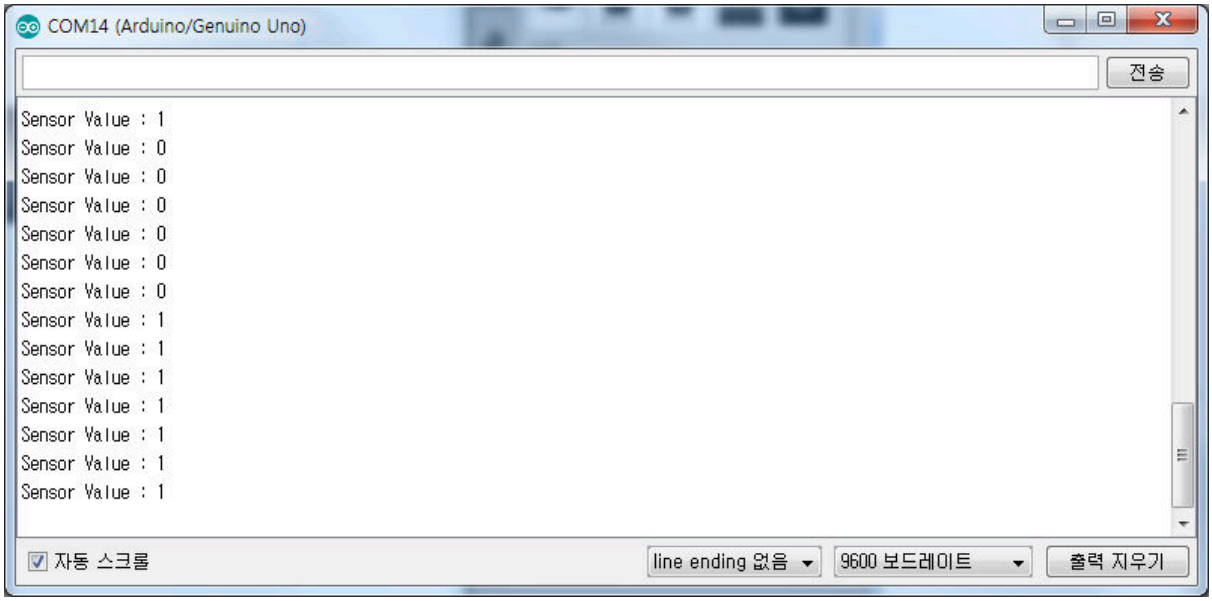
○ 소스코드

```
//*************************************************************************
// Project : 18_MINI_SWITCH in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*************************************************************************
//int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
```

```
  Serial.begin (9600);
}
void loop ()
{
  gValue = digitalRead(DigitalPin);

  if (gValue){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

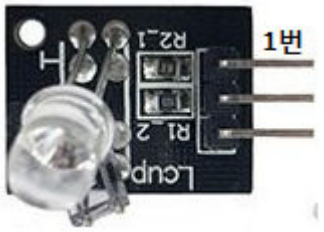
○ 결과확인



□ 손가락맥박센서 모듈

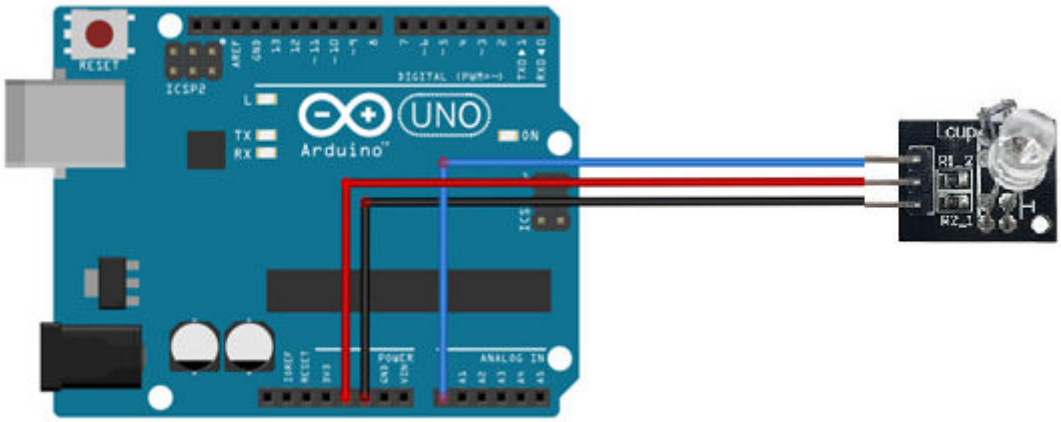
○ 모듈개요

손가락맥박센서모듈은 두 개의 적외선송수신 센서에 손가락을 넣어서 심박을 체크하는 센서모듈입니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(아날로그 출력)

○ 연결도



○ 소스코드

```
//************************************************************************************
// Project : 19_HEARTBREAK in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************************
int sensorPin = 0;
double alpha = 0.75;
int period = 100;
double change = 0.0;
double minval = 0.0;
void setup ()
{
  Serial.begin (9600);
}
void loop ()
```

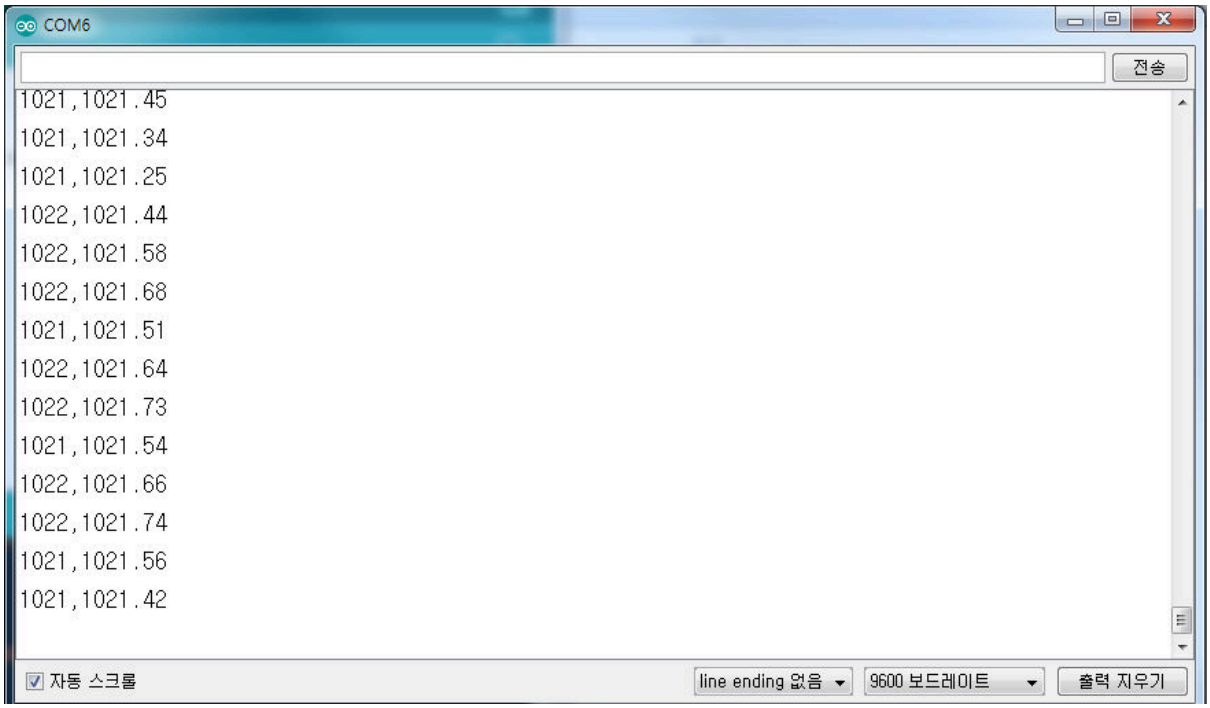
```
{
    static double oldValue = 0;
    static double oldChange = 0;

    int rawValue = analogRead (sensorPin);
    double value = alpha * oldValue + (1 - alpha) * rawValue;

    Serial.print (rawValue);
    Serial.print (",");
    Serial.println (value);
    oldValue = value;

    delay (period);
}
```

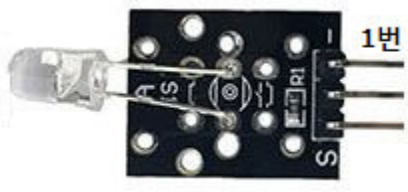
○ 결과확인



□ 7 컬러 플래쉬 모듈

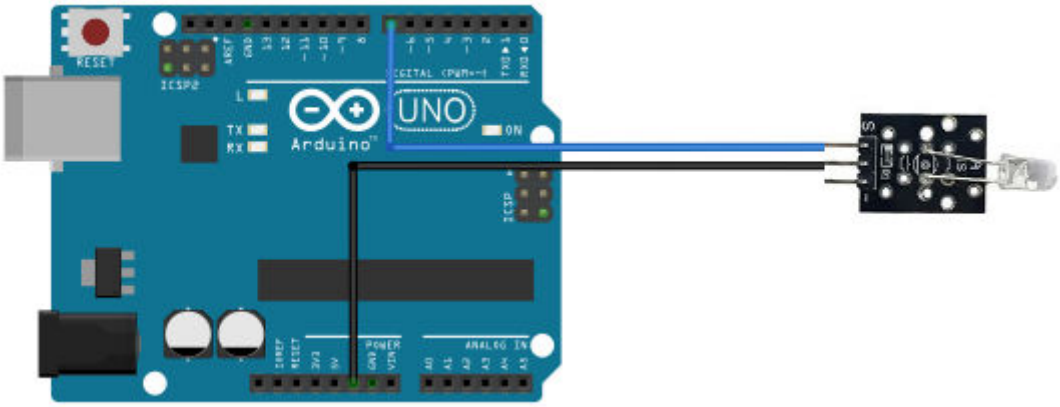
○ 모듈개요

7컬러 플래쉬 모듈 전원이 입력되면 하나의 LED에서 7가지 색상이 연속적으로 표시되는 LED모듈입니다.



PIN	이름	기 능
1	-	연결 없음
2		GND
3	V	Digital IN

○ 연결도



○ 소스코드

```
//*********************************************************************************
// Project : 19_HEARTBREAK in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int sensorPin = 0;
double alpha = 0.75;
int period = 100;
double change = 0.0;
double minval = 0.0;
void setup ()
{
  Serial.begin (9600);
}
void loop ()
{
    static double oldValue = 0;
    static double oldChange = 0;

    int rawValue = analogRead (sensorPin);
```

```
    double value = alpha * oldValue + (1 - alpha) * rawValue;

    Serial.print (rawValue);
    Serial.print (",");
    Serial.println (value);
    oldValue = value;

    delay (period);
}
```

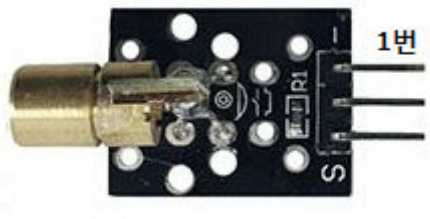
○ 결과확인

7가지 색상이 변화하는 것을 확인

□ 레이저송신 모듈

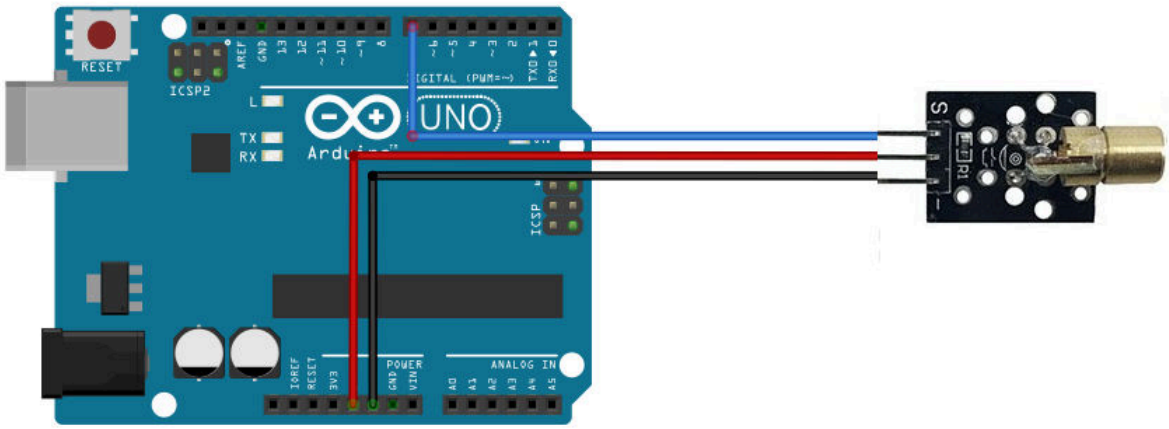
○ 모듈개요

레이저송신모듈은 유도방출에 원리로 빛을 증폭해서 발진하는 레이저모듈입니다. 일반적으로 회의에서 사용되는 레이저포인터와 같은 원리입니다. 주의 사항은 레이저모듈의 광원이 사람 눈에 비치게 되면 치명적일 수 있습니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal (디지털 입력)

○ 연결도



○ 소스코드

```
//*****************************************************************************
// Project : 13_Lager emit in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int DigitalPin = 7;  // Digital input
void setup ()
{
  pinMode (DigitalPin, OUTPUT);
  Serial.begin (9600);
}
void loop ()
{
  digitalWrite (DigitalPin, HIGH);
```

```
  delay(1000);
  digitalWrite (DigitalPin, LOW);
  delay(1000);
}
```

○ 결과확인

1 초 주기로 Laser 발광여부를 확인

□ 버튼스위치 모듈

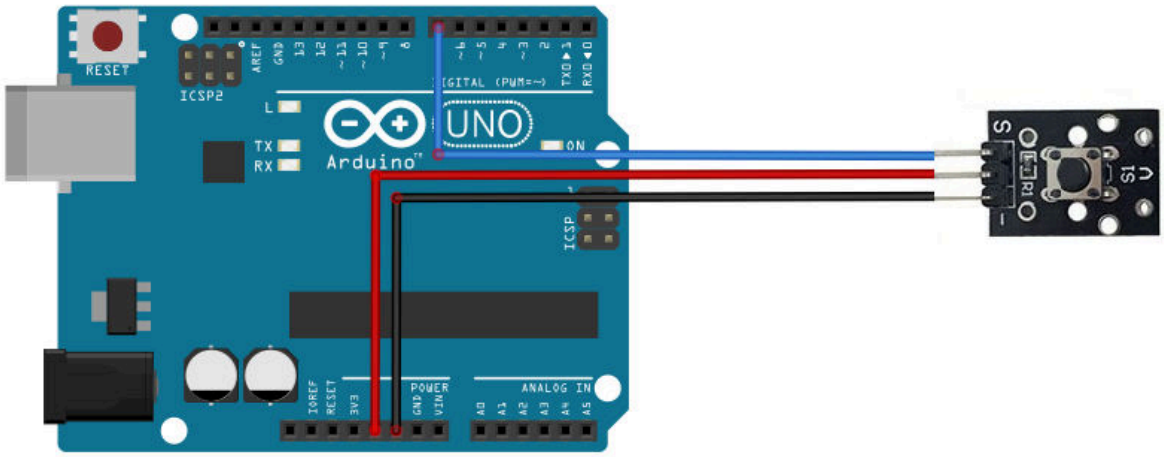
○ 모듈개요

버튼스위치모듈은 일반 택트 스위치로 구성되어있는 스위치 모듈입니다. 버튼을 누르면 스위치가 닫히게 되고 손을 때면 스위치가 열리게 되어있습니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal (디지털 출력)

○ 연결도



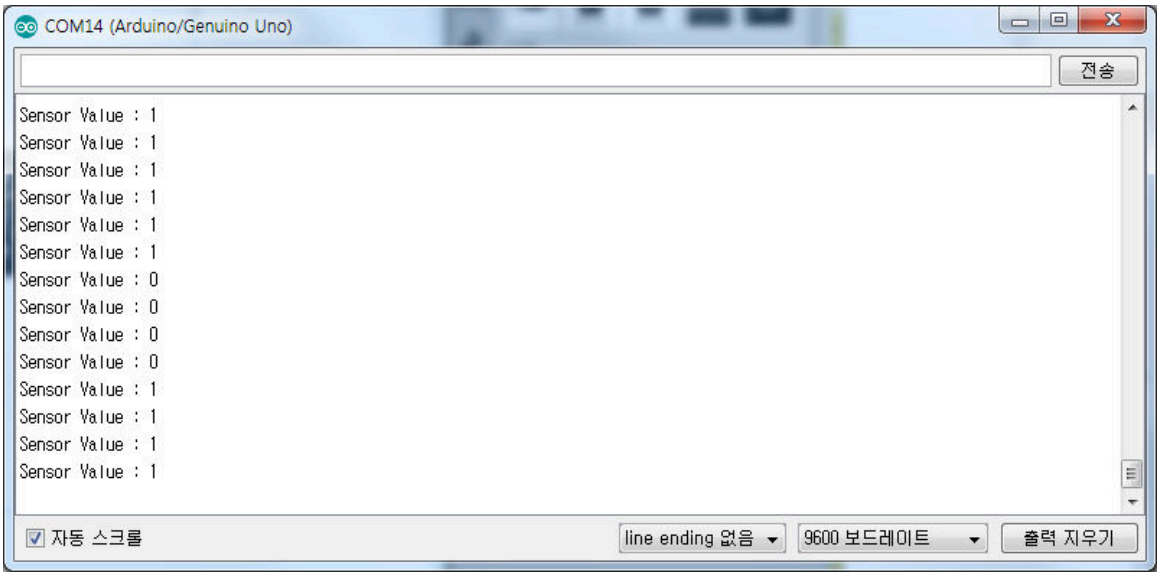
○ 소스코드

```
//********************************************************************************
// Project : 22_BUTTON in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//********************************************************************************
//int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
```

```
}
void loop ()
{
  gValue = digitalRead(DigitalPin);

  if (gValue){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

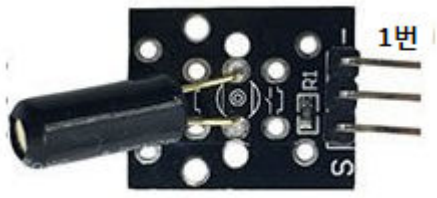
○ 결과확인



□ 충격센서 모듈

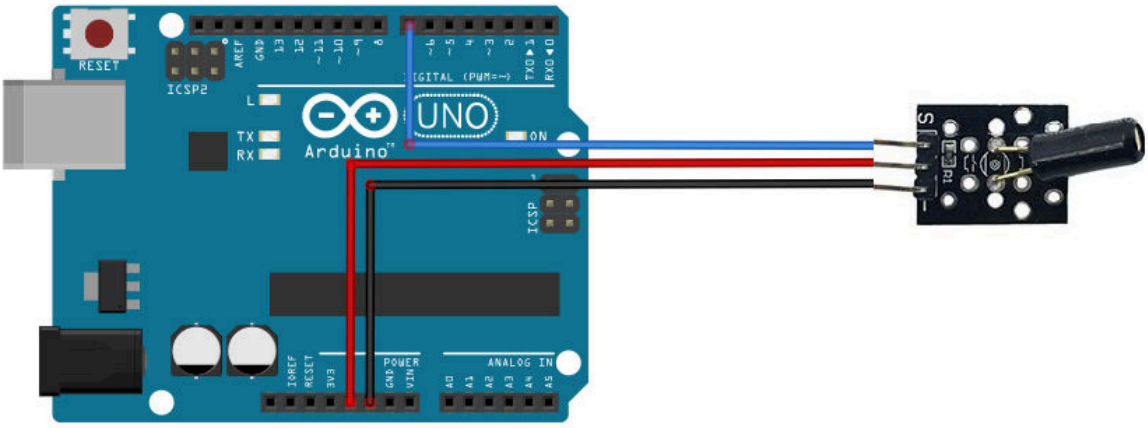
○ 모듈개요

충격센서모듈은 내부에 쇠구슬 또는 스프링이 들어가 있는 스위치모듈 입니다. 충격이 가해지면 쇠구슬 또는 스프링에 의해서 스위치가 열고 닫는 원리입니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal (디지털 출력)

○ 연결도



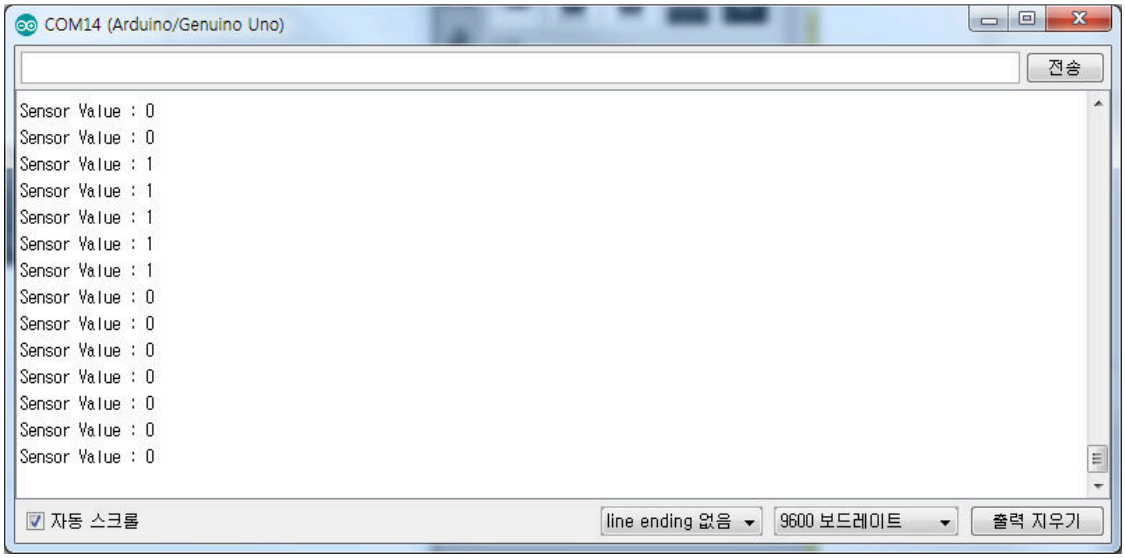
○ 소스코드

```
//************************************************************************
// Project : 23_SHOCK in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************
//int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
```

```
void loop ()
{
  gValue = digitalRead(DigitalPin);

  if (gValue){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
//  delay(500);
}
```

○ 결과확인



□ 로터리엔코더 모듈

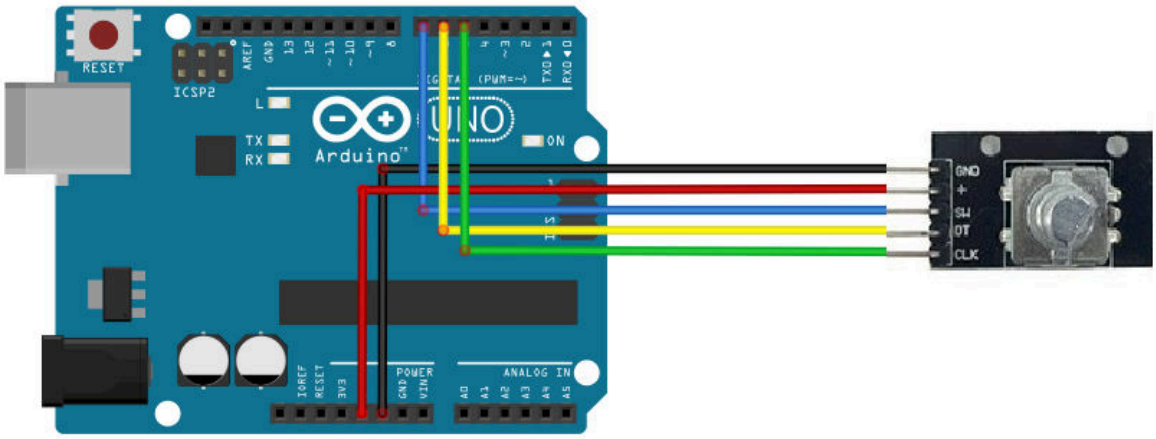
○ 모듈개요

로터리엔코더모듈은 회전을 측정할 수 있는 센서 모듈입니다. 회전 손잡이(엔코더)를 일정 각도로 회전을 하게 되면 디지털 신호가 펄스 단위로 출력이 됩니다. 그리고 Push 버튼 스위치 기능도 포함하고 있습니다.



PIN	이름	기 능
1	CLK	클럭(디지털 출력)
2	DT	데이터(디지털 출력)
3	SW	버튼(디지털 출력)
4	+	5V
5	GND	GND

○ 연결도



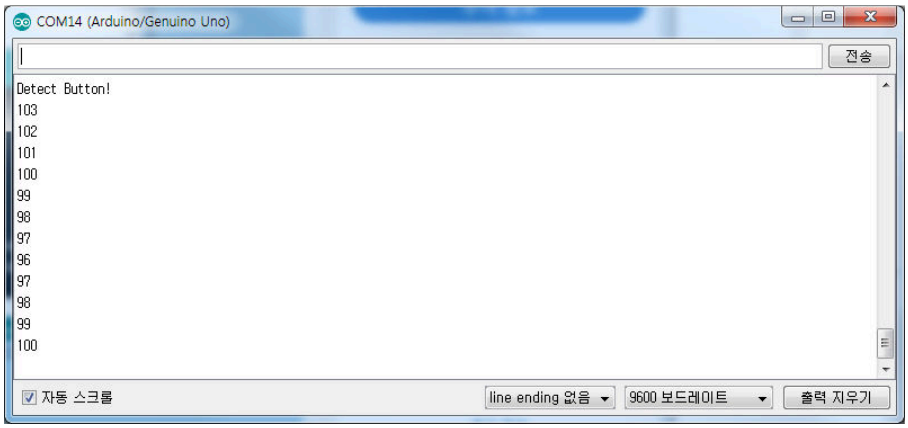
○ 소스코드

```
//*********************************************************************************
// Project : 24_ROTAY_ENCODERS in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int aPin = 5;
int bPin = 6;
int buttonPin = 7;
int temp;
int temprotation = 100;
int rotation = 100;
int gbutten_sts=0;
int getEncoderTurn()
{
  static int oldA = 0;
```

```
  static int oldB = 0;
  int result = 0;
  int newA = digitalRead(aPin);
  int newB = digitalRead(bPin);
  if (newA != oldA || newB != oldB){
    if (!oldA && newA){
      result = -(oldB * 2 - 1);
    }
  }
  oldA = newA;
  oldB = newB;
  return result;
}
void setup()
{
    pinMode(aPin, INPUT);
    pinMode(bPin, INPUT);
    pinMode(buttonPin, INPUT);
    digitalWrite(buttonPin, HIGH); // pull up setting
    Serial.begin(9600);
}
void loop()
{
  int change = getEncoderTurn();
  if(change != temp) {
    rotation = rotation + change;
    if(rotation != temprotation){
      Serial.println(rotation);
    }
    temprotation = rotation;
  }
  temp = change;
  if( !digitalRead(buttonPin) && gbutten_sts==0){
    Serial.println("Detect Button!");
    gbutten_sts=1;
  }else{
    gbutten_sts=0;
  }

  delay(1);
}
```

○ 결과확인



□ 매직컵라이트 모듈

○ 모듈개요

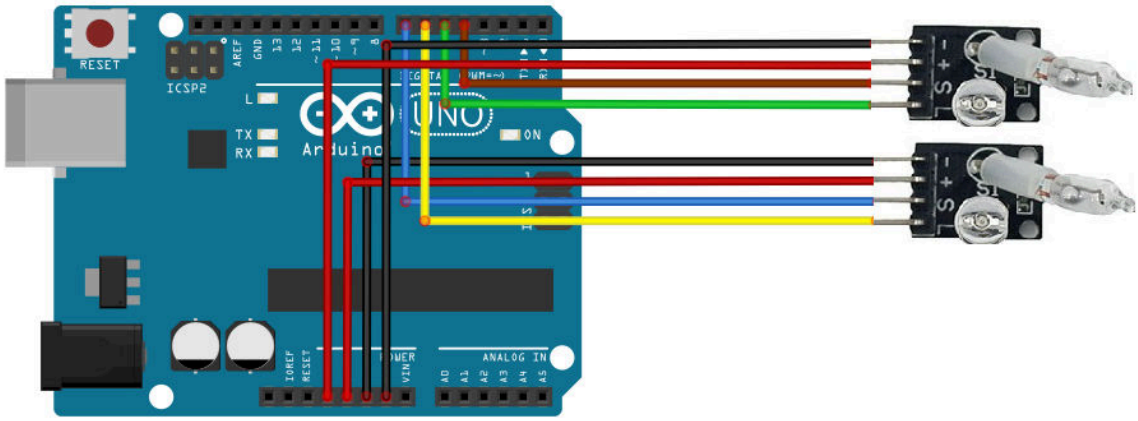
매직컵라이트모듈은 기울기센서와 LED가 결합된 모듈입니다. 기울기센서 내부에서는 수은이 기울기에 따라 스위치로 동작을 하고, 기울기가 인식되면 LED가 켜지게 되어있습니다. 2개를 한 쌍으로 구성 되어있습니다.

모듈이 파손될 경우 수은독성에 노출 될 수 있습니다. 주의가 필요합니다.



PIN	이름	기 능
1	L	아날로그 입력(PWM)
2	S	Signal(디지털 출력)
3	+	VCC
4	G	GND

○ 연결도



○ 소스코드

```
//*****************************************************************************
// Project : 25_2PCS_LIGHT_CUP in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int LedPinA = 6;
int LedPinB = 4;
int ButtonPinA = 7;
```

```
int ButtonPinB = 5;
int buttonStateA = 0;
int buttonStateB = 0;
int brightness = 0;
void setup ()
{
  pinMode (LedPinA, OUTPUT);
  pinMode (LedPinB, OUTPUT);
  pinMode (ButtonPinA, INPUT);
  pinMode (ButtonPinB, INPUT);
}
void loop ()
{
  buttonStateA = digitalRead (ButtonPinA);
  if (buttonStateA == HIGH && brightness != 255)
  {
    brightness++;
  }
  buttonStateB = digitalRead (ButtonPinB);
  if (buttonStateB == HIGH && brightness != 0)
  {
    brightness--;
  }
  analogWrite (LedPinA, brightness);
  analogWrite (LedPinB, 255 - brightness);
  delay (25);
}
```

○ 결과확인

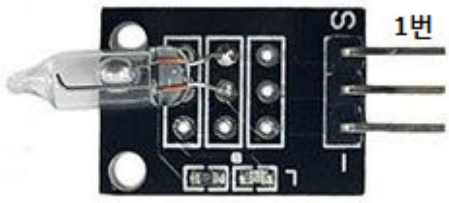
한 개 모듈이 기울어 지면 반대쪽 모듈에 LED 가 켜집니다.

□ 각도(틸트)스위치 모듈

○ 모듈개요

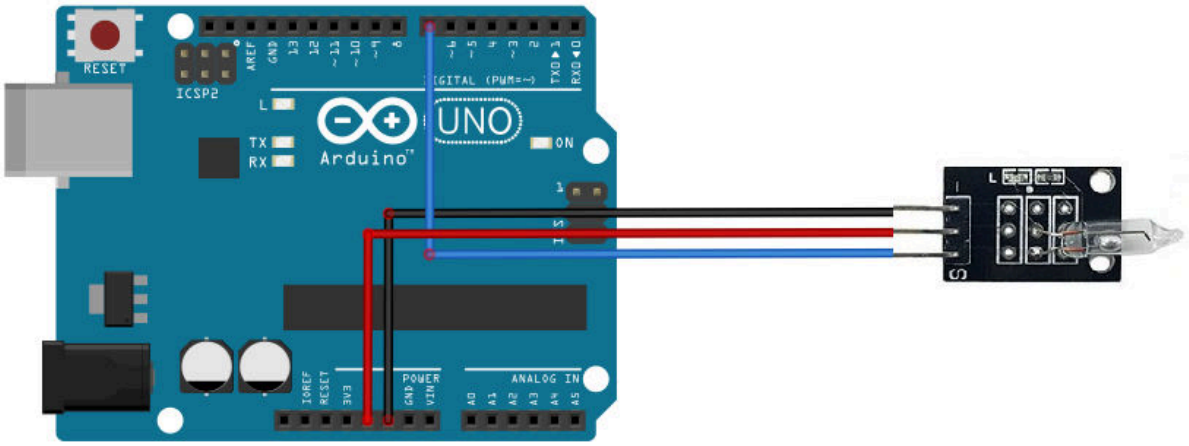
각도(Tilt)스위치 모듈은 기울기를 감지하는 센서모듈입니다. 센서가 기울어질 경우 내부 수은에 의해서 스위치 상태를 변화 시킵니다.

* 모듈이 파손될 경우 수은독성에 노출 될 수 있습니다. 주의가 필요합니다.



PIN	이름	기 능
1	S	Signal(디지털 출력)
2		5V
3	-	GND

○ 연결도



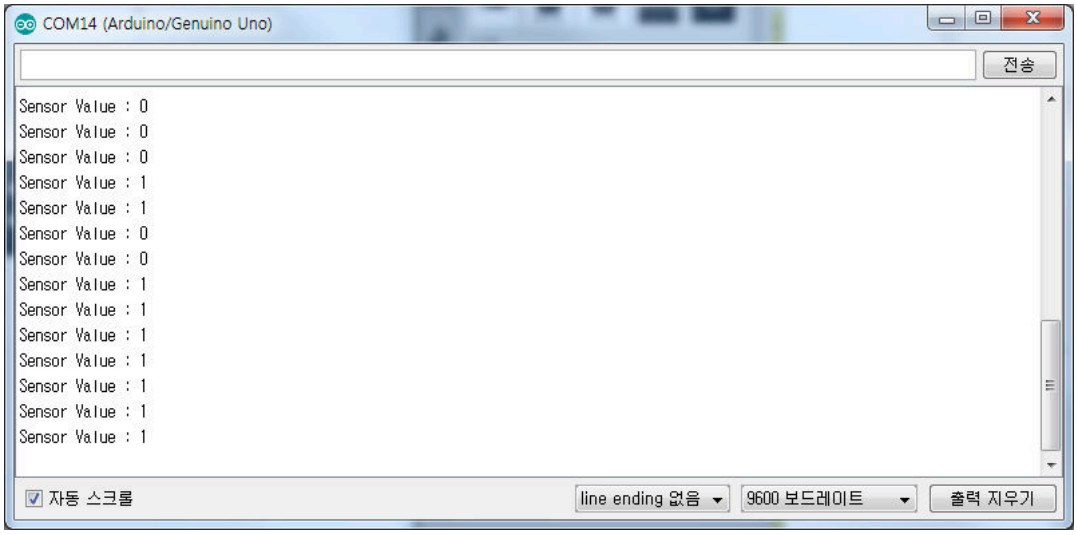
○ 소스코드

```
//*********************************************************************************
// Project : 26_TILD_SWITCH in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
//int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
```

```
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
void loop ()
{
  gValue = digitalRead(DigitalPin);

  if (gValue){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

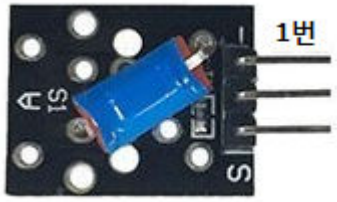
○ 결과확인



□ 볼스위치 모듈

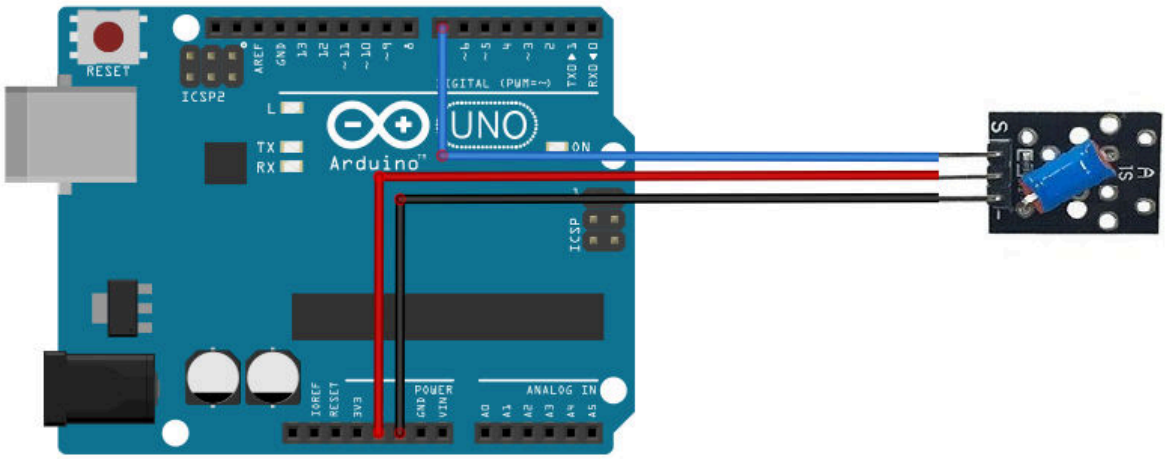
○ 모듈개요

볼스위치 모듈은 기울기를 감지하는 센서모듈입니다. 내부의 금속 볼이 기울기에 따라 스위치가 열고 닫힌 상태를 출력합니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(디지털 출력)

○ 연결도



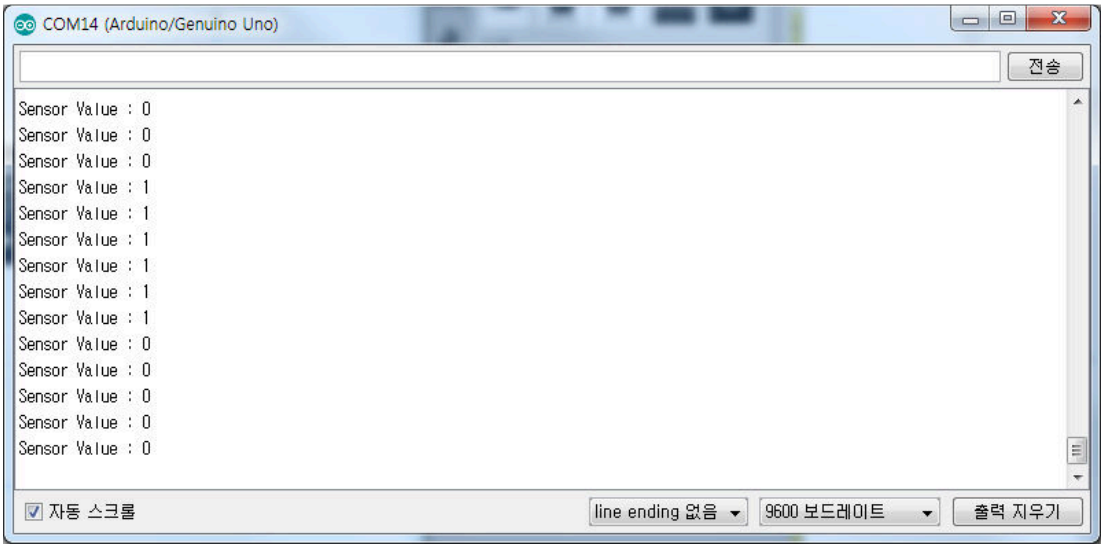
○ 소스코드

```
//*********************************************************************************
// Project : 27_BALL SWITCH in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
//int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
```

```
}
void loop ()
{
  gValue = digitalRead(DigitalPin);

  if (gValue){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

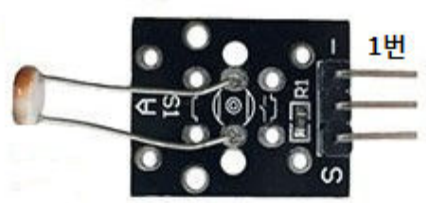
○ 결과확인



□ 광센서(CDS)센서 모듈

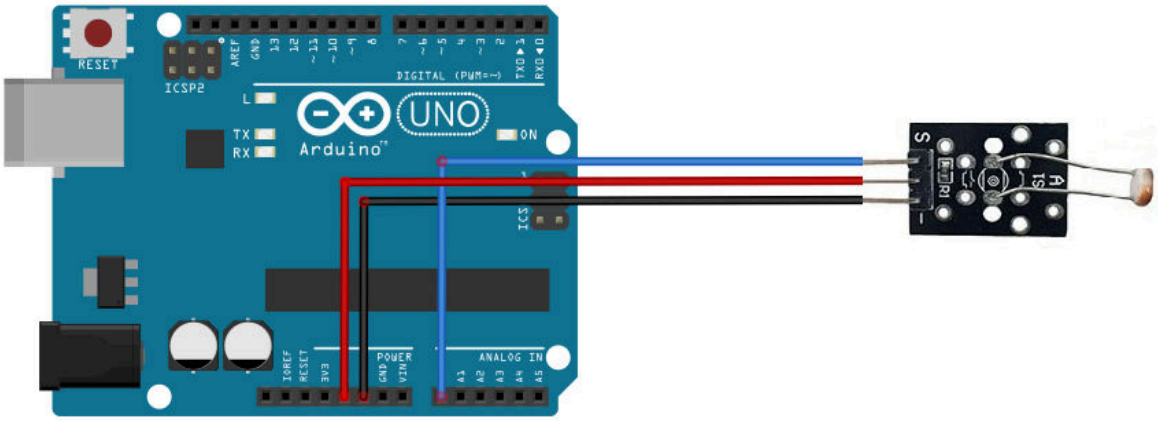
○ 모듈개요

광센서(CDS)센서 모듈은 빛의 밝기를 측정할 수 있는 센서모듈입니다. 빛에 양에 따라 저항값이 변화하는 CDS(황화카드뮴)소자로 구성되어있습니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(아날로그 출력)

○ 연결도



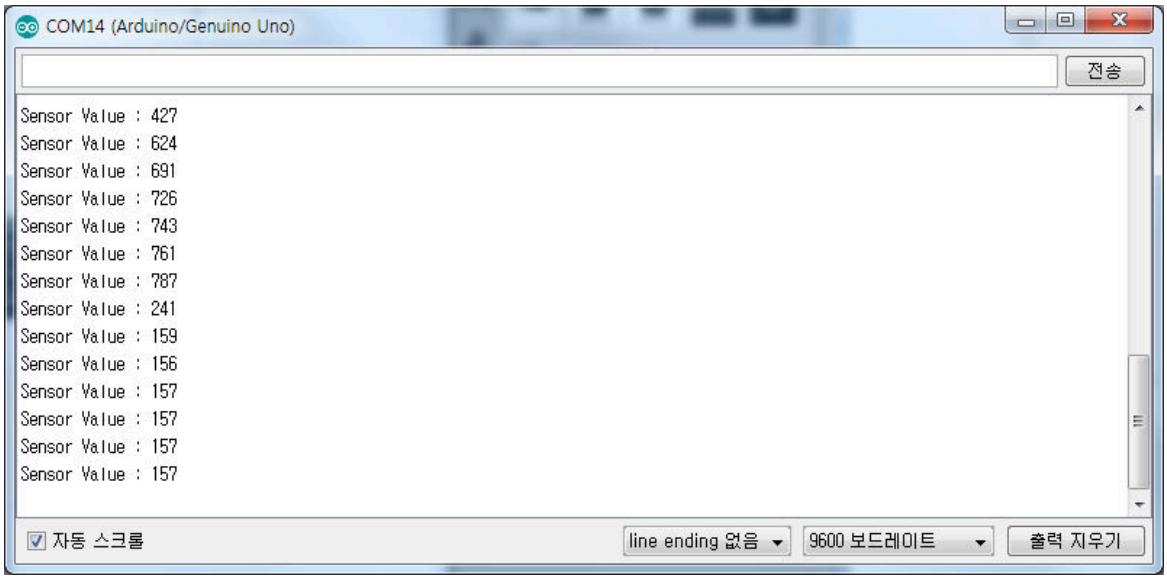
○ 소스코드

```
//*****************************************************************************
// Project : 28_PHOTOR_RESISTOR in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
//  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
void loop ()
```

```
{
  gValue = analogRead(AnalogPin);

  if (gValue>500)   {
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인



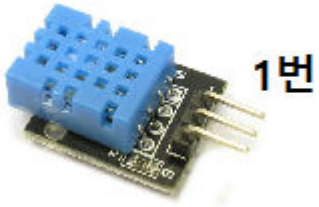
□ 온습도센서 모듈

○ 모듈개요

온습도센서모듈은 온도와 습도가 측정이 가능한 센서모듈입니다. DHT11이라고도 하며 입출력은 디지털 1-Wire로 되어있습니다. 오차율은 온도 2도, 습도 5% 입니다.

*제조사에 따라 또는 외부 모양에 따라 핀맵이 다를 수 있습니다.

1)



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(디지털 입출력)

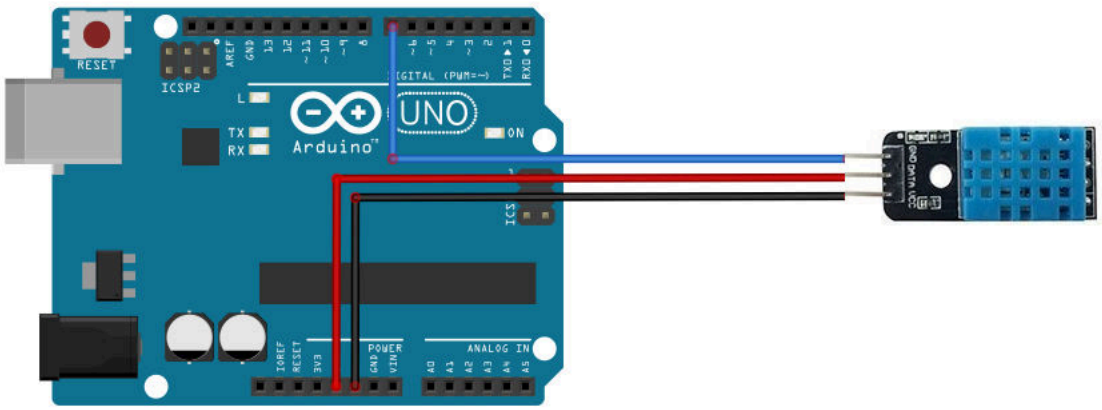
2)



PIN	이름	기 능
1	VCC	5V
2	DATA	Signal(디지털 입출력)
3	GND	GND

○ 연결도

*연결도는 1) 기준으로 작성이 되어있습니다.



○ 소스코드

```
//*****************************************************************************
// Project : 29_TEMP_AND_HUMIDITY in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
int DHT11pin = 7;
unsigned char dat[5];
unsigned char read_DHT11 () {
  unsigned char data;
  for (int i = 0; i < 8; i ++) {
    if (digitalRead (DHT11pin) == LOW) {
      while (digitalRead (DHT11pin) == LOW); // wait for 50us
      delayMicroseconds (30); // determine the duration of the high level to
determine the data is '0 'or '1'
      if (digitalRead (DHT11pin) == HIGH)
        data |= (1 << (7-i)); // high front and low in the post
      while (digitalRead (DHT11pin) == HIGH); // data '1 ', wait for the next one
receiver
     }
  }
return data;
}

void start_read () {
  digitalWrite (DHT11pin, LOW); // start signal
  delay (20);

  digitalWrite (DHT11pin, HIGH);
  delayMicroseconds (40);  // DHT11 response

  pinMode (DHT11pin, INPUT);
  while (digitalRead (DHT11pin) == HIGH);
  delayMicroseconds (80);
  if (digitalRead (DHT11pin) == LOW);
  delayMicroseconds (80);

  for (int i = 0; i < 4; i ++) // receive temperature and humidity data
    dat[i] = read_DHT11 ();

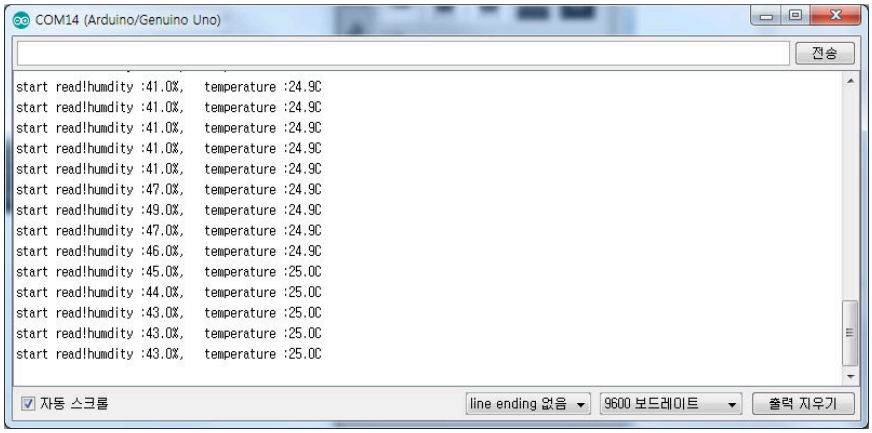
  pinMode (DHT11pin, OUTPUT);
  digitalWrite (DHT11pin, HIGH); // Stop signal
}

void setup () {
  Serial.begin (9600);
  pinMode (DHT11pin, OUTPUT);
}

void loop () {
  Serial.print ("start read!");
  start_read ();
  Serial.print ("humdity :");
  Serial.print (dat [0], DEC); // display  humidity-bit;
  Serial.print ('.');
  Serial.print (dat [1], DEC); // display humidity decimal ;
  Serial.print ("%,   ");
  Serial.print ("temperature :");
  Serial.print (dat [2], DEC); // display  temperature-bits;
  Serial.print ('.');
```

```
  Serial.print (dat [3], DEC); // display  temperature  decimal ;
  Serial.println ('C');
  delay (2000);
}
```

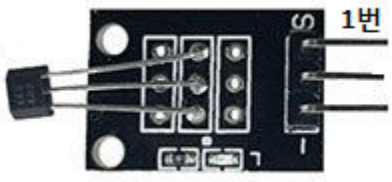
○ 결과확인



□ 아날로그 홀센서 모듈

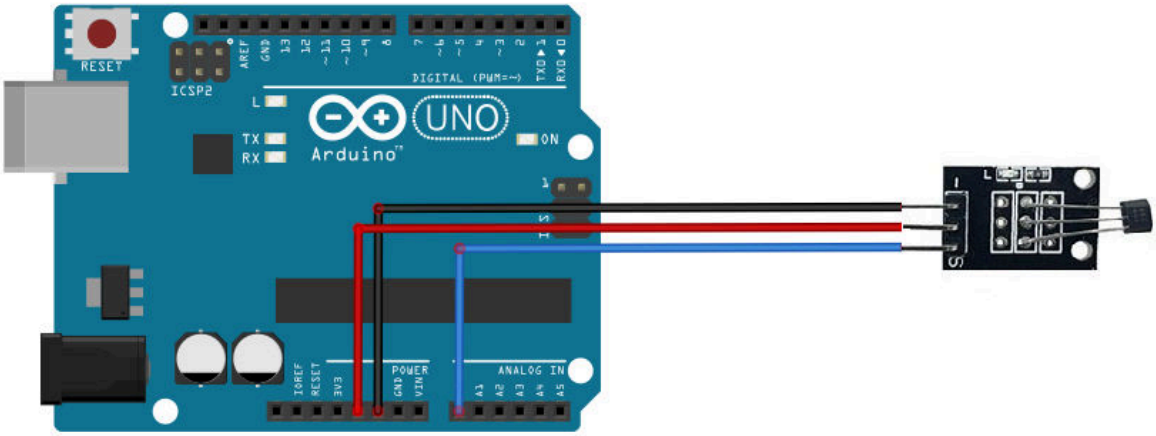
○ 모듈개요

아날로그 홀센서모듈은 자석(자기장)을 감지할 수 있는 센서모듈입니다. 도어 출입감지시스템으로 응용될 수 있습니다. 결과는 아날로그 값으로 출력이 됩니다.



PIN	이름	기 능
1	S	Signal(아날로그 출력)
2		5V
3	-	GND

○ 연결도



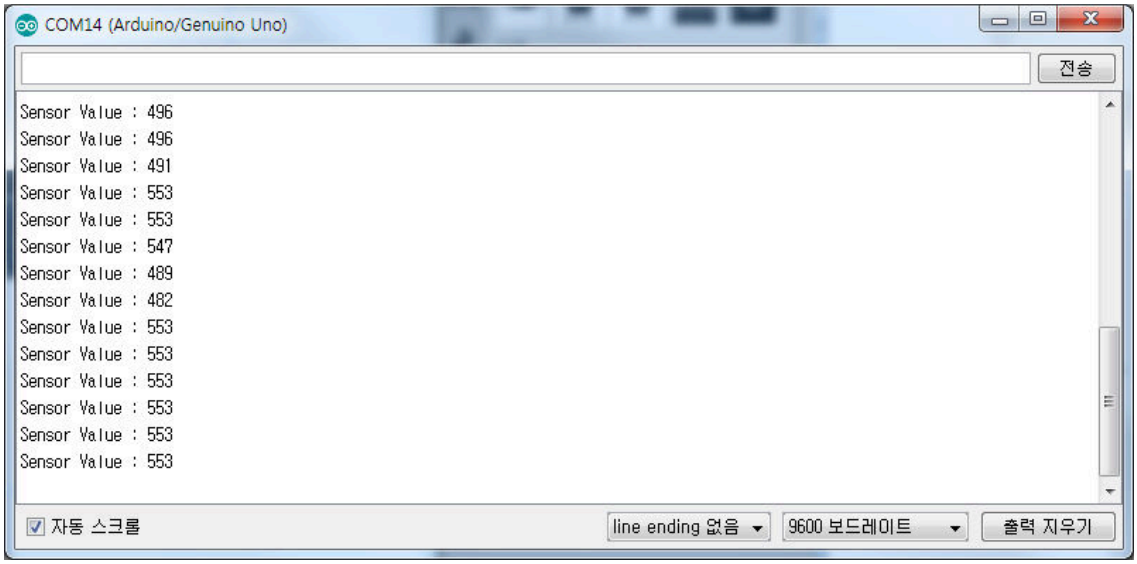
○ 소스코드

```
//*********************************************************************************
// Project : 30_ANALOG_HALL in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int AnalogPin = A0;    // Analog input
//int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  //pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
void loop ()
{
```

```
  gValue = analogRead(AnalogPin);

  if (gValue>500){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

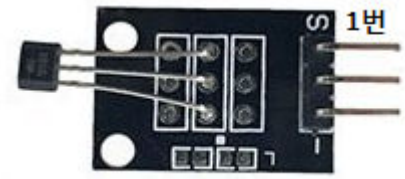
○ 결과확인



□ 마그네틱 홀센서 모듈

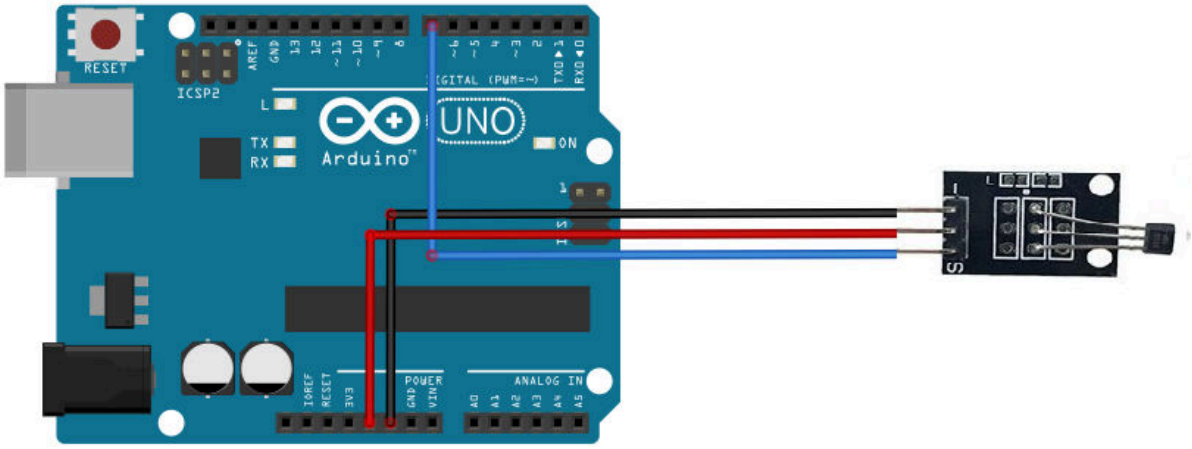
○ 모듈개요

마그네틱홀센서모듈은 자석(자기장)을 감지하는 센서모듈입니다. 결과는 디지털 신호를 출력이 됩니다.



PIN	이름	기 능
1	S	Signal(디지털 출력)
2		5V
3	-	GND

○ 연결도



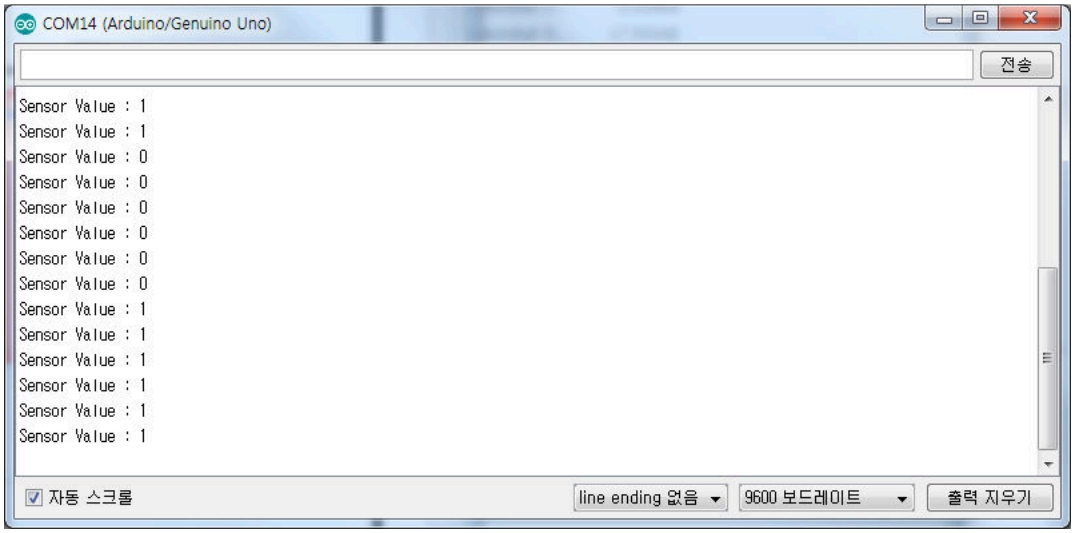
○ 소스코드

```
//************************************************************************************
// Project : 31_HALL_MAGNETIC in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************************
//int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
```

```
void loop ()
{
  gValue = digitalRead(DigitalPin);

  if (gValue)   {
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

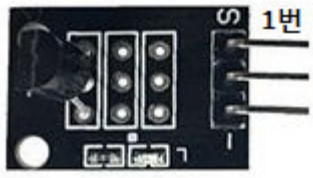
○ 결과확인



□ 온도센서 모듈

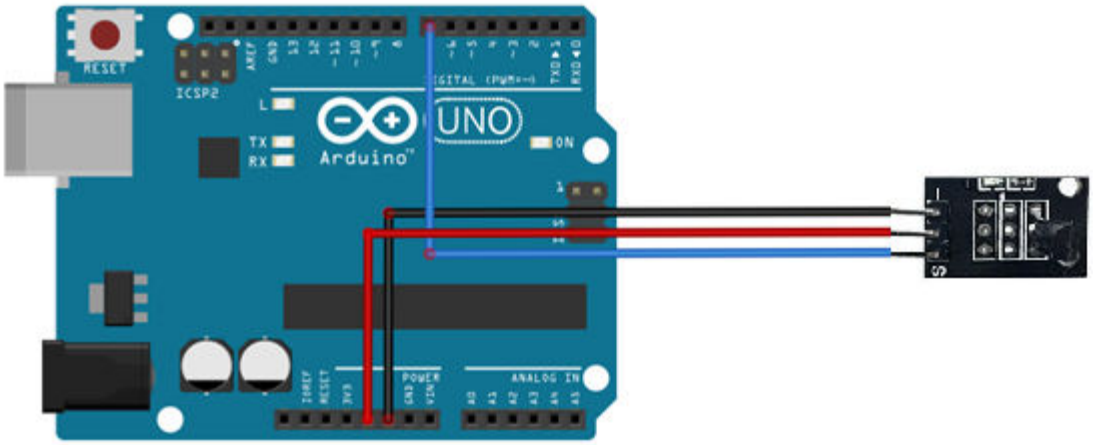
○ 모듈개요

온도센서모듈은 디지털온도센서모듈입니다. 온도센서칩은 DS18B20로 되어있으며, 인터페이스는 I2C로 통신이 가능합니다. 측정 온도 범위는 -55~+ 125도 입니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(디지털 입출력)

○ 연결도



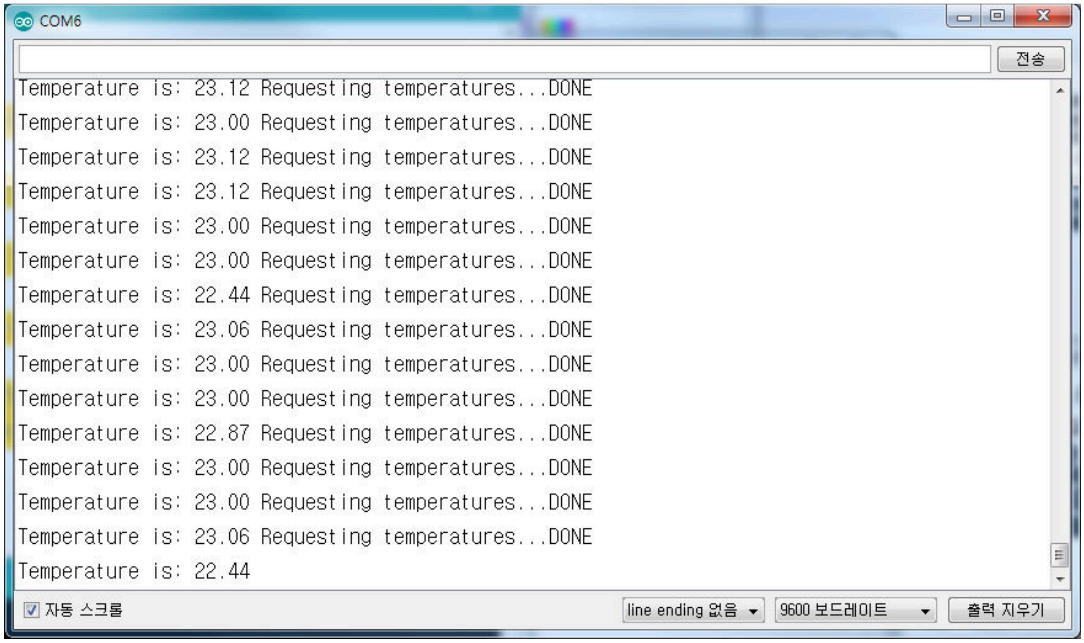
○ 소스코드

OneWire / DallasTemperature 라이브러리 설치 필요

```
//*****************************************************************************
// Project : 32_Temp sensor in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
/*****************************************************************/
// First we include the libraries
#include <OneWire.h>
#include <DallasTemperature.h>
/*****************************************************************/
// Data wire is plugged into pin 2 on the Arduino
#define ONE_WIRE_BUS 7
/*****************************************************************/
// Setup a oneWire instance to communicate with any OneWire devices
// (not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);
/*****************************************************************/
// Pass our oneWire reference to Dallas Temperature.
```

```
DallasTemperature sensors(&oneWire);
/*********************************************************************/
void setup(void)
{
 // start serial port
 Serial.begin(9600);
 Serial.println("Dallas Temperature IC Control Library Demo");
 // Start up the library
 sensors.begin();
}
void loop(void)
{
 // call sensors.requestTemperatures() to issue a global temperature
 // request to all devices on the bus
/*********************************************************************/
 Serial.print(" Requesting temperatures...");
 sensors.requestTemperatures(); // Send the command to get temperature readings
 Serial.println("DONE");
/*********************************************************************/
 Serial.print("Temperature is: ");
 Serial.print(sensors.getTempCByIndex(0)); // Why "byIndex"?
   // You can have more than one DS18B20 on the same bus.
   // 0 refers to the first IC on the wire
   delay(1000);
}
```

○ 결과확인

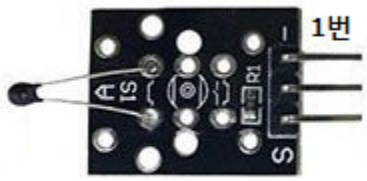


□ 아날로그 온도센서 모듈

○ 모듈개요

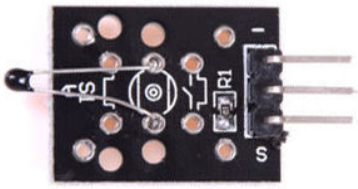
아날로그 온도센서모듈은 NTC 서미스터를 사용하는 온도센서모듈입니다. 온도가 상승하게 되면 저항이 증가하는 방식으로 전압분배 방식으로 아날로그가 출력이 됩니다.

1)



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(아날로그 출력)

2)

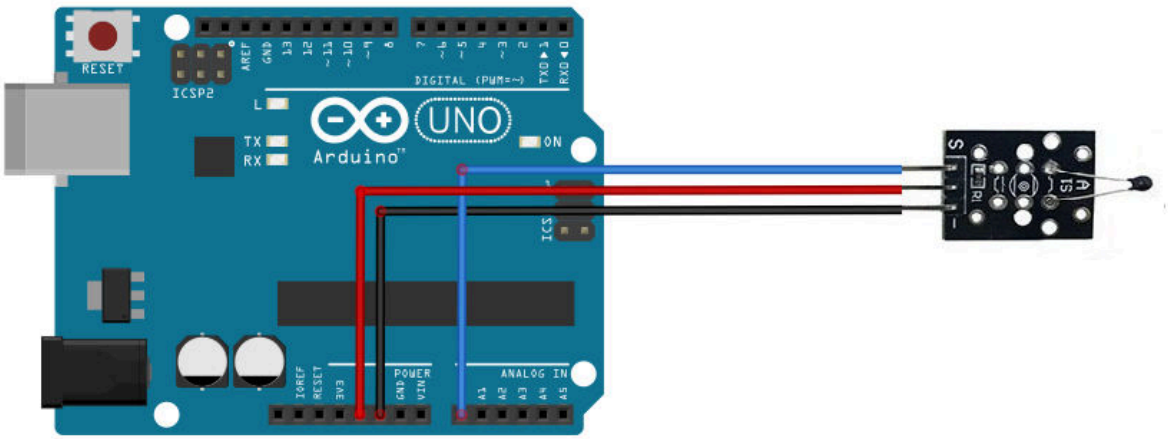


PIN	이름	기 능
1	S	Signal(아날로그 출력)
2	GND	GND
3	VCC	5V

○ 연결도

아래 연결도는 1)번 아날로그 온도센서 모듈 모양에 맞춰 되어있습니다.

2)번의 모양의 모듈일 경우 핀맵을 변경하셔야 합니다.

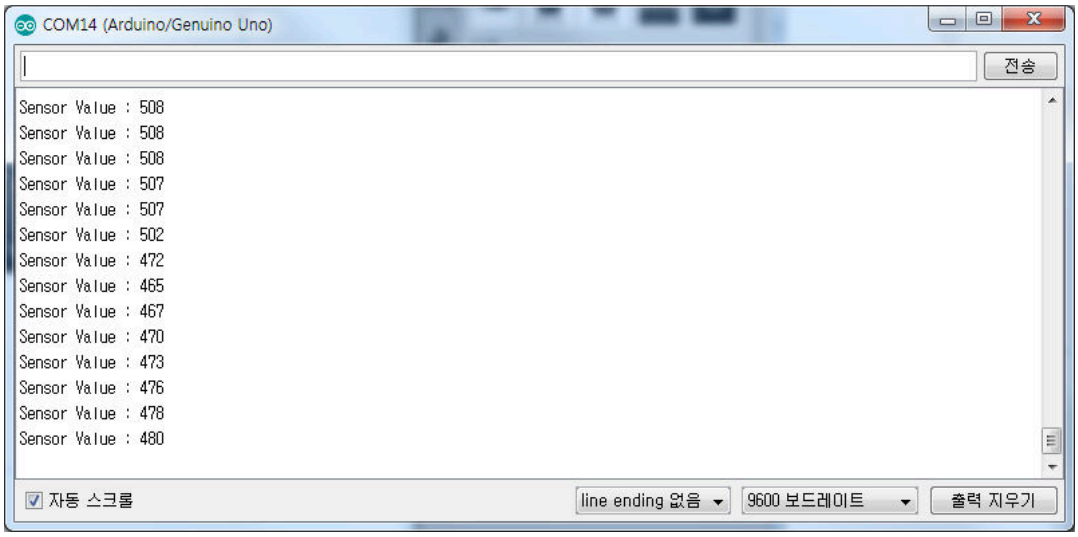


○ 소스코드

```
//*********************************************************************************
// Project : 33_ANALOG_TEMP in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*********************************************************************************
int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
//  pinMode (DigitalPin, INPUT);
  Serial.begin (9600);
}
void loop ()
{
  gValue = analogRead(AnalogPin);

  if (gValue>500)   {
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

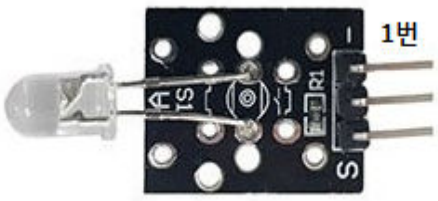
○ 결과확인



□ IR 적외선 송신 모듈

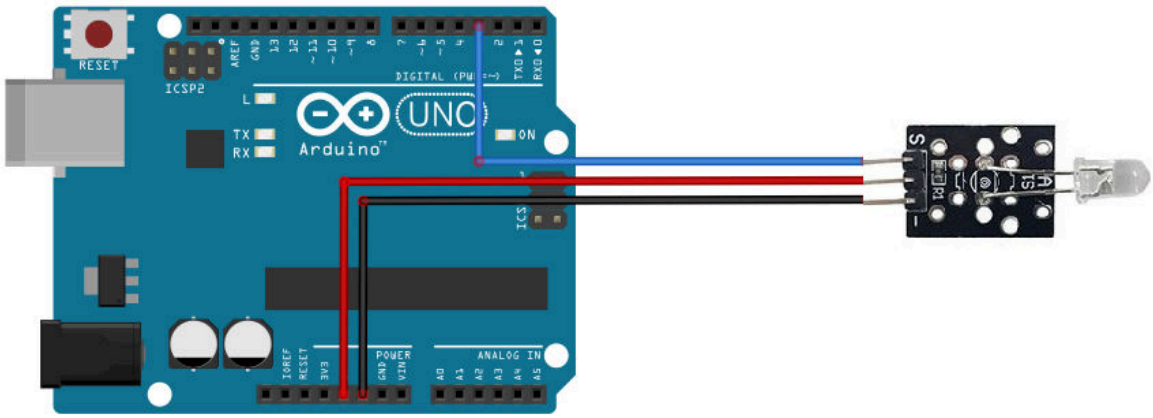
○ 모듈개요

IR적외선송신모듈은 적외선을 송신하는 모듈입니다. 리모콘 또는 장난감조정기 등에 응용할 수 있습니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(디지털 입력)

○ 연결도



○ 소스코드

```
//*****************************************************************************
// Project : 34_IR_EMISSION in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//*****************************************************************************
#include <IRremote.h>
IRsend irsend;
void setup ()
{
  Serial.begin (9600);
}
void loop () {
  for (int i = 0; i <50 ; i++) {
    irsend.sendSony(0xa90, 12); // Sony TV power code
    delay (40);
```

```
  }
}
```

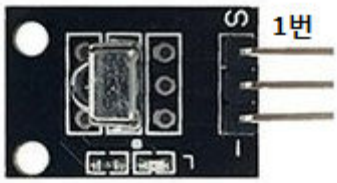
○ 결과확인

다음장 IR 수신 센서에서 확인

□ IR 적외선 수신 모듈

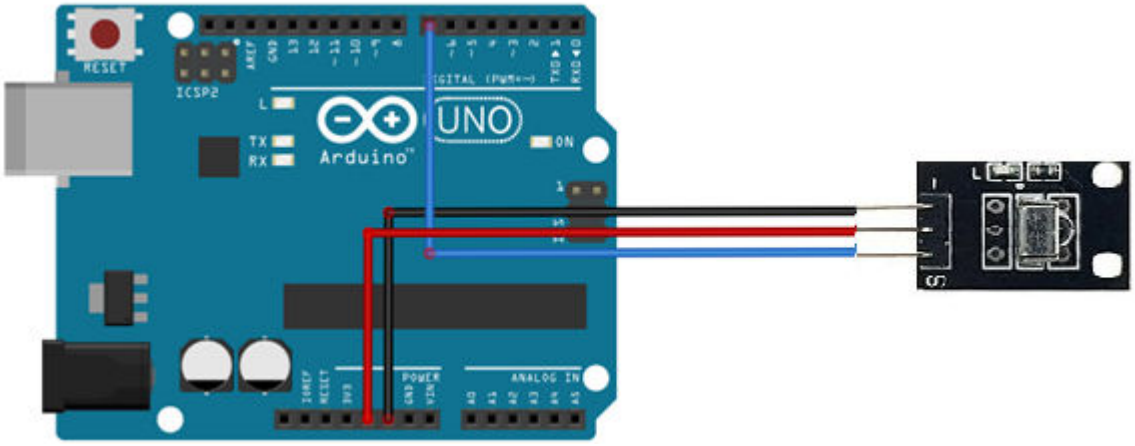
○ 모듈개요

IR적외선수신모듈은 적외선을 수신할 수 있는 모듈입니다. IR적외선 송신모듈을 활용해서 동작을 확인하실 수 있습니다.



PIN	이름	기 능
1	S	Signal(디지털 출력)
2		5V
3	-	GND

○ 연결도

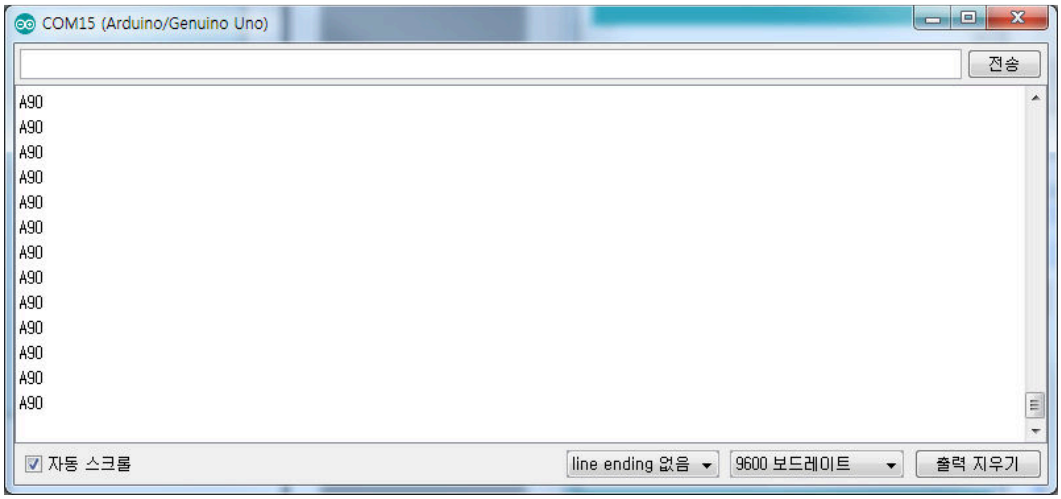


○ 소스코드

```
//************************************************************************************
// Project : 35_IR_RECEIVER in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************************
#include <IRremote.h>
int DigitalPin = 7;  // Digital Input
IRrecv irrecv (DigitalPin);
decode_results results;
void setup ()
{
    Serial.begin (9600);
    irrecv.enableIRIn (); // Start the receiver
}
void loop () {
    if (irrecv.decode (&results)) {
        Serial.println (results.value, HEX);
        irrecv.resume (); // Receive the next value
    }
```

```
}
```

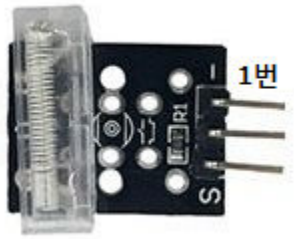
○ 결과확인


COM15 (Arduino/Genuino Uno)
전송
A90
A90
A90
A90
A90
A90
A90
A90
A90
A90
A90
A90
자동 스크롤
line ending 없음
9600 보드레이트
출력 지우기

□ 노크센서 모듈

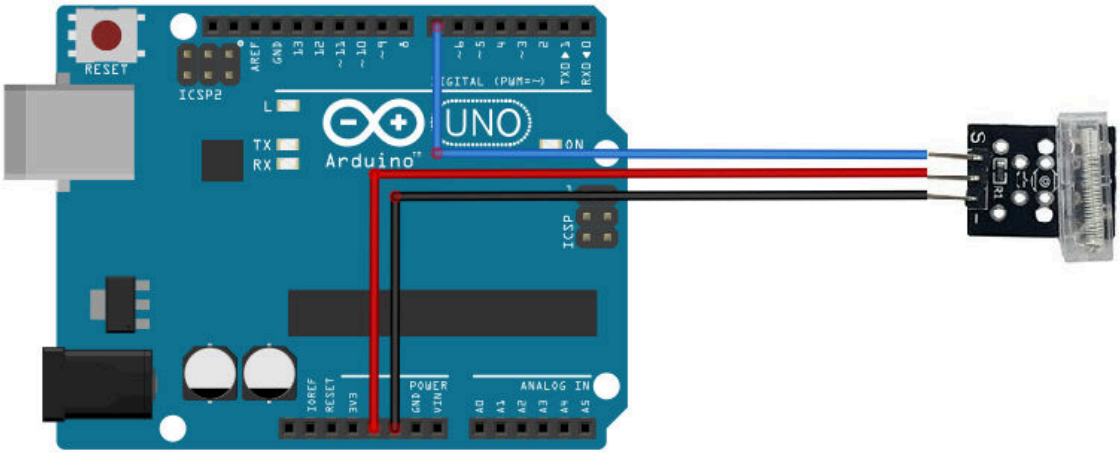
○ 모듈개요

노크센서모듈는 진동을 감지할 수 있는 센서모듈입니다. 센서 내부에 스프링이 진동을 감지할 경우 흔들림이 발생되어 출력신호와 접점이 되는 원리입니다.



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(디지털 출력)

○ 연결도



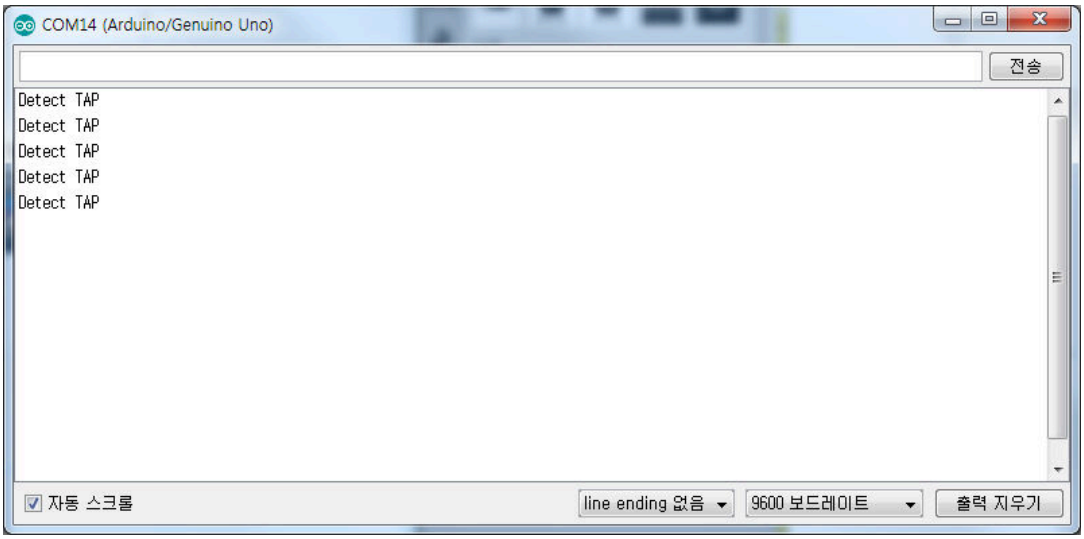
○ 소스코드

```
//********************************************************************************
// Project : 36_TAP_MODULE  in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//********************************************************************************
//int AnalogPin = A0;    // Analog input
int DigitalPin = 7;  // Digital input
int LedPin = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup ()
{
  pinMode (LedPin, OUTPUT);
  pinMode (DigitalPin, INPUT);
```

```
  Serial.begin (9600);
}
void loop ()
{
  gValue = digitalRead(DigitalPin);

  if (gValue){
    digitalWrite (LedPin, HIGH);
  }else {
    digitalWrite (LedPin, LOW);
    Serial.println ("Detect TAP");
     delay(1000);
  }
}
```

○ 결과확인



□ 포토인터럽터센서모듈

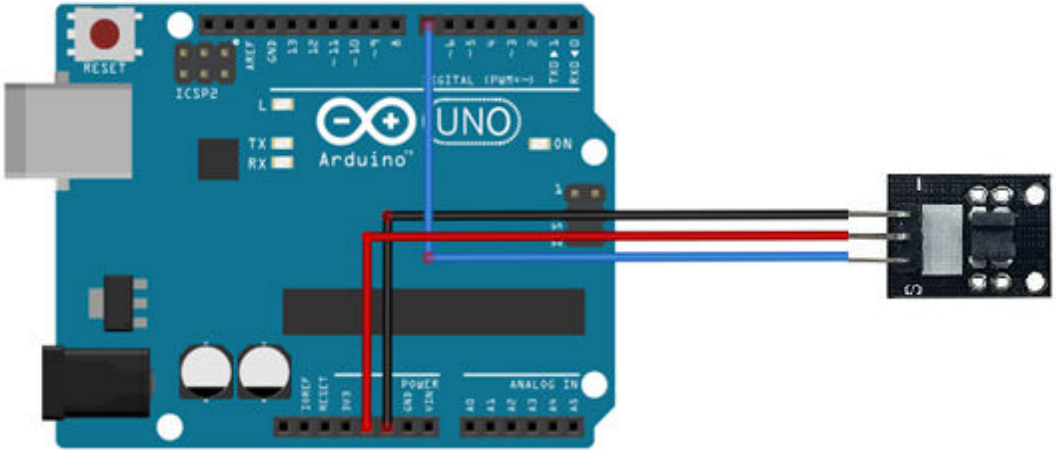
○ 모듈개요

포터인터럽터센서모듈은 물체의 유무 검출을 할 수 있는 센서모듈입니다. 적외선 발광, 수광 소자가 서로 마주보는 구조 되어 있어 물체가 발광부에서 출력되는 적외선을 차단할 경우 물체의 유무를 확인할 수 있습니다. 공장 자동화 등에 응용할 수 있습니다..



PIN	이름	기 능
1	-	GND
2		5V
3	S	Signal(디지털 출력)

○ 연결도

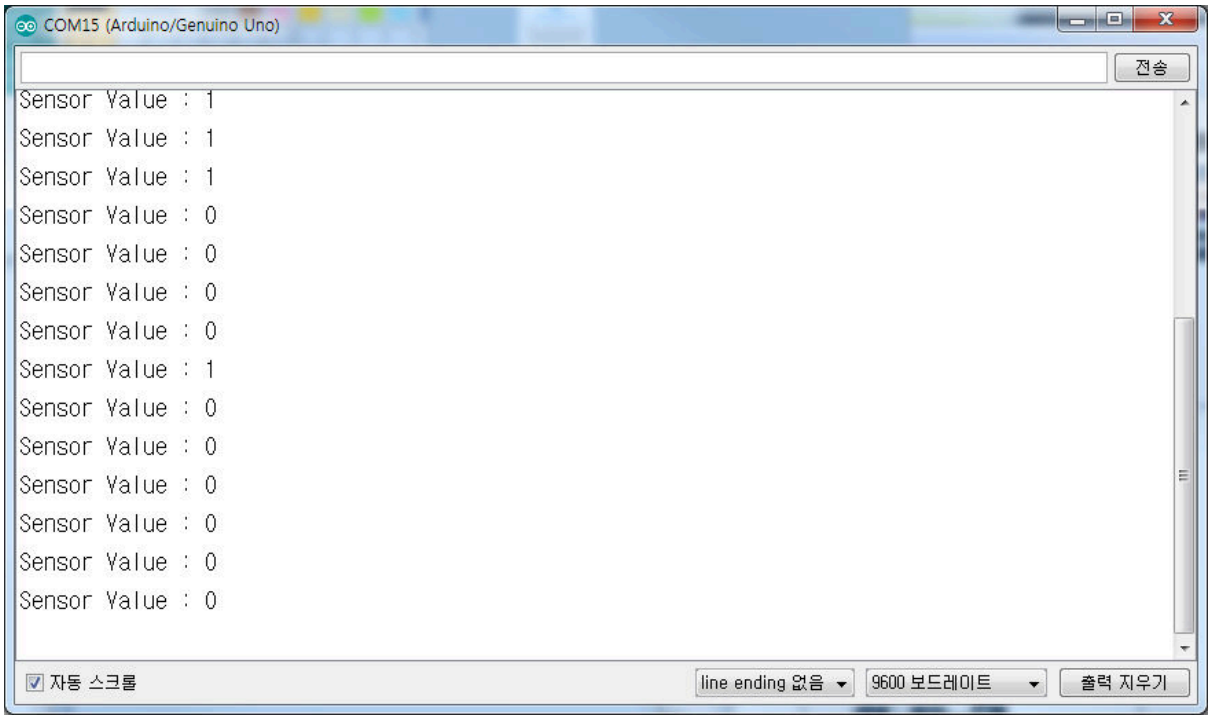


○ 소스코드

```
//************************************************************************************
// Project : 37_LIGHT_BLOCKING in Sensor Kit
// Board : Arduino Uno
// By : Kit Plus
//************************************************************************************
int DigitalPin =7;  // Digital input
int Led = 13;     // select the pin for the LED
int gValue = 0;     // variable for Sensor Data
void setup()
{
  Serial.begin (9600);
  pinMode(Led, OUTPUT); // LED pin as output
  pinMode(DigitalPin, INPUT); //photo interrupter pin as input
}
void loop()
{
  gValue=digitalRead(DigitalPin); //read the value of the sensor
```

```
  if(gValue == HIGH) // turn on LED when sensor is blocked
  {
    digitalWrite(Led,HIGH);
  }
  else
  {
    digitalWrite(Led,LOW);
  }
  Serial.print ("Sensor Value : ");
  Serial.println (gValue, DEC);
  delay(500);
}
```

○ 결과확인



□ 참고자료출처

https://www.arduino.cc/en/Main/Education

https://arduinomodules.info/category/switches-input/

http://fritzing.org/download/

* 본 문서의 저작권은 출처를 밝히고 비영리 목적으로 배포/활용 가능합니다.